

FLTK 1.5.0 Programming Manual



By F. Costantini, M. Melcher,
A. Schlosser, B. Spitzak, and M. Sweet.

Copyright © 1998 - 2025 by Bill Spitzak and others.

This software and manual are provided under the terms of the GNU Library General Public License.
Permission is granted to reproduce this manual or any portion for any purpose,
provided this copyright and permission notice are preserved.

Generated by Doxygen 1.12.0

March 10, 2025

Git revision 8838e976a0b87279e5a98e560ed3d6ef20dc9139

1 FLTK Programming Manual	1
1.1 Preface	2
1.1.1 Organization	2
1.1.2 Conventions	3
1.1.3 Abbreviations	4
1.1.4 Copyrights and Trademarks	4
1.2 Introduction to FLTK	4
1.2.1 History of FLTK	5
1.2.2 Features	6
1.2.3 Licensing	7
1.2.4 What Does "FLTK" Mean?	7
1.2.5 FLUID	7
1.2.6 Building and Installing FLTK with CMake	7
1.2.7 Building FLTK Under Microsoft Windows	8
1.2.7.1 Free and Commercial Microsoft Visual Studio Versions	8
1.2.7.2 Using the Visual C++ DLL Library	8
1.2.7.3 GNU toolsets (Cygwin or MinGW) hosted on Windows	9
1.2.8 Internet Resources	9
1.2.9 Reporting Bugs	10
1.3 FLTK Basics	10
1.3.1 Writing Your First FLTK Program	10
1.3.1.1 Creating the Widgets	11
1.3.1.2 Creating Widget Hierarchies	11
1.3.1.3 Get/Set Methods	12
1.3.1.4 Redrawing After Changing Attributes	12
1.3.1.5 Labels	12
1.3.1.6 Showing the Window	12
1.3.1.7 The Main Event Loop	12
1.3.2 Naming Conventions	13
1.3.3 Header Files	13
1.3.4 Compiling Programs that Use FLTK	13
1.3.4.1 Compiling Programs with Standard Compilers	13
1.3.4.2 Compiling Programs with the 'fltk-config' Script	14
1.3.4.3 Compiling Multiple Source Files with 'fltk-config'	15
1.3.4.4 Compiling Programs with Makefiles	15
1.3.4.5 Compiling Programs with Microsoft Visual C++	16
1.4 Common Widgets and Attributes	16
1.4.1 Buttons	16
1.4.2 Text	17
1.4.3 Valuers	18
1.4.4 Groups	19
1.4.5 Setting the Size and Position of Widgets	19

1.4.6 Colors	20
1.4.7 Box Types	21
1.4.7.1 Making Your Own Boxtypes	21
1.4.8 Labels and Label Types	22
1.4.9 Callbacks	27
1.4.10 When and Reason	28
1.4.11 Shortcuts	28
1.5 Coordinates and Layout Widgets	28
1.5.1 The Widget Coordinate System	28
1.5.2 Layout and Container Widgets	29
1.5.2.1 The FI_Flex Layout Widget	29
1.5.2.2 The FI_Grid Layout Widget	30
1.5.2.3 The FI_Pack Layout Widget	31
1.5.2.4 The FI_Scroll Container Widget	31
1.5.2.5 The FI_Tabs Container Widget	32
1.5.2.6 The FI_Tile Layout Widget	32
1.5.2.7 The FI_Wizard Container Widget	32
1.6 How Does Resizing Work?	33
1.6.1 Resizing can be disabled	33
1.6.2 Resizing can be simple	33
1.6.3 Resizing can be complex	34
1.6.4 Practical examples	35
1.7 Designing a Simple Text Editor	38
1.7.1 Determining the Goals of the Text Editor	38
1.7.2 Chapter 1: A Minimal App	39
1.7.3 Chapter 2: Adding a Menu Bar	39
1.7.4 Chapter 3: Adding a Text Editor widget	41
1.7.5 Chapter 4: Reading and Writing Files	42
1.7.6 Chapter 5: Cut, Copy, and Paste	44
1.7.7 Chapter 6: Find and Find Next	45
1.7.8 Chapter 7: Replace and Replace Next	45
1.7.9 Chapter 8: Editor Features	47
1.7.10 Chapter 9: Split Editor	48
1.7.11 Chapter 10: Syntax Highlighting	49
1.8 Drawing Things in FLTK	52
1.8.1 When Can You Draw Things in FLTK?	52
1.8.2 What Units Do FLTK Functions Use?	53
1.8.3 Drawing Functions	54
1.8.3.1 Boxes	55
1.8.3.2 Clipping	55
1.8.3.3 Colors	56
1.8.3.4 Color Contrast	59

1.8.3.5 Line Dashes and Thickness	60
1.8.3.6 Drawing Fast Shapes	61
1.8.3.7 Drawing Complex Shapes	63
1.8.3.8 Drawing Text	66
1.8.3.9 Fonts	68
1.8.3.10 Character Encoding	69
1.8.3.11 Drawing Overlays	69
1.8.4 Drawing Images	69
1.8.4.1 Direct Image Drawing	70
1.8.4.2 Direct Image Reading	71
1.8.4.3 Image Classes	72
1.8.5 Offscreen Drawing	73
1.9 Handling Events	74
1.9.1 The FLTK Event Model	74
1.9.2 Mouse Events	75
1.9.2.1 FL_PUSH	75
1.9.2.2 FL_DRAG	75
1.9.2.3 FL_RELEASE	76
1.9.2.4 FL_MOVE	76
1.9.2.5 FL_MOUSEWHEEL	76
1.9.3 Focus Events	76
1.9.3.1 FL_ENTER	76
1.9.3.2 FL_LEAVE	76
1.9.3.3 FL_FOCUS	76
1.9.3.4 FL_UNFOCUS	76
1.9.4 Keyboard Events	77
1.9.4.1 FL_KEYBOARD, FL_KEYDOWN, FL_KEYUP	77
1.9.4.2 FL_SHORTCUT	77
1.9.5 Widget Events	78
1.9.5.1 FL_DEACTIVATE	78
1.9.5.2 FL_ACTIVATE	78
1.9.5.3 FL_HIDE	78
1.9.5.4 FL_SHOW	78
1.9.6 Clipboard Events	78
1.9.6.1 FL_PASTE	78
1.9.6.2 FL_SELECTIONCLEAR	78
1.9.7 Drag and Drop Events	79
1.9.7.1 Dropped filenames	79
1.9.7.2 FL_DND_ENTER	79
1.9.7.3 FL_DND_DRAG	79
1.9.7.4 FL_DND_LEAVE	79
1.9.7.5 FL_DND_RELEASE	79

1.9.8 Other events	80
1.9.8.1 FL_SCREEN_CONFIGURATION_CHANGED	80
1.9.8.2 FL_FULLSCREEN	80
1.9.9 Fl::event_*() methods	80
1.9.10 Event Propagation	81
1.9.11 FLTK Compose-Character Sequences	82
1.10 Adding and Extending Widgets	82
1.10.1 Subclassing	82
1.10.2 Making a Subclass of Fl_Widget	82
1.10.3 The Constructor	83
1.10.4 Protected Methods of Fl_Widget	83
1.10.5 Handling Events	86
1.10.6 Drawing the Widget	87
1.10.7 Resizing the Widget	87
1.10.8 Making a Composite Widget	87
1.10.9 Cut and Paste Support	89
1.10.10 Drag And Drop Support	89
1.10.11 Making a subclass of Fl_Window	89
1.11 Using OpenGL	90
1.11.1 Using OpenGL in FLTK	90
1.11.2 Making a Subclass of Fl_Gl_Window	90
1.11.2.1 Defining the Subclass	90
1.11.2.2 The draw() Method	91
1.11.2.3 The handle() Method	91
1.11.3 OpenGL and support of HighDPI displays	92
1.11.4 Using OpenGL in Normal FLTK Windows	92
1.11.5 Using FLTK widgets in OpenGL Windows	93
1.11.6 OpenGL Drawing Functions	93
1.11.7 Speeding up OpenGL	94
1.11.8 Using OpenGL Optimizer with FLTK	95
1.11.9 Using OpenGL 3.0 (or higher versions)	96
1.12 FLTK Runtime Options	98
1.12.1 Runtime Options	98
1.12.2 Obtaining Current Settings	98
1.12.3 Administrative Tool	98
1.12.4 List of Options	99
1.13 Advanced FLTK	99
1.13.1 Multithreading	99
1.13.2 FLTK multithread locking - Fl::lock() and Fl::unlock()	100
1.13.3 Simple multithreaded examples using Fl::lock	101
1.13.4 FLTK multithreaded "lockless programming"	102
1.13.5 FLTK multithreaded Constraints	104

1.14 Unicode and UTF-8 Support	104
1.14.1 About Unicode, ISO 10646 and UTF-8	105
1.14.2 Unicode in FLTK	107
1.14.3 Illegal Unicode and UTF-8 Sequences	107
1.14.4 FLTK Unicode and UTF-8 Functions	108
1.14.5 FLTK Unicode Versions of System Calls	111
1.15 Constants and Enumerations	112
1.15.1 Version Numbers	112
1.15.2 Events	113
1.15.3 Callback "When" Conditions	113
1.15.4 Fl::event_button() Values	114
1.15.5 Fl::event_key() Values	114
1.15.6 Fl::event_state() Values	115
1.15.7 Alignment Values	116
1.15.8 Fonts	116
1.15.9 Colors	117
1.15.9.1 Color Constants	117
1.15.10 Cursors	118
1.15.11 FD "When" Conditions	119
1.15.12 Damage Masks	119
1.16 GLUT Compatibility	119
1.16.1 Using the GLUT Compatibility Header File	119
1.16.2 Known Problems	120
1.16.3 Mixing GLUT and FLTK Code	121
1.16.4 class Fl_Glut_Window	121
1.16.4.1 Class Hierarchy	121
1.16.4.2 Include Files	121
1.16.4.3 Description	121
1.16.4.4 Members	121
1.16.4.5 Methods	122
1.17 Forms Compatibility	122
1.17.1 Importing Forms Layout Files	123
1.17.2 Using the Compatibility Header File	123
1.17.3 Problems You Will Encounter	123
1.17.4 Additional Notes	125
1.18 Operating System Issues	127
1.18.1 Accessing the OS Interfaces	127
1.18.2 The Wayland/X11 hybrid library	127
1.18.3 The UNIX (X11) Interface	128
1.18.3.1 Handling Other X Events	128
1.18.3.2 Drawing using Xlib	130
1.18.3.3 Changing the Display, Screen, or X Visual	130

1.18.3.4 Using a Subclass of <code>Fl_Window</code> for Special X Stuff	132
1.18.3.5 Setting the Icon of a Window	134
1.18.3.6 X Resources	134
1.18.3.7 Display Scaling Factor	135
1.18.4 The Windows Interface	135
1.18.4.1 Using filenames with non-ASCII characters	135
1.18.4.2 Responding to <code>WM_QUIT</code>	135
1.18.4.3 Handling Other Windows API Messages	135
1.18.4.4 Drawing Things Using the Windows GDI	136
1.18.4.5 HighDPI support	136
1.18.4.6 Display Scaling Factor	136
1.18.4.7 Setting the Icon of a Window	136
1.18.4.8 How to Not Get a MSDOS Console Window	137
1.18.4.9 Known Windows Bugs and Problems	137
1.18.5 The Apple OS X Interface	137
1.18.5.1 Setting the icon of an application	139
1.18.5.2 Drawing Things Using Quartz	139
1.18.5.3 Internationalization	140
1.18.5.4 OpenGL and 'retina' displays	140
1.18.5.5 <code>Fl_Double_Window</code>	141
1.18.5.6 Mac File System Specifics	141
1.18.6 The Wayland Interface	141
1.18.6.1 HiDPI display support	142
1.18.6.2 Window icons	143
1.18.6.3 Window titlebars	143
1.19 Migrating Code from FLTK 1.3 to 1.4	143
1.19.1 Changes in Header Files	144
1.19.2 <code>Fl_Preferences</code>	144
1.19.3 <code>Fl::add_timeout</code> and friends	145
1.19.4 New <code>FL_OVERRIDE</code> Macro	145
1.19.5 <code>Fl_Image::copy()</code> 'const'	145
1.19.6 Using X11 specific code with a "hybrid" FLTK library	146
1.19.7 Modern CMake	147
1.19.8 New <code>FL_HELVETICA</code> Font on Windows	148
1.20 Software License	149
1.21 Example Source Code	154
1.21.1 Example Applications: Overview	154
1.21.1.1 <code>adjuster</code>	154
1.21.1.2 <code>animated</code>	154
1.21.1.3 <code>arc</code>	154
1.21.1.4 <code>ask</code>	155
1.21.1.5 <code>bitmap</code>	155

1.21.1.6 blocks	155
1.21.1.7 boxtype	155
1.21.1.8 browser	155
1.21.1.9 button	155
1.21.1.10 buttons	155
1.21.1.11 cairo_test	155
1.21.1.12 checkers	155
1.21.1.13 clipboard	155
1.21.1.14 clock	156
1.21.1.15 colbrowser	156
1.21.1.16 color_chooser	156
1.21.1.17 cube	156
1.21.1.18 CubeView	156
1.21.1.19 cursor	156
1.21.1.20 curve	156
1.21.1.21 demo	156
1.21.1.22 device	156
1.21.1.23 doublebuffer	157
1.21.1.24 editor	157
1.21.1.25 fast_slow	157
1.21.1.26 file_chooser	157
1.21.1.27 fonts	157
1.21.1.28 forms	157
1.21.1.29 fractals	157
1.21.1.30 fullscreen	157
1.21.1.31 gl_overlay	157
1.21.1.32 glpuzzle	157
1.21.1.33 hello	158
1.21.1.34 help_dialog	158
1.21.1.35 icon	158
1.21.1.36 iconize	158
1.21.1.37 image	158
1.21.1.38 inactive	158
1.21.1.39 input	158
1.21.1.40 input_choice	158
1.21.1.41 keyboard	158
1.21.1.42 label	159
1.21.1.43 line_style	159
1.21.1.44 list_visuals	159
1.21.1.45 mandelbrot	159
1.21.1.46 menubar	159
1.21.1.47 message	159

1.21.1.48 minimum	159
1.21.1.49 native-filechooser	159
1.21.1.50 navigation	159
1.21.1.51 offscreen	159
1.21.1.52 output	160
1.21.1.53 overlay	160
1.21.1.54 pack	160
1.21.1.55 pixmap	160
1.21.1.56 pixmap_browser	160
1.21.1.57 preferences	160
1.21.1.58 radio	160
1.21.1.59 resizebox	160
1.21.1.60 rotated_text	160
1.21.1.61 resize	160
1.21.1.62 scroll	160
1.21.1.63 shape	161
1.21.1.64 subwindow	161
1.21.1.65 sudoku	161
1.21.1.66 symbols	161
1.21.1.67 table	161
1.21.1.68 tabs	161
1.21.1.69 threads	161
1.21.1.70 tile	161
1.21.1.71 tiled_image	162
1.21.1.72 tree	162
1.21.1.73 twowin	162
1.21.1.74 unittests	162
1.21.1.75 utf8	162
1.21.1.76 valuator	162
1.21.1.77 windowfocus	162
1.21.1.78 FLUID	162
1.21.2 Example Applications: Images	162
1.21.2.1 cairo_test	162
1.21.2.2 icon	163
1.21.2.3 unittests	163
1.22 FAQ (Frequently Asked Questions)	164
1.22.1 Where do I start learning FLTK?	164
1.22.2 How do I make a box with text?	164
1.22.3 Can I use FLTK to make closed-source commercial applications?	165
1.22.4 Hitting the 'Escape' key closes windows - how do I prevent this?	165

2.1 The Escape Codes <code>Fl_Terminal</code> Supports	167
2.2 Useful Terminal Escape Code Documentation	168
2.3 <code>Fl_Terminal</code> Design Document	169
3 Development of the FLTK library	175
3.1 The Wayland backend for its developer	175
3.1.1 Introduction to Wayland	175
3.1.2 Building <code>libfltk</code> as a Wayland client	176
3.1.3 The hybrid Wayland/X11 platform	177
3.1.4 Listeners	177
3.1.5 Opening a Wayland connection	178
3.1.6 Wayland windows and surfaces	179
3.1.7 Menu windows and other popups	181
3.1.8 <code>Fl_Wayland_Graphics_Driver</code> and <code>Fl_Cairo_Graphics_Driver</code>	182
3.1.9 Wayland buffers	182
3.1.10 Throttling window redraws	183
3.1.11 Buffer factories	184
3.1.12 Displays and HighDPI support	185
3.1.13 Mouse and trackpad handling	186
3.1.14 Wayland cursors	187
3.1.15 Keyboard support	188
3.1.16 Support of text input methods	189
3.1.17 Interface with <code>libdecor</code>	190
3.1.18 Copy/Paste/Drag-n-Drop	191
3.1.19 EGL as support for OpenGL	191
3.1.20 FLTK-defined, Wayland-specific types	192
3.1.21 Documentation resources	193
3.2 Developer info for bundled libs	194
3.2.1 Introduction	194
3.2.2 Current status	194
3.2.3 How to update the bundled libraries	195
3.2.4 <code>zlib</code> :	195
3.2.5 <code>png</code> :	196
3.2.6 <code>jpeg</code> :	196
3.2.7 <code>nanosvg</code> :	197
3.2.8 <code>libdecor</code> :	198
3.3 Developer Information	198
3.3.1 Non-ASCII Characters	200
3.3.2 Document Structure	200
3.3.3 Creating Links	200
3.3.4 Paragraph Layout	201
3.3.5 Navigation Elements	202

4 Todo List	203
5 Deprecated List	207
6 Topic Index	211
6.1 Topics	211
7 Hierarchical Index	213
7.1 Class Hierarchy	213
8 Class Index	217
8.1 Class List	217
9 File Index	225
9.1 File List	225
10 Topic Documentation	231
10.1 Callback Function Typedefs	231
10.1.1 Detailed Description	232
10.1.2 Typedef Documentation	232
10.1.2.1 FI_Event_Dispatch	232
10.1.2.2 FI_Timeout_Handler	232
10.2 Windows handling functions	232
10.2.1 Detailed Description	233
10.2.2 Function Documentation	233
10.2.2.1 default_atclose()	233
10.2.2.2 first_window() [1/2]	233
10.2.2.3 first_window() [2/2]	233
10.2.2.4 grab() [1/2]	233
10.2.2.5 grab() [2/2]	233
10.2.2.6 modal()	233
10.2.2.7 next_window()	234
10.2.2.8 set_atclose()	234
10.2.3 Variable Documentation	234
10.2.3.1 atclose	234
10.3 Events handling functions	234
10.3.1 Detailed Description	237
10.3.2 Function Documentation	237
10.3.2.1 add_handler() [1/2]	237
10.3.2.2 add_handler() [2/2]	237
10.3.2.3 add_system_handler()	238
10.3.2.4 belowmouse() [1/2]	238
10.3.2.5 belowmouse() [2/2]	238
10.3.2.6 callback_reason()	238

10.3.2.7 compose()	239
10.3.2.8 compose_reset()	239
10.3.2.9 disable_im()	239
10.3.2.10 enable_im()	240
10.3.2.11 event()	240
10.3.2.12 event_button()	240
10.3.2.13 event_button1()	240
10.3.2.14 event_button2()	240
10.3.2.15 event_button3()	240
10.3.2.16 event_button4()	240
10.3.2.17 event_button5()	241
10.3.2.18 event_buttons()	241
10.3.2.19 event_clicks() [1/2]	241
10.3.2.20 event_clicks() [2/2]	241
10.3.2.21 event_clipboard()	241
10.3.2.22 event_clipboard_type()	241
10.3.2.23 event_dispatch()	242
10.3.2.24 event_dx()	242
10.3.2.25 event_dy()	242
10.3.2.26 event_inside() [1/2]	242
10.3.2.27 event_inside() [2/2]	243
10.3.2.28 event_is_click() [1/2]	243
10.3.2.29 event_is_click() [2/2]	243
10.3.2.30 event_key() [1/2]	244
10.3.2.31 event_key() [2/2]	244
10.3.2.32 event_length()	244
10.3.2.33 event_original_key()	244
10.3.2.34 event_state() [1/2]	245
10.3.2.35 event_state() [2/2]	245
10.3.2.36 event_text()	246
10.3.2.37 event_x_root()	246
10.3.2.38 event_y_root()	246
10.3.2.39 focus() [1/2]	246
10.3.2.40 focus() [2/2]	246
10.3.2.41 get_key()	247
10.3.2.42 get_mouse()	247
10.3.2.43 handle()	247
10.3.2.44 handle_()	247
10.3.2.45 last_handler()	248
10.3.2.46 pushed() [1/2]	248
10.3.2.47 pushed() [2/2]	248
10.3.2.48 remove_handler()	248

10.3.2.49	remove_system_handler()	248
10.3.2.50	test_shortcut()	249
10.3.3	Variable Documentation	249
10.3.3.1	fl_callback_reason_names	249
10.3.3.2	fl_eventnames	249
10.3.3.3	fl_fontnames	249
10.4	Selection & Clipboard functions	250
10.4.1	Detailed Description	250
10.4.2	Function Documentation	251
10.4.2.1	add_clipboard_notify()	251
10.4.2.2	clipboard_contains()	251
10.4.2.3	copy()	251
10.4.2.4	dnd()	252
10.4.2.5	paste() [1/2]	252
10.4.2.6	paste() [2/2]	252
10.4.2.7	selection()	253
10.4.2.8	selection_owner() [1/2]	253
10.4.2.9	selection_owner() [2/2]	253
10.4.2.10	selection_to_clipboard() [1/2]	253
10.4.2.11	selection_to_clipboard() [2/2]	254
10.5	Screen functions	254
10.5.1	Detailed Description	255
10.5.2	Function Documentation	255
10.5.2.1	keyboard_screen_scaling()	255
10.5.2.2	screen_count()	256
10.5.2.3	screen_dpi()	256
10.5.2.4	screen_num() [1/2]	256
10.5.2.5	screen_num() [2/2]	256
10.5.2.6	screen_scale() [1/2]	257
10.5.2.7	screen_scale() [2/2]	257
10.5.2.8	screen_scaling_supported()	257
10.5.2.9	screen_work_area() [1/3]	258
10.5.2.10	screen_work_area() [2/3]	258
10.5.2.11	screen_work_area() [3/3]	259
10.5.2.12	screen_xywh() [1/4]	260
10.5.2.13	screen_xywh() [2/4]	260
10.5.2.14	screen_xywh() [3/4]	260
10.5.2.15	screen_xywh() [4/4]	261
10.6	Color & Font functions	261
10.6.1	Detailed Description	263
10.6.2	Function Documentation	263
10.6.2.1	fl_color() [1/3]	263

10.6.2.2 fl_color() [2/3]	263
10.6.2.3 fl_color() [3/3]	264
10.6.2.4 fl_color_average()	264
10.6.2.5 fl_contrast()	264
10.6.2.6 fl_contrast_function()	266
10.6.2.7 fl_contrast_level() [1/2]	266
10.6.2.8 fl_contrast_level() [2/2]	266
10.6.2.9 fl_contrast_mode() [1/2]	267
10.6.2.10 fl_contrast_mode() [2/2]	267
10.6.2.11 fl_font() [1/2]	268
10.6.2.12 fl_font() [2/2]	268
10.6.2.13 fl_height() [1/2]	268
10.6.2.14 fl_height() [2/2]	269
10.6.2.15 fl_latin1_to_local()	269
10.6.2.16 fl_lightness()	269
10.6.2.17 fl_local_to_latin1()	269
10.6.2.18 fl_local_to_mac_roman()	270
10.6.2.19 fl_luminance()	270
10.6.2.20 fl_mac_roman_to_local()	271
10.6.2.21 fl_show_colormap()	271
10.6.2.22 fl_size()	272
10.6.2.23 fl_text_extents() [1/2]	272
10.6.2.24 fl_text_extents() [2/2]	272
10.6.2.25 fl_width()	272
10.6.2.26 free_color()	273
10.6.2.27 get_color() [1/3]	273
10.6.2.28 get_color() [2/3]	273
10.6.2.29 get_color() [3/3]	273
10.6.2.30 get_font()	273
10.6.2.31 get_font_name()	273
10.6.2.32 get_font_sizes()	274
10.6.2.33 set_color() [1/3]	274
10.6.2.34 set_color() [2/3]	274
10.6.2.35 set_color() [3/3]	274
10.6.2.36 set_font()	274
10.6.2.37 set_fonts()	275
10.7 Drawing functions	275
10.7.1 Detailed Description	280
10.7.2 Enumeration Type Documentation	280
10.7.2.1 anonymous enum	280
10.7.3 Function Documentation	281
10.7.3.1 fl_add_symbol()	281

10.7.3.2 fl_antialias()	281
10.7.3.3 fl_arc() [1/2]	281
10.7.3.4 fl_arc() [2/2]	283
10.7.3.5 fl_begin_complex_polygon()	284
10.7.3.6 fl_begin_offscreen()	284
10.7.3.7 fl_begin_points()	284
10.7.3.8 fl_can_do_alpha_blending()	284
10.7.3.9 fl_capture_window()	285
10.7.3.10 fl_circle()	285
10.7.3.11 fl_clip()	285
10.7.3.12 fl_clip_box()	286
10.7.3.13 fl_clip_region() [1/2]	286
10.7.3.14 fl_clip_region() [2/2]	286
10.7.3.15 fl_copy_offscreen()	287
10.7.3.16 fl_create_offscreen()	287
10.7.3.17 fl_cursor()	288
10.7.3.18 fl_curve()	288
10.7.3.19 fl_delete_offscreen()	288
10.7.3.20 fl_draw() [1/5]	288
10.7.3.21 fl_draw() [2/5]	289
10.7.3.22 fl_draw() [3/5]	289
10.7.3.23 fl_draw() [4/5]	290
10.7.3.24 fl_draw() [5/5]	290
10.7.3.25 fl_draw_arrow()	290
10.7.3.26 fl_draw_box()	290
10.7.3.27 fl_draw_box_focus()	291
10.7.3.28 fl_draw_check()	291
10.7.3.29 fl_draw_circle()	292
10.7.3.30 fl_draw_image() [1/2]	292
10.7.3.31 fl_draw_image() [2/2]	293
10.7.3.32 fl_draw_image_mono() [1/2]	293
10.7.3.33 fl_draw_image_mono() [2/2]	293
10.7.3.34 fl_draw_pixmap() [1/2]	294
10.7.3.35 fl_draw_pixmap() [2/2]	294
10.7.3.36 fl_draw_radio()	294
10.7.3.37 fl_draw_symbol()	295
10.7.3.38 fl_expand_text()	295
10.7.3.39 fl_focus_rect()	296
10.7.3.40 fl_frame()	296
10.7.3.41 fl_frame2()	296
10.7.3.42 fl_gap()	297
10.7.3.43 fl_line_style()	297

10.7.3.44 <code>fl_load_matrix()</code>	298
10.7.3.45 <code>fl_measure()</code>	298
10.7.3.46 <code>fl_measure_pixmap()</code> [1/2]	298
10.7.3.47 <code>fl_measure_pixmap()</code> [2/2]	299
10.7.3.48 <code>fl_mult_matrix()</code>	299
10.7.3.49 <code>fl_not_clipped()</code>	299
10.7.3.50 <code>fl_old_shortcut()</code>	300
10.7.3.51 <code>fl_overlay_clear()</code>	300
10.7.3.52 <code>fl_overlay_rect()</code>	301
10.7.3.53 <code>fl_override_scale()</code>	301
10.7.3.54 <code>fl_pie()</code>	302
10.7.3.55 <code>fl_polygon()</code>	302
10.7.3.56 <code>fl_pop_clip()</code>	302
10.7.3.57 <code>fl_push_clip()</code>	302
10.7.3.58 <code>fl_push_matrix()</code>	303
10.7.3.59 <code>fl_read_image()</code>	303
10.7.3.60 <code>fl_rect()</code> [1/3]	303
10.7.3.61 <code>fl_rect()</code> [2/3]	304
10.7.3.62 <code>fl_rect()</code> [3/3]	304
10.7.3.63 <code>fl_rectf()</code> [1/4]	304
10.7.3.64 <code>fl_rectf()</code> [2/4]	304
10.7.3.65 <code>fl_rectf()</code> [3/4]	304
10.7.3.66 <code>fl_rectf()</code> [4/4]	305
10.7.3.67 <code>fl_rescale_offscreen()</code>	305
10.7.3.68 <code>fl_reset_spot()</code>	305
10.7.3.69 <code>fl_restore_scale()</code>	305
10.7.3.70 <code>fl_rotate()</code>	305
10.7.3.71 <code>fl_rounded_rect()</code>	306
10.7.3.72 <code>fl_rounded_rectf()</code>	306
10.7.3.73 <code>fl_scale()</code> [1/2]	306
10.7.3.74 <code>fl_scale()</code> [2/2]	306
10.7.3.75 <code>fl_scroll()</code>	306
10.7.3.76 <code>fl_set_spot()</code>	307
10.7.3.77 <code>fl_set_status()</code>	307
10.7.3.78 <code>fl_shortcut_label()</code> [1/2]	308
10.7.3.79 <code>fl_shortcut_label()</code> [2/2]	308
10.7.3.80 <code>fl_transform_dx()</code>	309
10.7.3.81 <code>fl_transform_dy()</code>	309
10.7.3.82 <code>fl_transform_x()</code>	309
10.7.3.83 <code>fl_transform_y()</code>	309
10.7.3.84 <code>fl_transformed_vertex()</code>	310
10.7.3.85 <code>fl_translate()</code>	311

10.7.3.86 fl_vertex()	311
10.8 Multithreading support functions	311
10.8.1 Detailed Description	311
10.8.2 Function Documentation	312
10.8.2.1 awake() [1/2]	312
10.8.2.2 awake() [2/2]	312
10.8.2.3 lock()	312
10.8.2.4 thread_message()	312
10.8.2.5 unlock()	313
10.9 Safe widget deletion support functions	313
10.9.1 Detailed Description	313
10.9.2 Function Documentation	314
10.9.2.1 clear_widget_pointer()	314
10.9.2.2 delete_widget()	314
10.9.2.3 do_widget_deletion()	314
10.9.2.4 release_widget_pointer()	315
10.9.2.5 watch_widget_pointer()	315
10.10 Cairo Support Functions and Classes	315
10.10.1 Detailed Description	316
10.10.2 Function Documentation	316
10.10.2.1 cairo_autolink_context() [1/2]	316
10.10.2.2 cairo_autolink_context() [2/2]	316
10.10.2.3 cairo_cc()	317
10.10.2.4 cairo_flush()	317
10.10.2.5 cairo_make_current()	317
10.11 Unicode and UTF-8 functions	318
10.11.1 Detailed Description	320
10.11.2 Macro Definition Documentation	320
10.11.2.1 ERRORS_TO_CP1252	320
10.11.2.2 ERRORS_TO_ISO8859_1	320
10.11.2.3 STRICT_RFC3629	320
10.11.3 Function Documentation	320
10.11.3.1 fl_access()	320
10.11.3.2 fl_chdir()	321
10.11.3.3 fl_chmod()	321
10.11.3.4 fl_close_fd()	322
10.11.3.5 fl_fopen()	322
10.11.3.6 fl_getcwd()	322
10.11.3.7 fl_getenv()	323
10.11.3.8 fl_make_path()	323
10.11.3.9 fl_make_path_for_file()	323
10.11.3.10 fl_mkdir()	323

10.11.3.11 fl_nonspacing()	324
10.11.3.12 fl_open()	324
10.11.3.13 fl_open_ext()	324
10.11.3.14 fl_putenv()	325
10.11.3.15 fl_rename()	325
10.11.3.16 fl_rmdir()	326
10.11.3.17 fl_stat()	326
10.11.3.18 fl_system()	326
10.11.3.19 fl_ucs_to_Utf16()	327
10.11.3.20 fl_unlink()	327
10.11.3.21 fl_utf8back()	327
10.11.3.22 fl_utf8bytes()	327
10.11.3.23 fl_utf8decode()	328
10.11.3.24 fl_utf8encode()	328
10.11.3.25 fl_utf8from_mb()	328
10.11.3.26 fl_utf8froma()	329
10.11.3.27 fl_utf8fromwc()	329
10.11.3.28 fl_utf8fwd()	329
10.11.3.29 fl_utf8len()	330
10.11.3.30 fl_utf8len1()	330
10.11.3.31 fl_utf8locale()	330
10.11.3.32 fl_utf8strlen()	330
10.11.3.33 fl_utf8test()	330
10.11.3.34 fl_utf8to_mb()	331
10.11.3.35 fl_utf8toa()	331
10.11.3.36 fl_utf8toUtf16()	331
10.11.3.37 fl_utf8towc()	332
10.11.3.38 fl_utf_nb_char()	332
10.11.3.39 fl_utf_strcasecmp()	332
10.11.3.40 fl_utf_strncasecmp()	333
10.11.3.41 fl_utf_tolower()	333
10.11.3.42 fl_utf_toupper()	333
10.11.3.43 fl_wcwidth()	333
10.11.3.44 fl_wcwidth_()	334
10.12 String handling functions	334
10.12.1 Detailed Description	334
10.12.2 Function Documentation	334
10.12.2.1 fl_strdup()	334
10.13 Mac OS X-specific symbols	335
10.13.1 Detailed Description	335
10.13.2 Function Documentation	335
10.13.2.1 fl_mac_set_about()	335

10.13.2.2 fl_open_callback()	335
10.13.3 Variable Documentation	336
10.13.3.1 fl_mac_os_version	336
10.14 Common Dialog Classes and Functions	336
10.14.1 Detailed Description	337
10.14.2 Function Documentation	338
10.14.2.1 fl_alert()	338
10.14.2.2 fl_ask()	338
10.14.2.3 fl_beep()	338
10.14.2.4 fl_choice()	339
10.14.2.5 fl_choice_n()	340
10.14.2.6 fl_color_chooser() [1/2]	341
10.14.2.7 fl_color_chooser() [2/2]	341
10.14.2.8 fl_dir_chooser()	342
10.14.2.9 fl_file_chooser()	343
10.14.2.10 fl_file_chooser_callback()	343
10.14.2.11 fl_file_chooser_ok_label()	344
10.14.2.12 fl_input() [1/2]	344
10.14.2.13 fl_input() [2/2]	344
10.14.2.14 fl_message()	345
10.14.2.15 fl_message_hotspot() [1/2]	345
10.14.2.16 fl_message_hotspot() [2/2]	345
10.14.2.17 fl_message_icon()	346
10.14.2.18 fl_message_icon_label()	346
10.14.2.19 fl_message_position() [1/3]	346
10.14.2.20 fl_message_position() [2/3]	347
10.14.2.21 fl_message_position() [3/3]	347
10.14.2.22 fl_message_title()	348
10.14.2.23 fl_message_title_default()	348
10.14.2.24 fl_password() [1/2]	348
10.14.2.25 fl_password() [2/2]	349
10.14.3 Variable Documentation	349
10.14.3.1 error	349
10.14.3.2 fatal	350
10.14.3.3 warning	350
10.15 File names and URI utility functions	350
10.15.1 Detailed Description	351
10.15.2 Typedef Documentation	351
10.15.2.1 FI_File_Sort_F	351
10.15.3 Function Documentation	352
10.15.3.1 fl_decode_uri()	352
10.15.3.2 fl_filename_absolute() [1/2]	352

10.15.3.3 fl_filename_absolute() [2/2]	352
10.15.3.4 fl_filename_absolute_str() [1/2]	353
10.15.3.5 fl_filename_absolute_str() [2/2]	353
10.15.3.6 fl_filename_expand()	353
10.15.3.7 fl_filename_expand_str()	354
10.15.3.8 fl_filename_ext()	354
10.15.3.9 fl_filename_ext_str()	354
10.15.3.10 fl_filename_free_list()	355
10.15.3.11 fl_filename_isdir()	355
10.15.3.12 fl_filename_list()	355
10.15.3.13 fl_filename_match()	356
10.15.3.14 fl_filename_name()	357
10.15.3.15 fl_filename_name_str()	357
10.15.3.16 fl_filename_path_str()	357
10.15.3.17 fl_filename_relative() [1/2]	358
10.15.3.18 fl_filename_relative() [2/2]	358
10.15.3.19 fl_filename_relative_str() [1/2]	359
10.15.3.20 fl_filename_relative_str() [2/2]	359
10.15.3.21 fl_filename_setext()	359
10.15.3.22 fl_filename_setext_str()	360
10.15.3.23 fl_getcwd_str()	360
10.15.3.24 fl_open_uri()	360
11 Class Documentation	363
11.1 FI_Grid::Cell Class Reference	363
11.1.1 Constructor & Destructor Documentation	363
11.1.1.1 ~Cell()	363
11.1.2 Member Function Documentation	364
11.1.2.1 next()	364
11.2 FI_Terminal::CharStyle Class Reference	364
11.3 FI_GIF_Image::GIF_FRAME::CPAL Struct Reference	365
11.4 FI_Terminal::Cursor Class Reference	365
11.5 FI_Preferences::Entry Struct Reference	365
11.6 FI_Terminal::EscapeSeq Class Reference	366
11.7 FI Class Reference	366
11.7.1 Detailed Description	376
11.7.2 Member Enumeration Documentation	376
11.7.2.1 FI_Option	376
11.7.3 Member Function Documentation	377
11.7.3.1 abi_check()	377
11.7.3.2 abi_version()	377
11.7.3.3 add_check()	377

11.7.3.4 add_fd() [1/2]	378
11.7.3.5 add_fd() [2/2]	378
11.7.3.6 add_idle()	378
11.7.3.7 add_timeout()	378
11.7.3.8 api_version()	379
11.7.3.9 arg()	379
11.7.3.10 args() [1/2]	380
11.7.3.11 args() [2/2]	380
11.7.3.12 args_to_utf8()	381
11.7.3.13 background()	382
11.7.3.14 background2()	382
11.7.3.15 box_border_radius_max() [1/2]	382
11.7.3.16 box_border_radius_max() [2/2]	382
11.7.3.17 box_color()	383
11.7.3.18 box_dh()	383
11.7.3.19 box_dw()	383
11.7.3.20 box_dx()	383
11.7.3.21 box_dy()	383
11.7.3.22 box_shadow_width() [1/2]	384
11.7.3.23 box_shadow_width() [2/2]	384
11.7.3.24 check()	384
11.7.3.25 display()	384
11.7.3.26 dnd_text_ops() [1/2]	384
11.7.3.27 dnd_text_ops() [2/2]	384
11.7.3.28 draw_box_active()	384
11.7.3.29 draw_GL_text_with_textures() [1/2]	385
11.7.3.30 draw_GL_text_with_textures() [2/2]	385
11.7.3.31 flush()	385
11.7.3.32 get_system_colors()	385
11.7.3.33 gl_visual()	386
11.7.3.34 has_timeout()	386
11.7.3.35 hide_all_windows()	386
11.7.3.36 is_scheme()	387
11.7.3.37 menu_linespacing() [1/2]	387
11.7.3.38 menu_linespacing() [2/2]	387
11.7.3.39 now()	388
11.7.3.40 option() [1/2]	388
11.7.3.41 option() [2/2]	389
11.7.3.42 own_colormap()	389
11.7.3.43 program_should_quit() [1/2]	389
11.7.3.44 program_should_quit() [2/2]	390
11.7.3.45 readqueue()	390

11.7.3.46	ready()	390
11.7.3.47	release()	390
11.7.3.48	reload_scheme()	390
11.7.3.49	remove_check()	391
11.7.3.50	remove_next_timeout()	391
11.7.3.51	remove_timeout()	391
11.7.3.52	repeat_timeout()	392
11.7.3.53	run()	392
11.7.3.54	scheme()	393
11.7.3.55	scrollbar_size() [1/2]	393
11.7.3.56	scrollbar_size() [2/2]	393
11.7.3.57	seconds_between()	394
11.7.3.58	seconds_since()	394
11.7.3.59	set_box_color()	394
11.7.3.60	set_boxtype()	395
11.7.3.61	set_idle()	395
11.7.3.62	ticks_between()	395
11.7.3.63	ticks_since()	395
11.7.3.64	use_high_res_GL() [1/2]	396
11.7.3.65	use_high_res_GL() [2/2]	396
11.7.3.66	version()	396
11.7.3.67	visible_focus() [1/2]	396
11.7.3.68	visible_focus() [2/2]	396
11.7.3.69	visual()	397
11.7.3.70	wait() [1/2]	397
11.7.3.71	wait() [2/2]	397
11.7.4	Member Data Documentation	397
11.7.4.1	help	397
11.7.4.2	idle	397
11.8	FI_Adjuster Class Reference	398
11.8.1	Detailed Description	406
11.8.2	Constructor & Destructor Documentation	407
11.8.2.1	FI_Adjuster()	407
11.8.3	Member Function Documentation	407
11.8.3.1	draw()	407
11.8.3.2	handle()	407
11.8.3.3	soft() [1/2]	408
11.8.3.4	soft() [2/2]	408
11.8.3.5	value_damage()	408
11.9	FI_Anim_GIF_Image Class Reference	408
11.9.1	Detailed Description	413
11.9.2	Member Enumeration Documentation	413

11.9.2.1 Flags	413
11.9.3 Constructor & Destructor Documentation	414
11.9.3.1 FI_Anim_GIF_Image() [1/2]	414
11.9.3.2 FI_Anim_GIF_Image() [2/2]	414
11.9.3.3 ~FI_Anim_GIF_Image()	415
11.9.4 Member Function Documentation	415
11.9.4.1 canvas() [1/2]	415
11.9.4.2 canvas() [2/2]	415
11.9.4.3 canvas_h()	415
11.9.4.4 canvas_w()	416
11.9.4.5 color_average()	416
11.9.4.6 copy()	416
11.9.4.7 delay() [1/2]	416
11.9.4.8 delay() [2/2]	416
11.9.4.9 desaturate()	417
11.9.4.10 draw()	417
11.9.4.11 frame() [1/2]	417
11.9.4.12 frame() [2/2]	417
11.9.4.13 frame_count()	418
11.9.4.14 frame_h()	418
11.9.4.15 frame_uncache() [1/2]	418
11.9.4.16 frame_uncache() [2/2]	418
11.9.4.17 frame_w()	419
11.9.4.18 frame_x()	419
11.9.4.19 frame_y()	419
11.9.4.20 frames()	419
11.9.4.21 image() [1/2]	420
11.9.4.22 image() [2/2]	420
11.9.4.23 is_animated()	420
11.9.4.24 load()	420
11.9.4.25 name()	421
11.9.4.26 next()	421
11.9.4.27 on_extension_data()	421
11.9.4.28 on_frame_data()	421
11.9.4.29 playing()	421
11.9.4.30 resize() [1/2]	421
11.9.4.31 resize() [2/2]	421
11.9.4.32 speed() [1/2]	422
11.9.4.33 speed() [2/2]	422
11.9.4.34 start()	422
11.9.4.35 stop()	422
11.9.4.36 uncache()	422

11.9.4.37 valid()	422
11.9.5 Member Data Documentation	423
11.9.5.1 loop	423
11.9.5.2 min_delay	423
11.10 FI_Bitmap Class Reference	423
11.10.1 Detailed Description	426
11.10.2 Constructor & Destructor Documentation	426
11.10.2.1 FI_Bitmap() [1/4]	426
11.10.2.2 FI_Bitmap() [2/4]	426
11.10.2.3 FI_Bitmap() [3/4]	426
11.10.2.4 FI_Bitmap() [4/4]	426
11.10.3 Member Function Documentation	427
11.10.3.1 copy()	427
11.10.3.2 draw()	428
11.10.3.3 label() [1/2]	428
11.10.3.4 label() [2/2]	428
11.10.3.5 uncache()	428
11.11 FI_BMP_Image Class Reference	428
11.11.1 Detailed Description	431
11.11.2 Constructor & Destructor Documentation	432
11.11.2.1 FI_BMP_Image() [1/2]	432
11.11.2.2 FI_BMP_Image() [2/2]	432
11.12 FI_Box Class Reference	433
11.12.1 Detailed Description	439
11.12.2 Constructor & Destructor Documentation	439
11.12.2.1 FI_Box() [1/2]	439
11.12.2.2 FI_Box() [2/2]	440
11.12.3 Member Function Documentation	440
11.12.3.1 draw()	440
11.12.3.2 handle()	440
11.13 FI_Browser Class Reference	441
11.13.1 Detailed Description	454
11.13.2 Constructor & Destructor Documentation	455
11.13.2.1 FI_Browser()	455
11.13.3 Member Function Documentation	455
11.13.3.1 _remove()	455
11.13.3.2 add()	456
11.13.3.3 bottomline()	456
11.13.3.4 clear()	456
11.13.3.5 column_char() [1/2]	456
11.13.3.6 column_char() [2/2]	456
11.13.3.7 column_widths() [1/2]	457

11.13.3.8 column_widths() [2/2]	457
11.13.3.9 data() [1/2]	457
11.13.3.10 data() [2/2]	457
11.13.3.11 display()	458
11.13.3.12 displayed()	458
11.13.3.13 find_line()	458
11.13.3.14 format_char() [1/2]	459
11.13.3.15 format_char() [2/2]	459
11.13.3.16 full_height()	459
11.13.3.17 hide() [1/2]	460
11.13.3.18 hide() [2/2]	460
11.13.3.19 icon() [1/2]	460
11.13.3.20 icon() [2/2]	460
11.13.3.21 incr_height()	461
11.13.3.22 insert() [1/2]	461
11.13.3.23 insert() [2/2]	461
11.13.3.24 item_at()	461
11.13.3.25 item_draw()	462
11.13.3.26 item_first()	462
11.13.3.27 item_height()	462
11.13.3.28 item_last()	463
11.13.3.29 item_next()	463
11.13.3.30 item_prev()	463
11.13.3.31 item_select()	464
11.13.3.32 item_selected()	464
11.13.3.33 item_swap()	464
11.13.3.34 item_text()	464
11.13.3.35 item_width()	465
11.13.3.36 lineno()	465
11.13.3.37 lineposition()	465
11.13.3.38 load()	466
11.13.3.39 make_visible()	466
11.13.3.40 middleline()	466
11.13.3.41 move()	467
11.13.3.42 remove()	467
11.13.3.43 remove_icon()	467
11.13.3.44 select()	467
11.13.3.45 selected()	468
11.13.3.46 show() [1/2]	468
11.13.3.47 show() [2/2]	468
11.13.3.48 size()	468
11.13.3.49 swap() [1/2]	468

11.13.3.50 swap() [2/2]	469
11.13.3.51 text() [1/2]	469
11.13.3.52 text() [2/2]	469
11.13.3.53 textsize()	470
11.13.3.54 topline() [1/2]	470
11.13.3.55 topline() [2/2]	470
11.13.3.56 value() [1/2]	470
11.13.3.57 value() [2/2]	470
11.13.3.58 visible()	471
11.14 FI_Browser_ Class Reference	471
11.14.1 Detailed Description	482
11.14.2 Member Enumeration Documentation	482
11.14.2.1 anonymous enum	482
11.14.3 Constructor & Destructor Documentation	483
11.14.3.1 FI_Browser_()	483
11.14.4 Member Function Documentation	483
11.14.4.1 bbox()	483
11.14.4.2 deleting()	483
11.14.4.3 deselect()	484
11.14.4.4 display()	484
11.14.4.5 displayed()	484
11.14.4.6 draw()	484
11.14.4.7 find_item()	484
11.14.4.8 full_height()	485
11.14.4.9 full_width()	485
11.14.4.10 handle()	485
11.14.4.11 has_scrollbar()	485
11.14.4.12 hposition() [1/2]	486
11.14.4.13 hposition() [2/2]	486
11.14.4.14 incr_height()	486
11.14.4.15 inserting()	486
11.14.4.16 item_at()	487
11.14.4.17 item_draw()	487
11.14.4.18 item_first()	487
11.14.4.19 item_height()	487
11.14.4.20 item_last()	488
11.14.4.21 item_next()	488
11.14.4.22 item_prev()	488
11.14.4.23 item_quick_height()	488
11.14.4.24 item_select()	488
11.14.4.25 item_selected()	489
11.14.4.26 item_swap()	489

11.14.4.27	item_text()	489
11.14.4.28	item_width()	489
11.14.4.29	leftedge()	490
11.14.4.30	linespacing() [1/2]	490
11.14.4.31	linespacing() [2/2]	490
11.14.4.32	new_list()	490
11.14.4.33	position() [1/2]	490
11.14.4.34	position() [2/2]	491
11.14.4.35	redraw_line()	491
11.14.4.36	redraw_lines()	491
11.14.4.37	replacing()	491
11.14.4.38	resize()	491
11.14.4.39	scrollbar_left()	492
11.14.4.40	scrollbar_right()	492
11.14.4.41	scrollbar_size() [1/2]	492
11.14.4.42	scrollbar_size() [2/2]	492
11.14.4.43	scrollbar_width() [1/2]	492
11.14.4.44	scrollbar_width() [2/2]	493
11.14.4.45	select()	493
11.14.4.46	select_only()	493
11.14.4.47	selection()	493
11.14.4.48	sort()	493
11.14.4.49	swapping()	494
11.14.4.50	textfont()	494
11.14.4.51	vposition() [1/2]	494
11.14.4.52	vposition() [2/2]	494
11.14.5	Member Data Documentation	495
11.14.5.1	hscrollbar	495
11.14.5.2	scrollbar	495
11.15	FI_Button Class Reference	495
11.15.1	Detailed Description	502
11.15.2	Constructor & Destructor Documentation	503
11.15.2.1	FI_Button()	503
11.15.3	Member Function Documentation	503
11.15.3.1	clear()	503
11.15.3.2	compact() [1/2]	503
11.15.3.3	compact() [2/2]	504
11.15.3.4	down_box() [1/2]	504
11.15.3.5	down_box() [2/2]	505
11.15.3.6	draw()	505
11.15.3.7	handle()	505
11.15.3.8	set()	506

11.15.3.9 shortcut() [1/2]	506
11.15.3.10 shortcut() [2/2]	506
11.15.3.11 value()	506
11.16 FI_Cairo_State Class Reference	507
11.16.1 Detailed Description	507
11.16.2 Member Function Documentation	507
11.16.2.1 cc()	507
11.17 FI_Cairo_Window Class Reference	507
11.17.1 Detailed Description	520
11.17.2 Member Function Documentation	521
11.17.2.1 draw()	521
11.17.2.2 set_draw_cb()	521
11.18 FI_Callback_User_Data Class Reference	521
11.18.1 Detailed Description	521
11.19 FI_Chart Class Reference	522
11.19.1 Detailed Description	529
11.19.2 Constructor & Destructor Documentation	530
11.19.2.1 FI_Chart()	530
11.19.3 Member Function Documentation	530
11.19.3.1 add()	530
11.19.3.2 autosize() [1/2]	530
11.19.3.3 autosize() [2/2]	531
11.19.3.4 bounds() [1/2]	532
11.19.3.5 bounds() [2/2]	532
11.19.3.6 draw()	532
11.19.3.7 draw_barchart()	532
11.19.3.8 draw_horbarchart()	533
11.19.3.9 draw_linechart()	533
11.19.3.10 draw_piechart()	534
11.19.3.11 insert()	534
11.19.3.12 maxsize()	535
11.19.3.13 replace()	535
11.19.3.14 size()	535
11.20 FL_CHART_ENTRY Struct Reference	535
11.20.1 Detailed Description	536
11.21 FI_Check_Browser Class Reference	536
11.21.1 Detailed Description	547
11.21.2 Member Function Documentation	548
11.21.2.1 add() [1/2]	548
11.21.2.2 add() [2/2]	548
11.21.2.3 handle()	548
11.21.2.4 item_at()	548

11.21.2.5 item_draw()	548
11.21.2.6 item_first()	549
11.21.2.7 item_height()	549
11.21.2.8 item_next()	549
11.21.2.9 item_prev()	549
11.21.2.10 item_select()	549
11.21.2.11 item_selected()	550
11.21.2.12 item_swap()	550
11.21.2.13 item_text()	550
11.21.2.14 item_width()	550
11.21.2.15 nitems()	551
11.21.2.16 remove()	551
11.22 FI_Check_Button Class Reference	551
11.22.1 Detailed Description	559
11.22.2 Constructor & Destructor Documentation	559
11.22.2.1 FI_Check_Button()	559
11.23 FI_Choice Class Reference	559
11.23.1 Detailed Description	568
11.23.2 Constructor & Destructor Documentation	570
11.23.2.1 FI_Choice()	570
11.23.3 Member Function Documentation	570
11.23.3.1 draw()	570
11.23.3.2 handle()	570
11.23.3.3 value() [1/3]	571
11.23.3.4 value() [2/3]	571
11.23.3.5 value() [3/3]	571
11.24 FI_Clock Class Reference	571
11.24.1 Detailed Description	579
11.24.2 Constructor & Destructor Documentation	579
11.24.2.1 FI_Clock() [1/2]	579
11.24.2.2 FI_Clock() [2/2]	580
11.24.3 Member Function Documentation	580
11.24.3.1 handle()	580
11.25 FI_Clock_Output Class Reference	581
11.25.1 Detailed Description	588
11.25.2 Constructor & Destructor Documentation	589
11.25.2.1 FI_Clock_Output()	589
11.25.3 Member Function Documentation	589
11.25.3.1 draw() [1/2]	589
11.25.3.2 draw() [2/2]	589
11.25.3.3 hour()	589
11.25.3.4 minute()	589

11.25.3.5 second()	590
11.25.3.6 shadow() [1/2]	590
11.25.3.7 shadow() [2/2]	590
11.25.3.8 value() [1/3]	590
11.25.3.9 value() [2/3]	590
11.25.3.10 value() [3/3]	591
11.26 FI_Color_Chooser Class Reference	591
11.26.1 Detailed Description	600
11.26.2 Constructor & Destructor Documentation	601
11.26.2.1 FI_Color_Chooser()	601
11.26.3 Member Function Documentation	601
11.26.3.1 b()	601
11.26.3.2 g()	601
11.26.3.3 handle()	601
11.26.3.4 hsv()	602
11.26.3.5 hsv2rgb()	602
11.26.3.6 hue()	602
11.26.3.7 mode() [1/2]	603
11.26.3.8 mode() [2/2]	603
11.26.3.9 r()	603
11.26.3.10 rgb()	603
11.26.3.11 rgb2hsv()	603
11.26.3.12 saturation()	604
11.26.3.13 value()	604
11.27 FI_Copy_Surface Class Reference	604
11.27.1 Detailed Description	606
11.27.2 Constructor & Destructor Documentation	606
11.27.2.1 FI_Copy_Surface()	606
11.27.3 Member Function Documentation	606
11.27.3.1 is_current()	606
11.27.3.2 origin() [1/2]	606
11.27.3.3 origin() [2/2]	607
11.27.3.4 printable_rect()	607
11.27.3.5 set_current()	607
11.27.3.6 translate()	608
11.27.3.7 untranslate()	608
11.28 FI_Counter Class Reference	608
11.28.1 Detailed Description	616
11.28.2 Constructor & Destructor Documentation	617
11.28.2.1 FI_Counter()	617
11.28.3 Member Function Documentation	617
11.28.3.1 arrow_widths()	617

11.28.3.2 draw()	617
11.28.3.3 handle()	617
11.28.3.4 lstep()	618
11.28.3.5 step() [1/2]	618
11.28.3.6 step() [2/2]	618
11.29 FI_Device_Plugin Class Reference	619
11.29.1 Detailed Description	619
11.29.2 Member Function Documentation	620
11.29.2.1 rectangle_capture()	620
11.30 FI_Dial Class Reference	620
11.30.1 Detailed Description	628
11.30.2 Constructor & Destructor Documentation	629
11.30.2.1 FI_Dial()	629
11.30.3 Member Function Documentation	629
11.30.3.1 angle1()	629
11.30.3.2 draw() [1/2]	629
11.30.3.3 draw() [2/2]	629
11.30.3.4 handle() [1/2]	629
11.30.3.5 handle() [2/2]	630
11.31 FI_Display_Device Class Reference	630
11.31.1 Detailed Description	631
11.32 FI_Double_Window Class Reference	631
11.32.1 Detailed Description	643
11.32.2 Constructor & Destructor Documentation	643
11.32.2.1 ~FI_Double_Window()	643
11.32.3 Member Function Documentation	643
11.32.3.1 as_double_window()	643
11.32.3.2 flush()	643
11.32.3.3 hide()	644
11.32.3.4 resize()	644
11.32.3.5 show()	644
11.33 FI_End Class Reference	645
11.33.1 Detailed Description	645
11.34 FI_EPS_File_Surface Class Reference	645
11.34.1 Detailed Description	647
11.34.2 Constructor & Destructor Documentation	647
11.34.2.1 FI_EPS_File_Surface()	647
11.34.2.2 ~FI_EPS_File_Surface()	647
11.34.3 Member Function Documentation	648
11.34.3.1 close()	648
11.34.3.2 origin() [1/2]	648
11.34.3.3 origin() [2/2]	648

11.34.3.4 printable_rect()	648
11.34.3.5 translate()	649
11.34.3.6 untranslate()	649
11.35 FI_File_Browser Class Reference	649
11.35.1 Detailed Description	662
11.35.2 Constructor & Destructor Documentation	663
11.35.2.1 FI_File_Browser()	663
11.35.3 Member Function Documentation	663
11.35.3.1 errmsg() [1/2]	663
11.35.3.2 errmsg() [2/2]	663
11.35.3.3 filetype() [1/2]	663
11.35.3.4 filetype() [2/2]	663
11.35.3.5 filter() [1/2]	663
11.35.3.6 filter() [2/2]	663
11.35.3.7 iconsize() [1/2]	663
11.35.3.8 iconsize() [2/2]	664
11.35.3.9 load()	664
11.36 FI_File_Chooser Class Reference	664
11.36.1 Detailed Description	667
11.36.2 Member Enumeration Documentation	669
11.36.2.1 Type	669
11.36.3 Constructor & Destructor Documentation	669
11.36.3.1 FI_File_Chooser()	669
11.36.4 Member Function Documentation	669
11.36.4.1 add_extra()	669
11.36.4.2 filter()	670
11.36.4.3 iconsize() [1/2]	670
11.36.4.4 iconsize() [2/2]	670
11.36.4.5 preview()	670
11.36.4.6 shown()	670
11.36.4.7 value() [1/2]	671
11.36.4.8 value() [2/2]	671
11.36.5 Member Data Documentation	671
11.36.5.1 showHiddenButton	671
11.37 FI_File_Icon Class Reference	671
11.37.1 Detailed Description	673
11.37.2 Constructor & Destructor Documentation	673
11.37.2.1 FI_File_Icon()	673
11.37.3 Member Function Documentation	673
11.37.3.1 add()	673
11.37.3.2 add_color()	673
11.37.3.3 add_vertex() [1/2]	673

11.37.3.4 add_vertex() [2/2]	674
11.37.3.5 draw()	674
11.37.3.6 find()	674
11.37.3.7 label()	674
11.37.3.8 labeltype()	675
11.37.3.9 load()	675
11.37.3.10 load_fti()	675
11.37.3.11 load_image()	675
11.37.3.12 load_system_icons()	676
11.37.3.13 next()	676
11.37.3.14 type()	676
11.38 FI_File_Input Class Reference	676
11.38.1 Detailed Description	686
11.38.2 Constructor & Destructor Documentation	687
11.38.2.1 FI_File_Input()	687
11.38.3 Member Function Documentation	687
11.38.3.1 down_box()	687
11.38.3.2 draw()	687
11.38.3.3 errorcolor() [1/2]	687
11.38.3.4 errorcolor() [2/2]	687
11.38.3.5 handle()	688
11.38.3.6 value() [1/2]	688
11.38.3.7 value() [2/2]	688
11.39 FI_Fill_Dial Class Reference	688
11.39.1 Detailed Description	697
11.40 FI_Fill_Slider Class Reference	697
11.40.1 Detailed Description	705
11.41 FI_Flex Class Reference	705
11.41.1 Detailed Description	715
11.41.2 Member Enumeration Documentation	716
11.41.2.1 anonymous enum	716
11.41.3 Constructor & Destructor Documentation	716
11.41.3.1 FI_Flex() [1/4]	716
11.41.3.2 FI_Flex() [2/4]	716
11.41.3.3 FI_Flex() [3/4]	717
11.41.3.4 FI_Flex() [4/4]	717
11.41.4 Member Function Documentation	718
11.41.4.1 alloc_size()	718
11.41.4.2 draw()	718
11.41.4.3 end()	718
11.41.4.4 fixed() [1/3]	718
11.41.4.5 fixed() [2/3]	719

11.41.4.6 fixed() [3/3]	719
11.41.4.7 gap() [1/2]	719
11.41.4.8 gap() [2/2]	719
11.41.4.9 horizontal()	720
11.41.4.10 layout()	720
11.41.4.11 margin() [1/4]	720
11.41.4.12 margin() [2/4]	720
11.41.4.13 margin() [3/4]	721
11.41.4.14 margin() [4/4]	721
11.41.4.15 need_layout() [1/2]	722
11.41.4.16 need_layout() [2/2]	722
11.41.4.17 on_remove()	722
11.41.4.18 resize()	722
11.41.4.19 spacing() [1/2]	722
11.41.4.20 spacing() [2/2]	723
11.42 FI_Float_Input Class Reference	723
11.42.1 Detailed Description	733
11.42.2 Constructor & Destructor Documentation	733
11.42.2.1 FI_Float_Input()	733
11.43 FI_FormsBitmap Class Reference	733
11.43.1 Detailed Description	740
11.43.2 Member Function Documentation	740
11.43.2.1 draw()	740
11.43.2.2 set()	740
11.44 FI_FormsPixmap Class Reference	740
11.44.1 Detailed Description	747
11.44.2 Constructor & Destructor Documentation	747
11.44.2.1 FI_FormsPixmap()	747
11.44.3 Member Function Documentation	747
11.44.3.1 draw()	747
11.44.3.2 Pixmap()	748
11.44.3.3 set()	748
11.45 FI_FormsText Class Reference	748
11.45.1 Member Function Documentation	755
11.45.1.1 draw()	755
11.46 FI_Free Class Reference	755
11.46.1 Detailed Description	762
11.46.2 Constructor & Destructor Documentation	762
11.46.2.1 FI_Free()	762
11.46.3 Member Function Documentation	763
11.46.3.1 draw()	763
11.46.3.2 handle()	763

11.47 FI_GIF_Image Class Reference	764
11.47.1 Detailed Description	767
11.47.2 Constructor & Destructor Documentation	767
11.47.2.1 FI_GIF_Image() [1/3]	767
11.47.2.2 FI_GIF_Image() [2/3]	767
11.47.2.3 FI_GIF_Image() [3/3]	768
11.47.3 Member Data Documentation	768
11.47.3.1 animate	768
11.48 FI_GI_Choice Class Reference	769
11.49 FI_GI_Window Class Reference	769
11.49.1 Detailed Description	783
11.49.2 Constructor & Destructor Documentation	783
11.49.2.1 FI_GI_Window() [1/2]	783
11.49.2.2 FI_GI_Window() [2/2]	783
11.49.3 Member Function Documentation	783
11.49.3.1 as_gl_window() [1/2]	783
11.49.3.2 as_gl_window() [2/2]	783
11.49.3.3 can_do()	784
11.49.3.4 can_do_overlay()	784
11.49.3.5 context() [1/2]	784
11.49.3.6 context() [2/2]	784
11.49.3.7 context_valid()	784
11.49.3.8 draw()	785
11.49.3.9 draw_begin()	785
11.49.3.10 draw_end()	785
11.49.3.11 flush()	786
11.49.3.12 handle()	786
11.49.3.13 hide()	786
11.49.3.14 make_current()	786
11.49.3.15 make_overlay_current()	786
11.49.3.16 mode() [1/3]	786
11.49.3.17 mode() [2/3]	786
11.49.3.18 mode() [3/3]	787
11.49.3.19 ortho()	787
11.49.3.20 pixel_h()	787
11.49.3.21 pixel_w()	788
11.49.3.22 pixels_per_unit()	788
11.49.3.23 redraw_overlay()	788
11.49.3.24 resize()	788
11.49.3.25 show()	789
11.49.3.26 swap_buffers()	789
11.49.3.27 swap_interval() [1/2]	789

11.49.3.28 swap_interval() [2/2]	789
11.49.3.29 valid()	790
11.50 FI_Glut_Bitmap_Font Struct Reference	790
11.50.1 Detailed Description	790
11.51 FI_Glut_StrokeChar Struct Reference	790
11.52 FI_Glut_StrokeFont Struct Reference	790
11.53 FI_Glut_StrokeStrip Struct Reference	791
11.54 FI_Glut_StrokeVertex Struct Reference	791
11.55 FI_Glut_Window Class Reference	791
11.55.1 Detailed Description	805
11.55.2 Member Function Documentation	805
11.55.2.1 draw()	805
11.55.2.2 draw_overlay()	806
11.55.2.3 handle()	806
11.56 FI_Grid Class Reference	806
11.56.1 Detailed Description	817
11.56.2 Constructor & Destructor Documentation	818
11.56.2.1 FI_Grid()	818
11.56.3 Member Function Documentation	818
11.56.3.1 cell() [1/2]	818
11.56.3.2 cell() [2/2]	819
11.56.3.3 clear_layout()	819
11.56.3.4 col_gap() [1/2]	819
11.56.3.5 col_gap() [2/2]	820
11.56.3.6 col_weight() [1/2]	820
11.56.3.7 col_weight() [2/2]	820
11.56.3.8 col_width() [1/2]	821
11.56.3.9 col_width() [2/2]	821
11.56.3.10 debug()	821
11.56.3.11 draw()	822
11.56.3.12 draw_grid()	822
11.56.3.13 gap() [1/2]	822
11.56.3.14 gap() [2/2]	822
11.56.3.15 layout() [1/2]	823
11.56.3.16 layout() [2/2]	823
11.56.3.17 margin() [1/2]	824
11.56.3.18 margin() [2/2]	824
11.56.3.19 need_layout()	824
11.56.3.20 on_remove()	825
11.56.3.21 resize()	825
11.56.3.22 row_gap() [1/2]	825
11.56.3.23 row_gap() [2/2]	825

11.56.3.24 row_height() [1/2]	826
11.56.3.25 row_height() [2/2]	826
11.56.3.26 row_weight() [1/2]	826
11.56.3.27 row_weight() [2/2]	826
11.56.3.28 show_grid() [1/2]	827
11.56.3.29 show_grid() [2/2]	827
11.56.3.30 widget() [1/2]	827
11.56.3.31 widget() [2/2]	828
11.57 FI_Group Class Reference	828
11.57.1 Detailed Description	837
11.57.2 Constructor & Destructor Documentation	837
11.57.2.1 FI_Group()	837
11.57.2.2 ~FI_Group()	838
11.57.3 Member Function Documentation	838
11.57.3.1 array()	838
11.57.3.2 as_group() [1/2]	838
11.57.3.3 as_group() [2/2]	838
11.57.3.4 begin()	838
11.57.3.5 bounds()	839
11.57.3.6 child()	839
11.57.3.7 clear()	839
11.57.3.8 clip_children() [1/2]	840
11.57.3.9 clip_children() [2/2]	840
11.57.3.10 current() [1/2]	840
11.57.3.11 current() [2/2]	840
11.57.3.12 delete_child()	840
11.57.3.13 draw()	841
11.57.3.14 draw_child()	841
11.57.3.15 draw_children()	841
11.57.3.16 end()	841
11.57.3.17 find()	842
11.57.3.18 focus()	842
11.57.3.19 handle()	842
11.57.3.20 init_sizes()	842
11.57.3.21 insert() [1/2]	843
11.57.3.22 insert() [2/2]	843
11.57.3.23 on_insert()	843
11.57.3.24 on_move()	843
11.57.3.25 on_remove()	844
11.57.3.26 remove() [1/3]	844
11.57.3.27 remove() [2/3]	844
11.57.3.28 remove() [3/3]	845

11.57.3.29 resizable() [1/3]	845
11.57.3.30 resizable() [2/3]	845
11.57.3.31 resizable() [3/3]	845
11.57.3.32 resize()	846
11.57.3.33 sizes()	847
11.57.3.34 update_child()	847
11.58 FI_Help_Block Struct Reference	848
11.59 FI_Help_Dialog Class Reference	848
11.59.1 Detailed Description	849
11.59.2 Member Function Documentation	849
11.59.2.1 load()	849
11.59.2.2 show()	850
11.59.2.3 textsize()	850
11.59.2.4 value() [1/2]	850
11.59.2.5 value() [2/2]	850
11.60 FI_Help_Font_Stack Struct Reference	850
11.61 FI_Help_Font_Style Struct Reference	851
11.61.1 Detailed Description	851
11.62 FI_Help_Link Struct Reference	851
11.62.1 Detailed Description	851
11.63 FI_Help_Target Struct Reference	852
11.63.1 Detailed Description	852
11.64 FI_Help_View Class Reference	852
11.64.1 Detailed Description	862
11.64.2 Constructor & Destructor Documentation	863
11.64.2.1 ~FI_Help_View()	863
11.64.3 Member Function Documentation	863
11.64.3.1 copy()	863
11.64.3.2 draw()	864
11.64.3.3 find()	864
11.64.3.4 handle()	864
11.64.3.5 leftline()	864
11.64.3.6 link()	865
11.64.3.7 load()	865
11.64.3.8 resize()	865
11.64.3.9 scrollbar_size() [1/2]	865
11.64.3.10 scrollbar_size() [2/2]	866
11.64.3.11 text_selected()	866
11.64.3.12 topline() [1/2]	866
11.64.3.13 topline() [2/2]	866
11.64.3.14 value()	867
11.65 FI_Hold_Browser Class Reference	867

11.65.1 Detailed Description	880
11.65.2 Constructor & Destructor Documentation	880
11.65.2.1 FI_Hold_Browser()	880
11.66 FI_Hor_Fill_Slider Class Reference	881
11.67 FI_Hor_Nice_Slider Class Reference	889
11.67.1 Detailed Description	897
11.68 FI_Hor_Slider Class Reference	898
11.68.1 Detailed Description	906
11.69 FI_Hor_Value_Slider Class Reference	906
11.70 FI_ICO_Image Class Reference	915
11.70.1 Detailed Description	918
11.70.2 Constructor & Destructor Documentation	918
11.70.2.1 FI_ICO_Image()	918
11.71 FI_Image Class Reference	919
11.71.1 Detailed Description	921
11.71.2 Constructor & Destructor Documentation	921
11.71.2.1 FI_Image()	921
11.71.3 Member Function Documentation	921
11.71.3.1 as_shared_image()	921
11.71.3.2 color_average()	922
11.71.3.3 copy() [1/2]	922
11.71.3.4 copy() [2/2]	922
11.71.3.5 count()	923
11.71.3.6 d()	923
11.71.3.7 data() [1/2]	923
11.71.3.8 data() [2/2]	923
11.71.3.9 desaturate()	924
11.71.3.10 draw() [1/2]	924
11.71.3.11 draw() [2/2]	924
11.71.3.12 draw_empty()	924
11.71.3.13 draw_scaled()	924
11.71.3.14 fail()	926
11.71.3.15 h() [1/2]	926
11.71.3.16 h() [2/2]	926
11.71.3.17 inactive()	926
11.71.3.18 label() [1/2]	927
11.71.3.19 label() [2/2]	927
11.71.3.20 ld() [1/2]	927
11.71.3.21 ld() [2/2]	927
11.71.3.22 release()	927
11.71.3.23 RGB_scaling()	928
11.71.3.24 scale()	928

11.71.3.25 scaling_algorithm()	928
11.71.3.26 uncache()	928
11.71.3.27 w() [1/2]	929
11.71.3.28 w() [2/2]	929
11.72 FI_Image_Reader Class Reference	929
11.73 FI_Image_Surface Class Reference	929
11.73.1 Detailed Description	931
11.73.2 Constructor & Destructor Documentation	932
11.73.2.1 FI_Image_Surface()	932
11.73.3 Member Function Documentation	932
11.73.3.1 highres_image()	932
11.73.3.2 image()	932
11.73.3.3 is_current()	932
11.73.3.4 mask()	933
11.73.3.5 offscreen()	933
11.73.3.6 origin() [1/2]	934
11.73.3.7 origin() [2/2]	934
11.73.3.8 printable_rect()	934
11.73.3.9 rescale()	934
11.73.3.10 set_current()	935
11.73.3.11 translate()	935
11.73.3.12 untranslate()	935
11.74 FI_Input Class Reference	935
11.74.1 Detailed Description	945
11.74.2 Constructor & Destructor Documentation	946
11.74.2.1 FI_Input()	946
11.74.3 Member Function Documentation	947
11.74.3.1 draw()	947
11.74.3.2 handle()	947
11.74.3.3 handle_key()	947
11.74.3.4 handle_rmb()	948
11.75 FI_Input_ Class Reference	948
11.75.1 Detailed Description	957
11.75.2 Constructor & Destructor Documentation	957
11.75.2.1 FI_Input_()	957
11.75.2.2 ~FI_Input_()	958
11.75.3 Member Function Documentation	958
11.75.3.1 append()	958
11.75.3.2 apply_undo()	958
11.75.3.3 can_redo()	959
11.75.3.4 can_undo()	959
11.75.3.5 copy()	959

11.75.3.6 copy_cuts()	959
11.75.3.7 cursor_color() [1/2]	959
11.75.3.8 cursor_color() [2/2]	960
11.75.3.9 cut() [1/3]	960
11.75.3.10 cut() [2/3]	960
11.75.3.11 cut() [3/3]	960
11.75.3.12 drawtext() [1/2]	961
11.75.3.13 drawtext() [2/2]	961
11.75.3.14 dvalue()	961
11.75.3.15 handle_mouse()	962
11.75.3.16 handletext()	962
11.75.3.17 index()	962
11.75.3.18 input_type() [1/2]	962
11.75.3.19 input_type() [2/2]	962
11.75.3.20 insert()	963
11.75.3.21 insert_position() [1/3]	963
11.75.3.22 insert_position() [2/3]	963
11.75.3.23 insert_position() [3/3]	964
11.75.3.24 ivalue()	964
11.75.3.25 line_end()	964
11.75.3.26 line_start()	964
11.75.3.27 mark() [1/2]	965
11.75.3.28 mark() [2/2]	965
11.75.3.29 maximum_size() [1/2]	965
11.75.3.30 maximum_size() [2/2]	965
11.75.3.31 position() [1/3]	965
11.75.3.32 position() [2/3]	966
11.75.3.33 position() [3/3]	966
11.75.3.34 readonly() [1/2]	966
11.75.3.35 readonly() [2/2]	966
11.75.3.36 redo()	966
11.75.3.37 replace()	966
11.75.3.38 resize()	967
11.75.3.39 shortcut() [1/2]	967
11.75.3.40 shortcut() [2/2]	968
11.75.3.41 size() [1/2]	968
11.75.3.42 size() [2/2]	968
11.75.3.43 static_value() [1/2]	968
11.75.3.44 static_value() [2/2]	969
11.75.3.45 tab_nav() [1/2]	969
11.75.3.46 tab_nav() [2/2]	969
11.75.3.47 textcolor() [1/2]	970

11.75.3.48 <code>textcolor()</code> [2/2]	970
11.75.3.49 <code>textfont()</code> [1/2]	970
11.75.3.50 <code>textfont()</code> [2/2]	970
11.75.3.51 <code>textsize()</code> [1/2]	970
11.75.3.52 <code>textsize()</code> [2/2]	971
11.75.3.53 <code>undo()</code>	972
11.75.3.54 <code>up_down_position()</code>	972
11.75.3.55 <code>value()</code> [1/5]	972
11.75.3.56 <code>value()</code> [2/5]	972
11.75.3.57 <code>value()</code> [3/5]	973
11.75.3.58 <code>value()</code> [4/5]	973
11.75.3.59 <code>value()</code> [5/5]	973
11.75.3.60 <code>word_end()</code>	974
11.75.3.61 <code>word_start()</code>	974
11.75.3.62 <code>wrap()</code> [1/2]	974
11.75.3.63 <code>wrap()</code> [2/2]	975
11.76 FI_Input_Choice Class Reference	975
11.76.1 Detailed Description	984
11.76.2 Constructor & Destructor Documentation	986
11.76.2.1 <code>FI_Input_Choice()</code>	986
11.76.3 Member Function Documentation	986
11.76.3.1 <code>add()</code>	986
11.76.3.2 <code>draw()</code>	986
11.76.3.3 <code>inp_x()</code>	986
11.76.3.4 <code>input()</code>	986
11.76.3.5 <code>menu_x()</code>	986
11.76.3.6 <code>menubutton()</code>	987
11.76.3.7 <code>resize()</code>	987
11.76.3.8 <code>update_menubutton()</code>	987
11.76.3.9 <code>value()</code> [1/2]	987
11.76.3.10 <code>value()</code> [2/2]	988
11.77 FI_Int_Input Class Reference	988
11.77.1 Detailed Description	998
11.77.2 Constructor & Destructor Documentation	998
11.77.2.1 <code>FI_Int_Input()</code>	998
11.78 FI_JPEG_Image Class Reference	998
11.78.1 Detailed Description	1001
11.78.2 Constructor & Destructor Documentation	1001
11.78.2.1 <code>FI_JPEG_Image()</code> [1/2]	1001
11.78.2.2 <code>FI_JPEG_Image()</code> [2/2]	1002
11.79 FI_Label Struct Reference	1002
11.79.1 Detailed Description	1003

11.79.2 Member Function Documentation	1003
11.79.2.1 draw()	1003
11.79.2.2 measure()	1003
11.79.3 Member Data Documentation	1003
11.79.3.1 type	1003
11.80 FI_Light_Button Class Reference	1004
11.80.1 Detailed Description	1011
11.80.2 Constructor & Destructor Documentation	1011
11.80.2.1 FI_Light_Button()	1011
11.80.3 Member Function Documentation	1012
11.80.3.1 draw()	1012
11.80.3.2 handle()	1012
11.81 FI_Line_Dial Class Reference	1013
11.82 FI_Mac_App_Menu Class Reference	1021
11.82.1 Member Function Documentation	1022
11.82.1.1 custom_application_menu_items()	1022
11.82.2 Member Data Documentation	1023
11.82.2.1 print	1023
11.83 FI_Menu_Class Reference	1023
11.83.1 Detailed Description	1032
11.83.2 Constructor & Destructor Documentation	1032
11.83.2.1 FI_Menu_()	1032
11.83.3 Member Function Documentation	1032
11.83.3.1 add() [1/2]	1032
11.83.3.2 add() [2/2]	1033
11.83.3.3 clear()	1035
11.83.3.4 clear_submenu()	1035
11.83.3.5 copy()	1036
11.83.3.6 down_box()	1036
11.83.3.7 find_index() [1/3]	1036
11.83.3.8 find_index() [2/3]	1036
11.83.3.9 find_index() [3/3]	1037
11.83.3.10 find_item() [1/2]	1037
11.83.3.11 find_item() [2/2]	1038
11.83.3.12 find_item_with_argument()	1038
11.83.3.13 find_item_with_user_data()	1038
11.83.3.14 global()	1039
11.83.3.15 insert()	1039
11.83.3.16 item_pathname()	1040
11.83.3.17 menu() [1/2]	1040
11.83.3.18 menu() [2/2]	1041
11.83.3.19 menu_box() [1/2]	1041

11.83.3.20 menu_box() [2/2]	1041
11.83.3.21 menu_end()	1041
11.83.3.22 mode() [1/2]	1042
11.83.3.23 mode() [2/2]	1042
11.83.3.24 mvalue()	1042
11.83.3.25 picked()	1042
11.83.3.26 prev_mvalue()	1042
11.83.3.27 remove()	1042
11.83.3.28 replace()	1043
11.83.3.29 size()	1043
11.83.3.30 test_shortcut()	1043
11.83.3.31 text() [1/2]	1043
11.83.3.32 text() [2/2]	1043
11.83.3.33 textcolor()	1043
11.83.3.34 textfont() [1/2]	1044
11.83.3.35 textfont() [2/2]	1044
11.83.3.36 textsize() [1/2]	1044
11.83.3.37 textsize() [2/2]	1044
11.83.3.38 value() [1/3]	1044
11.83.3.39 value() [2/3]	1044
11.83.3.40 value() [3/3]	1045
11.84 FI_Menu_Bar Class Reference	1046
11.84.1 Detailed Description	1055
11.84.2 Constructor & Destructor Documentation	1056
11.84.2.1 FI_Menu_Bar()	1056
11.84.3 Member Function Documentation	1056
11.84.3.1 draw()	1056
11.84.3.2 handle()	1056
11.84.3.3 play_menu()	1057
11.84.3.4 update()	1057
11.85 FI_Menu_Button Class Reference	1057
11.85.1 Detailed Description	1066
11.85.2 Member Enumeration Documentation	1067
11.85.2.1 popup_buttons	1067
11.85.3 Constructor & Destructor Documentation	1067
11.85.3.1 FI_Menu_Button()	1067
11.85.4 Member Function Documentation	1067
11.85.4.1 draw()	1067
11.85.4.2 handle()	1068
11.85.4.3 popup()	1068
11.86 FI_Menu_Item Struct Reference	1068
11.86.1 Detailed Description	1072

11.86.2 Member Function Documentation	1073
11.86.2.1 add()	1073
11.86.2.2 argument() [1/2]	1074
11.86.2.3 argument() [2/2]	1074
11.86.2.4 callback() [1/5]	1074
11.86.2.5 callback() [2/5]	1074
11.86.2.6 callback() [3/5]	1074
11.86.2.7 callback() [4/5]	1074
11.86.2.8 callback() [5/5]	1075
11.86.2.9 check()	1075
11.86.2.10 checkbox()	1075
11.86.2.11 checked()	1075
11.86.2.12 deactivate()	1075
11.86.2.13 do_callback() [1/3]	1075
11.86.2.14 do_callback() [2/3]	1075
11.86.2.15 do_callback() [3/3]	1076
11.86.2.16 find_shortcut()	1076
11.86.2.17 image() [1/2]	1076
11.86.2.18 image() [2/2]	1076
11.86.2.19 image_label()	1076
11.86.2.20 insert()	1077
11.86.2.21 label() [1/3]	1077
11.86.2.22 label() [2/3]	1078
11.86.2.23 label() [3/3]	1078
11.86.2.24 labelcolor() [1/2]	1078
11.86.2.25 labelcolor() [2/2]	1078
11.86.2.26 labelfont() [1/2]	1078
11.86.2.27 labelfont() [2/2]	1079
11.86.2.28 labeltype() [1/2]	1079
11.86.2.29 labeltype() [2/2]	1079
11.86.2.30 measure()	1079
11.86.2.31 multi_label()	1079
11.86.2.32 next() [1/2]	1079
11.86.2.33 next() [2/2]	1079
11.86.2.34 popup()	1080
11.86.2.35 pulldown()	1080
11.86.2.36 radio()	1080
11.86.2.37 set()	1080
11.86.2.38 setonly()	1081
11.86.2.39 shortcut()	1081
11.86.2.40 size()	1081
11.86.2.41 submenu()	1081

11.86.2.42 test_shortcut()	1081
11.86.2.43 uncheck()	1082
11.86.2.44 value()	1082
11.87 FI_Menu_Window Class Reference	1082
11.87.1 Detailed Description	1095
11.88 FI_Multi_Browser Class Reference	1095
11.88.1 Detailed Description	1108
11.88.2 Constructor & Destructor Documentation	1108
11.88.2.1 FI_Multi_Browser()	1108
11.89 FI_Multi_Label Struct Reference	1108
11.89.1 Detailed Description	1109
11.89.2 Member Function Documentation	1110
11.89.2.1 label() [1/2]	1110
11.89.2.2 label() [2/2]	1110
11.89.3 Member Data Documentation	1111
11.89.3.1 labela	1111
11.89.3.2 labelb	1111
11.89.3.3 typea	1111
11.89.3.4 typeb	1111
11.90 FI_Multiline_Input Class Reference	1111
11.90.1 Detailed Description	1121
11.90.2 Constructor & Destructor Documentation	1121
11.90.2.1 FI_Multiline_Input()	1121
11.91 FI_Multiline_Output Class Reference	1121
11.91.1 Detailed Description	1131
11.91.2 Constructor & Destructor Documentation	1132
11.91.2.1 FI_Multiline_Output()	1132
11.92 FI_Native_File_Chooser Class Reference	1132
11.92.1 Detailed Description	1133
11.92.2 Member Enumeration Documentation	1134
11.92.2.1 Option	1134
11.92.2.2 Type	1135
11.92.3 Constructor & Destructor Documentation	1135
11.92.3.1 FI_Native_File_Chooser()	1135
11.92.3.2 ~FI_Native_File_Chooser()	1135
11.92.4 Member Function Documentation	1135
11.92.4.1 count()	1135
11.92.4.2 directory()	1135
11.92.4.3 errmsg()	1136
11.92.4.4 filename() [1/2]	1136
11.92.4.5 filename() [2/2]	1136
11.92.4.6 filter() [1/2]	1136

11.92.4.7 filter() [2/2]	1136
11.92.4.8 filter_value() [1/2]	1136
11.92.4.9 filter_value() [2/2]	1137
11.92.4.10 options()	1137
11.92.4.11 preset_file()	1137
11.92.4.12 show()	1137
11.92.4.13 title() [1/2]	1138
11.92.4.14 title() [2/2]	1138
11.93 FI_Nice_Slider Class Reference	1138
11.94 FI_Output Class Reference	1146
11.94.1 Detailed Description	1156
11.94.2 Constructor & Destructor Documentation	1157
11.94.2.1 FI_Output()	1157
11.95 FI_Overlay_Window Class Reference	1157
11.95.1 Detailed Description	1170
11.95.2 Constructor & Destructor Documentation	1170
11.95.2.1 FI_Overlay_Window()	1170
11.95.3 Member Function Documentation	1170
11.95.3.1 as_overlay_window()	1170
11.95.3.2 draw_overlay()	1170
11.95.3.3 flush()	1170
11.95.3.4 hide()	1170
11.95.3.5 redraw_overlay()	1170
11.95.3.6 resize()	1171
11.95.3.7 show()	1171
11.96 FI_Pack Class Reference	1171
11.96.1 Detailed Description	1180
11.96.2 Constructor & Destructor Documentation	1180
11.96.2.1 FI_Pack()	1180
11.96.3 Member Function Documentation	1181
11.96.3.1 clear()	1181
11.96.3.2 draw()	1181
11.96.3.3 horizontal()	1181
11.96.3.4 resize()	1181
11.97 FI_Paged_Device Class Reference	1182
11.97.1 Detailed Description	1184
11.97.2 Member Enumeration Documentation	1184
11.97.2.1 Page_Format	1184
11.97.2.2 Page_Layout	1185
11.97.3 Member Function Documentation	1185
11.97.3.1 begin_job()	1185
11.97.3.2 begin_page()	1186

11.97.3.3 end_job()	1186
11.97.3.4 end_page()	1186
11.97.3.5 margins()	1187
11.97.3.6 rotate()	1187
11.97.3.7 scale()	1187
11.97.3.8 start_job()	1187
11.97.3.9 start_page()	1188
11.98 FI_PDF_File_Surface Class Reference	1188
11.98.1 Detailed Description	1191
11.98.2 Member Function Documentation	1191
11.98.2.1 begin_document()	1191
11.98.2.2 begin_job() [1/2]	1191
11.98.2.3 begin_job() [2/2]	1192
11.98.2.4 begin_page()	1192
11.98.2.5 end_job()	1192
11.98.2.6 end_page()	1192
11.98.2.7 is_current()	1192
11.98.2.8 margins()	1193
11.98.2.9 origin() [1/2]	1193
11.98.2.10 origin() [2/2]	1193
11.98.2.11 printable_rect()	1193
11.98.2.12 rotate()	1194
11.98.2.13 scale()	1194
11.98.2.14 set_current()	1194
11.98.2.15 translate()	1195
11.98.2.16 untranslate()	1195
11.99 FI_Pixmap Class Reference	1195
11.99.1 Detailed Description	1197
11.99.2 Constructor & Destructor Documentation	1198
11.99.2.1 FI_Pixmap() [1/4]	1198
11.99.2.2 FI_Pixmap() [2/4]	1198
11.99.2.3 FI_Pixmap() [3/4]	1198
11.99.2.4 FI_Pixmap() [4/4]	1198
11.99.3 Member Function Documentation	1198
11.99.3.1 color_average()	1198
11.99.3.2 copy()	1198
11.99.3.3 desaturate()	1199
11.99.3.4 draw()	1199
11.99.3.5 label() [1/2]	1199
11.99.3.6 label() [2/2]	1200
11.99.3.7 uncache()	1200
11.100 FI_Plugin Class Reference	1200

11.100.1 Detailed Description	1200
11.100.2 Constructor & Destructor Documentation	1201
11.100.2.1 FI_Plugin()	1201
11.101 FI_Plugin_Manager Class Reference	1201
11.101.1 Detailed Description	1205
11.101.2 Constructor & Destructor Documentation	1205
11.101.2.1 ~FI_Plugin_Manager()	1205
11.101.3 Member Function Documentation	1205
11.101.3.1 addPlugin()	1205
11.101.3.2 load()	1205
11.101.3.3 loadAll()	1205
11.101.3.4 removePlugin()	1205
11.102 FI_PNG_Image Class Reference	1206
11.102.1 Detailed Description	1208
11.102.2 Constructor & Destructor Documentation	1209
11.102.2.1 FI_PNG_Image() [1/2]	1209
11.102.2.2 FI_PNG_Image() [2/2]	1209
11.103 FI_PNM_Image Class Reference	1209
11.103.1 Detailed Description	1212
11.103.2 Constructor & Destructor Documentation	1212
11.103.2.1 FI_PNM_Image()	1212
11.104 FI_Positioner Class Reference	1212
11.104.1 Detailed Description	1220
11.104.2 Constructor & Destructor Documentation	1220
11.104.2.1 FI_Positioner()	1220
11.104.3 Member Function Documentation	1220
11.104.3.1 draw()	1220
11.104.3.2 handle()	1221
11.105 FI_PostScript_File_Device Class Reference	1221
11.105.1 Detailed Description	1224
11.105.2 Member Function Documentation	1225
11.105.2.1 begin_job() [1/3]	1225
11.105.2.2 begin_job() [2/3]	1225
11.105.2.3 begin_job() [3/3]	1226
11.105.2.4 begin_page()	1226
11.105.2.5 end_current()	1226
11.105.2.6 end_job()	1227
11.105.2.7 end_page()	1227
11.105.2.8 margins()	1227
11.105.2.9 origin() [1/2]	1227
11.105.2.10 origin() [2/2]	1228
11.105.2.11 printable_rect()	1228

11.105.2.12 rotate()	1228
11.105.2.13 scale()	1228
11.105.2.14 set_current()	1229
11.105.2.15 start_job() [1/2]	1229
11.105.2.16 start_job() [2/2]	1229
11.105.2.17 translate()	1229
11.105.2.18 untranslate()	1230
11.106 FI_Preferences Class Reference	1230
11.106.1 Detailed Description	1233
11.106.2 Member Typedef Documentation	1234
11.106.2.1 ID	1234
11.106.3 Member Enumeration Documentation	1234
11.106.3.1 Root	1234
11.106.4 Constructor & Destructor Documentation	1235
11.106.4.1 FI_Preferences() [1/8]	1235
11.106.4.2 FI_Preferences() [2/8]	1236
11.106.4.3 FI_Preferences() [3/8]	1237
11.106.4.4 FI_Preferences() [4/8]	1237
11.106.4.5 FI_Preferences() [5/8]	1238
11.106.4.6 FI_Preferences() [6/8]	1238
11.106.4.7 FI_Preferences() [7/8]	1238
11.106.4.8 ~FI_Preferences()	1238
11.106.4.9 FI_Preferences() [8/8]	1238
11.106.5 Member Function Documentation	1239
11.106.5.1 delete_entry()	1239
11.106.5.2 delete_group()	1239
11.106.5.3 dirty()	1239
11.106.5.4 entries()	1239
11.106.5.5 entry()	1240
11.106.5.6 entry_exists()	1241
11.106.5.7 file_access() [1/2]	1241
11.106.5.8 file_access() [2/2]	1241
11.106.5.9 filename() [1/2]	1242
11.106.5.10 filename() [2/2]	1243
11.106.5.11 flush()	1243
11.106.5.12 get() [1/9]	1244
11.106.5.13 get() [2/9]	1244
11.106.5.14 get() [3/9]	1244
11.106.5.15 get() [4/9]	1245
11.106.5.16 get() [5/9]	1245
11.106.5.17 get() [6/9]	1246
11.106.5.18 get() [7/9]	1246

11.106.5.19	get() [8/9]	1246
11.106.5.20	get() [9/9]	1247
11.106.5.21	get_userdata_path()	1247
11.106.5.22	group()	1248
11.106.5.23	group_exists()	1248
11.106.5.24	groups()	1248
11.106.5.25	new_UUID()	1249
11.106.5.26	set() [1/8]	1249
11.106.5.27	set() [2/8]	1249
11.106.5.28	set() [3/8]	1249
11.106.5.29	set() [4/8]	1250
11.106.5.30	set() [5/8]	1250
11.106.5.31	set() [6/8]	1250
11.106.5.32	set() [7/8]	1251
11.106.5.33	set() [8/8]	1251
11.106.5.34	size()	1251
11.106.6	Member Data Documentation	1252
11.106.6.1	CORE_READ_OK	1252
11.106.6.2	CORE_WRITE_OK	1252
11.106.6.3	NONE	1252
11.107	FI_Printer Class Reference	1252
11.107.1	Detailed Description	1256
11.107.2	Member Function Documentation	1257
11.107.2.1	begin_job()	1257
11.107.2.2	begin_page()	1257
11.107.2.3	end_job()	1257
11.107.2.4	end_page()	1257
11.107.2.5	is_current()	1258
11.107.2.6	margins()	1258
11.107.2.7	origin() [1/2]	1258
11.107.2.8	origin() [2/2]	1258
11.107.2.9	printable_rect()	1259
11.107.2.10	rotate()	1259
11.107.2.11	scale()	1259
11.107.2.12	set_current()	1259
11.107.2.13	translate()	1260
11.107.2.14	untranslate()	1260
11.108	FI_Progress Class Reference	1260
11.108.1	Detailed Description	1267
11.108.2	Constructor & Destructor Documentation	1267
11.108.2.1	FI_Progress()	1267
11.108.3	Member Function Documentation	1267

11.108.3.1 draw()	1267
11.108.3.2 maximum() [1/2]	1268
11.108.3.3 maximum() [2/2]	1268
11.108.3.4 minimum() [1/2]	1268
11.108.3.5 minimum() [2/2]	1268
11.108.3.6 value() [1/2]	1268
11.108.3.7 value() [2/2]	1268
11.109 FI_Radio_Button Class Reference	1268
11.109.1 Constructor & Destructor Documentation	1276
11.109.1.1 FI_Radio_Button()	1276
11.110 FI_Radio_Light_Button Class Reference	1276
11.111 FI_Radio_Round_Button Class Reference	1284
11.111.1 Constructor & Destructor Documentation	1292
11.111.1.1 FI_Radio_Round_Button()	1292
11.112 FI_Rect Class Reference	1292
11.112.1 Detailed Description	1293
11.112.2 Constructor & Destructor Documentation	1294
11.112.2.1 FI_Rect()	1294
11.112.3 Member Function Documentation	1294
11.112.3.1 b()	1294
11.112.3.2 inset() [1/3]	1294
11.112.3.3 inset() [2/3]	1294
11.112.3.4 inset() [3/3]	1294
11.112.3.5 r()	1295
11.113 FI_Scroll::FI_Region_LRTB Struct Reference	1295
11.113.1 Detailed Description	1295
11.114 FI_Scroll::FI_Region_XYWH Struct Reference	1295
11.114.1 Detailed Description	1295
11.115 FI_Repeat_Button Class Reference	1296
11.115.1 Detailed Description	1303
11.115.2 Constructor & Destructor Documentation	1303
11.115.2.1 FI_Repeat_Button()	1303
11.115.3 Member Function Documentation	1303
11.115.3.1 handle()	1303
11.116 FI_Return_Button Class Reference	1304
11.116.1 Detailed Description	1312
11.116.2 Constructor & Destructor Documentation	1312
11.116.2.1 FI_Return_Button()	1312
11.116.3 Member Function Documentation	1312
11.116.3.1 draw()	1312
11.116.3.2 handle()	1312
11.117 FI_RGB_Image Class Reference	1313

11.117.1 Detailed Description	1316
11.117.2 Constructor & Destructor Documentation	1316
11.117.2.1 FI_RGB_Image() [1/3]	1316
11.117.2.2 FI_RGB_Image() [2/3]	1317
11.117.2.3 FI_RGB_Image() [3/3]	1317
11.117.3 Member Function Documentation	1317
11.117.3.1 as_svg_image()	1317
11.117.3.2 color_average()	1317
11.117.3.3 copy()	1318
11.117.3.4 desaturate()	1318
11.117.3.5 draw()	1318
11.117.3.6 label() [1/2]	1319
11.117.3.7 label() [2/2]	1319
11.117.3.8 max_size() [1/2]	1319
11.117.3.9 max_size() [2/2]	1319
11.117.3.10 normalize()	1319
11.117.3.11 uncache()	1319
11.117.4 Member Data Documentation	1320
11.117.4.1 array	1320
11.118 FI_Roller Class Reference	1320
11.118.1 Detailed Description	1328
11.118.2 Constructor & Destructor Documentation	1328
11.118.2.1 FI_Roller()	1328
11.118.3 Member Function Documentation	1328
11.118.3.1 draw()	1328
11.118.3.2 handle()	1328
11.119 FI_Round_Button Class Reference	1329
11.119.1 Detailed Description	1337
11.119.2 Constructor & Destructor Documentation	1337
11.119.2.1 FI_Round_Button()	1337
11.120 FI_Round_Clock Class Reference	1337
11.120.1 Detailed Description	1345
11.120.2 Constructor & Destructor Documentation	1345
11.120.2.1 FI_Round_Clock()	1345
11.121 FI_Scheme Class Reference	1346
11.121.1 Member Function Documentation	1346
11.121.1.1 add_scheme_name()	1346
11.121.1.2 names()	1347
11.121.1.3 num_schemes()	1347
11.122 FI_Scheme_Choice Class Reference	1347
11.122.1 Constructor & Destructor Documentation	1357
11.122.1.1 FI_Scheme_Choice()	1357

11.122.2 Member Function Documentation	1357
11.122.2.1 handle()	1357
11.122.2.2 init_value()	1357
11.122.2.3 scheme_cb_()	1357
11.123 FI_Scroll Class Reference	1358
11.123.1 Detailed Description	1367
11.123.2 Constructor & Destructor Documentation	1368
11.123.2.1 FI_Scroll()	1368
11.123.2.2 ~FI_Scroll()	1368
11.123.3 Member Function Documentation	1368
11.123.3.1 bbox()	1368
11.123.3.2 delete_child()	1369
11.123.3.3 draw()	1369
11.123.3.4 fix_scrollbar_order()	1369
11.123.3.5 handle()	1370
11.123.3.6 on_insert()	1370
11.123.3.7 on_move()	1371
11.123.3.8 recalc_scrollbars()	1371
11.123.3.9 resize()	1371
11.123.3.10 scroll_to()	1372
11.123.3.11 scrollbar_size() [1/2]	1372
11.123.3.12 scrollbar_size() [2/2]	1372
11.123.3.13 xposition()	1373
11.123.3.14 yposition()	1373
11.124 FI_Scrollbar Class Reference	1373
11.124.1 Detailed Description	1382
11.124.2 Constructor & Destructor Documentation	1382
11.124.2.1 FI_Scrollbar()	1382
11.124.3 Member Function Documentation	1382
11.124.3.1 draw()	1382
11.124.3.2 handle()	1383
11.124.3.3 linesize()	1383
11.124.3.4 value() [1/3]	1383
11.124.3.5 value() [2/3]	1384
11.124.3.6 value() [3/3]	1384
11.125 FI_Scroll::FI_Scrollbar_Data Struct Reference	1384
11.125.1 Detailed Description	1385
11.126 FI_Secret_Input Class Reference	1385
11.126.1 Detailed Description	1394
11.126.2 Constructor & Destructor Documentation	1394
11.126.2.1 FI_Secret_Input()	1394
11.126.3 Member Function Documentation	1395

11.126.3.1 handle()	1395
11.127 FI_Select_Browser Class Reference	1395
11.127.1 Detailed Description	1408
11.127.2 Constructor & Destructor Documentation	1408
11.127.2.1 FI_Select_Browser()	1408
11.128 FI_Shared_Image Class Reference	1409
11.128.1 Detailed Description	1412
11.128.2 Constructor & Destructor Documentation	1412
11.128.2.1 FI_Shared_Image() [1/2]	1412
11.128.2.2 FI_Shared_Image() [2/2]	1413
11.128.2.3 ~FI_Shared_Image()	1413
11.128.3 Member Function Documentation	1413
11.128.3.1 add()	1413
11.128.3.2 add_handler()	1413
11.128.3.3 as_shared_image()	1413
11.128.3.4 color_average()	1414
11.128.3.5 compare()	1414
11.128.3.6 copy() [1/2]	1415
11.128.3.7 copy() [2/2]	1415
11.128.3.8 copy_()	1415
11.128.3.9 desaturate()	1415
11.128.3.10 draw()	1416
11.128.3.11 find()	1416
11.128.3.12 get() [1/2]	1416
11.128.3.13 get() [2/2]	1417
11.128.3.14 image()	1417
11.128.3.15 images()	1418
11.128.3.16 num_images()	1418
11.128.3.17 original()	1418
11.128.3.18 refcount()	1418
11.128.3.19 release()	1419
11.128.3.20 uncache()	1419
11.128.3.21 update()	1419
11.129 FI_Shortcut_Button Class Reference	1419
11.129.1 Detailed Description	1427
11.129.2 Constructor & Destructor Documentation	1427
11.129.2.1 FI_Shortcut_Button()	1427
11.129.3 Member Function Documentation	1427
11.129.3.1 draw()	1427
11.129.3.2 handle()	1427
11.129.3.3 value() [1/2]	1427
11.129.3.4 value() [2/2]	1428

11.130 FI_Simple_Counter Class Reference	1428
11.130.1 Detailed Description	1436
11.131 FI_Single_Window Class Reference	1437
11.131.1 Detailed Description	1449
11.131.2 Member Function Documentation	1449
11.131.2.1 flush()	1449
11.131.2.2 show()	1449
11.132 FI_Slider Class Reference	1450
11.132.1 Detailed Description	1458
11.132.2 Constructor & Destructor Documentation	1458
11.132.2.1 FI_Slider()	1458
11.132.3 Member Function Documentation	1459
11.132.3.1 bounds()	1459
11.132.3.2 draw()	1459
11.132.3.3 handle()	1459
11.132.3.4 scrollvalue()	1460
11.132.3.5 slider_size()	1460
11.133 FI_Spinner Class Reference	1460
11.133.1 Detailed Description	1470
11.133.2 Constructor & Destructor Documentation	1470
11.133.2.1 FI_Spinner()	1470
11.133.3 Member Function Documentation	1470
11.133.3.1 draw()	1470
11.133.3.2 handle()	1470
11.133.3.3 resize()	1471
11.133.3.4 step() [1/2]	1471
11.133.3.5 step() [2/2]	1471
11.133.3.6 type() [1/2]	1472
11.133.3.7 type() [2/2]	1472
11.133.3.8 value()	1472
11.133.3.9 wrap() [1/2]	1472
11.133.3.10 wrap() [2/2]	1472
11.134 FI_Spinner::FI_Spinner_Input Class Reference	1473
11.134.1 Member Function Documentation	1483
11.134.1.1 handle()	1483
11.135 FI_Surface_Device Class Reference	1483
11.135.1 Detailed Description	1484
11.135.2 Member Function Documentation	1484
11.135.2.1 end_current()	1484
11.135.2.2 is_current()	1484
11.135.2.3 pop_current()	1484
11.135.2.4 push_current()	1485

11.135.2.5 set_current()	1485
11.135.2.6 surface()	1485
11.136 FI_SVG_File_Surface Class Reference	1485
11.136.1 Detailed Description	1487
11.136.2 Constructor & Destructor Documentation	1488
11.136.2.1 FI_SVG_File_Surface()	1488
11.136.2.2 ~FI_SVG_File_Surface()	1488
11.136.3 Member Function Documentation	1488
11.136.3.1 close()	1488
11.136.3.2 origin() [1/2]	1488
11.136.3.3 origin() [2/2]	1488
11.136.3.4 printable_rect()	1489
11.136.3.5 translate()	1489
11.136.3.6 untranslate()	1489
11.137 FI_SVG_Image Class Reference	1489
11.137.1 Detailed Description	1493
11.137.2 Constructor & Destructor Documentation	1494
11.137.2.1 FI_SVG_Image() [1/3]	1494
11.137.2.2 FI_SVG_Image() [2/3]	1494
11.137.2.3 FI_SVG_Image() [3/3]	1494
11.137.3 Member Function Documentation	1495
11.137.3.1 as_svg_image()	1495
11.137.3.2 color_average()	1495
11.137.3.3 copy()	1495
11.137.3.4 desaturate()	1496
11.137.3.5 draw()	1496
11.137.3.6 normalize()	1496
11.137.3.7 resize()	1496
11.137.4 Member Data Documentation	1496
11.137.4.1 proportional	1496
11.138 FI_Sys_Menu_Bar Class Reference	1497
11.138.1 Detailed Description	1507
11.138.2 Member Enumeration Documentation	1508
11.138.2.1 window_menu_style_enum	1508
11.138.3 Constructor & Destructor Documentation	1508
11.138.3.1 FI_Sys_Menu_Bar()	1508
11.138.4 Member Function Documentation	1508
11.138.4.1 about()	1508
11.138.4.2 add() [1/3]	1508
11.138.4.3 add() [2/3]	1509
11.138.4.4 add() [3/3]	1509
11.138.4.5 clear()	1509

11.138.4.6 clear_submenu()	1510
11.138.4.7 create_window_menu()	1510
11.138.4.8 draw()	1510
11.138.4.9 insert() [1/2]	1510
11.138.4.10 insert() [2/2]	1510
11.138.4.11 menu()	1511
11.138.4.12 mode()	1511
11.138.4.13 play_menu()	1511
11.138.4.14 remove()	1511
11.138.4.15 replace()	1512
11.138.4.16 update()	1512
11.138.4.17 window_menu_style()	1512
11.139 FI_Table Class Reference	1512
11.139.1 Detailed Description	1526
11.139.2 Member Enumeration Documentation	1528
11.139.2.1 TableContext	1528
11.139.3 Constructor & Destructor Documentation	1528
11.139.3.1 FI_Table()	1528
11.139.3.2 ~FI_Table()	1528
11.139.4 Member Function Documentation	1528
11.139.4.1 array()	1528
11.139.4.2 callback()	1528
11.139.4.3 callback_col()	1529
11.139.4.4 callback_context()	1529
11.139.4.5 callback_row()	1529
11.139.4.6 child()	1529
11.139.4.7 children()	1530
11.139.4.8 clear()	1530
11.139.4.9 col_header()	1530
11.139.4.10 col_resize()	1530
11.139.4.11 col_resize_min()	1530
11.139.4.12 col_width()	1530
11.139.4.13 col_width_all()	1531
11.139.4.14 cursor2rowcol()	1531
11.139.4.15 damage_zone()	1531
11.139.4.16 do_callback()	1531
11.139.4.17 draw()	1531
11.139.4.18 draw_cell()	1531
11.139.4.19 find_cell()	1533
11.139.4.20 get_selection()	1533
11.139.4.21 handle()	1533
11.139.4.22 init_sizes()	1533

11.139.4.23	insert()	1534
11.139.4.24	is_interactive_resize()	1534
11.139.4.25	is_selected()	1534
11.139.4.26	move_cursor()	1534
11.139.4.27	recalc_dimensions()	1534
11.139.4.28	redraw_range()	1534
11.139.4.29	resize()	1535
11.139.4.30	row_col_clamp()	1535
11.139.4.31	row_header()	1535
11.139.4.32	row_height()	1535
11.139.4.33	row_height_all()	1535
11.139.4.34	row_resize()	1535
11.139.4.35	row_resize_min()	1536
11.139.4.36	rows()	1536
11.139.4.37	scrollbar_size() [1/2]	1536
11.139.4.38	scrollbar_size() [2/2]	1536
11.139.4.39	set_selection()	1536
11.139.4.40	tab_cell_nav() [1/2]	1537
11.139.4.41	tab_cell_nav() [2/2]	1537
11.139.4.42	table_box()	1537
11.139.4.43	table_resized()	1537
11.139.4.44	table_scrolled()	1537
11.139.4.45	top_row() [1/2]	1538
11.139.4.46	top_row() [2/2]	1538
11.139.4.47	visible_cells()	1538
11.139.4.48	when()	1538
11.140	FI_Table_Row Class Reference	1539
11.140.1	Detailed Description	1552
11.140.2	Constructor & Destructor Documentation	1552
11.140.2.1	FI_Table_Row()	1552
11.140.2.2	~FI_Table_Row()	1553
11.140.3	Member Function Documentation	1553
11.140.3.1	clear()	1553
11.140.3.2	handle()	1553
11.140.3.3	row_selected()	1553
11.140.3.4	rows()	1553
11.140.3.5	select_all_rows()	1554
11.140.3.6	select_row()	1554
11.140.3.7	type()	1554
11.141	FI_Tabs Class Reference	1555
11.141.1	Detailed Description	1564
11.141.2	Member Enumeration Documentation	1567

11.141.2.1 anonymous enum	1567
11.141.3 Constructor & Destructor Documentation	1568
11.141.3.1 FI_Tabs()	1568
11.141.4 Member Function Documentation	1568
11.141.4.1 clear_tab_positions()	1568
11.141.4.2 client_area()	1568
11.141.4.3 draw()	1569
11.141.4.4 draw_tab()	1569
11.141.4.5 handle()	1569
11.141.4.6 handle_overflow()	1570
11.141.4.7 handle_overflow_menu()	1570
11.141.4.8 hit_close()	1570
11.141.4.9 hit_overflow_menu()	1571
11.141.4.10 hit_tabs_area()	1571
11.141.4.11 maybe_do_callback()	1571
11.141.4.12 on_insert()	1572
11.141.4.13 on_move()	1572
11.141.4.14 on_remove()	1572
11.141.4.15 push() [1/2]	1572
11.141.4.16 push() [2/2]	1572
11.141.4.17 redraw_tabs()	1572
11.141.4.18 resize()	1572
11.141.4.19 show()	1573
11.141.4.20 tab_align() [1/2]	1573
11.141.4.21 tab_align() [2/2]	1573
11.141.4.22 tab_height()	1573
11.141.4.23 tab_positions()	1573
11.141.4.24 take_focus()	1574
11.141.4.25 value() [1/2]	1574
11.141.4.26 value() [2/2]	1575
11.141.4.27 which()	1576
11.141.5 Member Data Documentation	1576
11.141.5.1 overflow_type	1576
11.141.5.2 tab_count	1576
11.141.5.3 tab_flags	1576
11.141.5.4 tab_pos	1577
11.141.5.5 tab_width	1577
11.142 FI_Terminal Class Reference	1577
11.142.1 Detailed Description	1592
11.142.2 FI_Terminal	1593
11.142.2.1 Examples	1594
11.142.2.2 Writing To Terminal From Applications	1594

11.142.2.3 Text Attributes	1595
11.142.2.4 Text and Background Colors	1595
11.142.2.5 Features	1596
11.142.2.6 Margins	1597
11.142.2.7 Caveats	1598
11.142.3 Member Enumeration Documentation	1598
11.142.3.1 Attrib	1598
11.142.3.2 CharFlags	1598
11.142.3.3 OutFlags	1599
11.142.3.4 RedrawStyle	1599
11.142.3.5 ScrollbarStyle	1599
11.142.4 Constructor & Destructor Documentation	1599
11.142.4.1 Fl_Terminal() [1/2]	1599
11.142.4.2 Fl_Terminal() [2/2]	1600
11.142.4.3 ~Fl_Terminal()	1600
11.142.5 Member Function Documentation	1600
11.142.5.1 ansi() [1/2]	1600
11.142.5.2 ansi() [2/2]	1600
11.142.5.3 append()	1601
11.142.5.4 append_ascii()	1601
11.142.5.5 append_utf8()	1602
11.142.5.6 box()	1602
11.142.5.7 clear() [1/2]	1602
11.142.5.8 clear() [2/2]	1602
11.142.5.9 clear_screen()	1602
11.142.5.10 clear_screen_home()	1603
11.142.5.11 color()	1603
11.142.5.12 cursor_col()	1603
11.142.5.13 cursor_cr()	1603
11.142.5.14 cursor_down()	1603
11.142.5.15 cursor_right()	1603
11.142.5.16 cursor_row()	1604
11.142.5.17 cursor_up()	1604
11.142.5.18 delete_rows()	1604
11.142.5.19 display_columns() [1/2]	1604
11.142.5.20 display_columns() [2/2]	1604
11.142.5.21 display_rows() [1/2]	1604
11.142.5.22 display_rows() [2/2]	1604
11.142.5.23 draw()	1605
11.142.5.24 draw_buff()	1605
11.142.5.25 draw_row()	1605
11.142.5.26 draw_row_bg()	1605

11.142.5.27 error_char() [1/2]	1605
11.142.5.28 error_char() [2/2]	1606
11.142.5.29 get_selection()	1606
11.142.5.30 h_to_row()	1606
11.142.5.31 handle()	1607
11.142.5.32 handle_unknown_char() [1/2]	1607
11.142.5.33 handle_unknown_char() [2/2]	1607
11.142.5.34 history_lines()	1607
11.142.5.35 history_use()	1607
11.142.5.36 hscrollbar_style() [1/2]	1607
11.142.5.37 hscrollbar_style() [2/2]	1608
11.142.5.38 insert_char()	1608
11.142.5.39 insert_rows()	1608
11.142.5.40 is_inside_selection()	1608
11.142.5.41 output_translate()	1609
11.142.5.42 plot_char() [1/2]	1609
11.142.5.43 plot_char() [2/2]	1609
11.142.5.44 print_char() [1/2]	1610
11.142.5.45 print_char() [2/2]	1610
11.142.5.46 printf()	1610
11.142.5.47 redraw_rate()	1611
11.142.5.48 redraw_style() [1/2]	1611
11.142.5.49 redraw_style() [2/2]	1611
11.142.5.50 reset_terminal()	1611
11.142.5.51 resize()	1612
11.142.5.52 scroll()	1612
11.142.5.53 scrollbar_actual_size()	1612
11.142.5.54 scrollbar_size() [1/2]	1612
11.142.5.55 scrollbar_size() [2/2]	1612
11.142.5.56 selection_extend()	1613
11.142.5.57 selection_text()	1613
11.142.5.58 selection_text_len()	1613
11.142.5.59 show_unknown() [1/2]	1613
11.142.5.60 show_unknown() [2/2]	1613
11.142.5.61 text()	1613
11.142.5.62 textattrib() [1/2]	1614
11.142.5.63 textattrib() [2/2]	1614
11.142.5.64 textbgcolor()	1614
11.142.5.65 textbgcolor_default() [1/2]	1615
11.142.5.66 textbgcolor_default() [2/2]	1615
11.142.5.67 textbgcolor_xterm()	1615
11.142.5.68 textcolor()	1616

11.142.5.69 textfgcolor()	1616
11.142.5.70 textfgcolor_default() [1/2]	1616
11.142.5.71 textfgcolor_default() [2/2]	1616
11.142.5.72 textfgcolor_xterm()	1617
11.142.5.73 textfont()	1617
11.142.5.74 textsize()	1617
11.142.5.75 u8c_disp_row()	1617
11.142.5.76 u8c_hist_row()	1618
11.142.5.77 u8c_hist_use_row()	1618
11.142.5.78 u8c_ring_row()	1618
11.142.5.79 vprintf()	1618
11.142.5.80 w_to_col()	1619
11.142.5.81 walk_selection()	1619
11.142.6 Member Data Documentation	1619
11.142.6.1 hscrollbar	1619
11.142.6.2 scrollbar	1620
11.143 FI_Text_Buffer Class Reference	1620
11.143.1 Detailed Description	1625
11.143.2 Constructor & Destructor Documentation	1625
11.143.2.1 FI_Text_Buffer()	1625
11.143.3 Member Function Documentation	1625
11.143.3.1 add_modify_callback()	1625
11.143.3.2 address() [1/2]	1626
11.143.3.3 address() [2/2]	1626
11.143.3.4 append()	1626
11.143.3.5 appendfile()	1626
11.143.3.6 byte_at()	1626
11.143.3.7 can_redo()	1627
11.143.3.8 can_undo()	1627
11.143.3.9 canUndo()	1627
11.143.3.10 char_at()	1627
11.143.3.11 copy()	1627
11.143.3.12 count_displayed_characters()	1628
11.143.3.13 count_lines()	1628
11.143.3.14 estimate_lines()	1628
11.143.3.15 findchar_backward()	1628
11.143.3.16 findchar_forward()	1629
11.143.3.17 highlight_text()	1629
11.143.3.18 insert()	1629
11.143.3.19 insert_()	1629
11.143.3.20 insertfile()	1630
11.143.3.21 is_word_separator()	1630

11.143.3.22	length()	1630
11.143.3.23	line_end()	1630
11.143.3.24	line_start()	1630
11.143.3.25	line_text()	1631
11.143.3.26	loadfile()	1631
11.143.3.27	next_char()	1631
11.143.3.28	outputfile()	1631
11.143.3.29	prev_char()	1632
11.143.3.30	printf()	1632
11.143.3.31	remove()	1632
11.143.3.32	remove_()	1632
11.143.3.33	replace()	1633
11.143.3.34	rewind_lines()	1633
11.143.3.35	savefile()	1633
11.143.3.36	search_backward()	1633
11.143.3.37	search_forward()	1634
11.143.3.38	secondary_selection_text()	1634
11.143.3.39	selection_text()	1634
11.143.3.40	skip_displayed_characters()	1634
11.143.3.41	tab_distance()	1635
11.143.3.42	text() [1/2]	1635
11.143.3.43	text() [2/2]	1635
11.143.3.44	text_range()	1635
11.143.3.45	undo()	1636
11.143.3.46	vprintf()	1636
11.143.3.47	word_end()	1636
11.143.3.48	word_start()	1636
11.143.4	Member Data Documentation	1637
11.143.4.1	file_encoding_warning_message	1637
11.143.4.2	mTabDist	1637
11.143.4.3	transcoding_warning_action	1637
11.144	FI_Text_Display Class Reference	1637
11.144.1	Detailed Description	1652
11.144.2	Member Enumeration Documentation	1654
11.144.2.1	anonymous enum	1654
11.144.2.2	anonymous enum	1654
11.144.2.3	anonymous enum	1654
11.144.3	Constructor & Destructor Documentation	1655
11.144.3.1	FI_Text_Display()	1655
11.144.3.2	~FI_Text_Display()	1655
11.144.4	Member Function Documentation	1655
11.144.4.1	absolute_top_line_number()	1655

11.144.4.2 buffer() [1/3]	1655
11.144.4.3 buffer() [2/3]	1656
11.144.4.4 buffer() [3/3]	1656
11.144.4.5 buffer_modified_cb()	1656
11.144.4.6 buffer_predelete_cb()	1657
11.144.4.7 calc_last_char()	1657
11.144.4.8 calc_line_starts()	1657
11.144.4.9 clear_rect()	1657
11.144.4.10 col_to_x()	1658
11.144.4.11 count_lines()	1658
11.144.4.12 cursor_color() [1/2]	1658
11.144.4.13 cursor_color() [2/2]	1658
11.144.4.14 cursor_style()	1659
11.144.4.15 display_insert()	1659
11.144.4.16 display_needs_recalc()	1659
11.144.4.17 draw()	1659
11.144.4.18 draw_cursor()	1659
11.144.4.19 draw_line_numbers()	1660
11.144.4.20 draw_range()	1661
11.144.4.21 draw_string()	1661
11.144.4.22 draw_text()	1661
11.144.4.23 draw_vline()	1662
11.144.4.24 empty_vlines()	1662
11.144.4.25 extend_range_for_styles()	1662
11.144.4.26 find_line_end()	1663
11.144.4.27 find_wrap_range()	1663
11.144.4.28 find_x()	1663
11.144.4.29 get_absolute_top_line_number()	1664
11.144.4.30 grammar_underline_color() [1/2]	1664
11.144.4.31 grammar_underline_color() [2/2]	1664
11.144.4.32 handle()	1664
11.144.4.33 handle_rmb()	1664
11.144.4.34 handle_vline()	1665
11.144.4.35 highlight_data()	1665
11.144.4.36 in_selection()	1666
11.144.4.37 insert()	1666
11.144.4.38 insert_position() [1/2]	1667
11.144.4.39 insert_position() [2/2]	1667
11.144.4.40 line_end()	1667
11.144.4.41 line_start()	1667
11.144.4.42 linenumber_align()	1668
11.144.4.43 linenumber_bgcolor()	1668

11.144.4.44 <code>linenumber_fgcolor()</code>	1668
11.144.4.45 <code>linenumber_font()</code>	1668
11.144.4.46 <code>linenumber_format()</code>	1668
11.144.4.47 <code>linenumber_size()</code>	1669
11.144.4.48 <code>linenumber_width()</code>	1669
11.144.4.49 <code>longest_vline()</code>	1669
11.144.4.50 <code>maintain_absolute_top_line_number()</code>	1669
11.144.4.51 <code>maintaining_absolute_top_line_number()</code>	1669
11.144.4.52 <code>measure_deleted_lines()</code>	1670
11.144.4.53 <code>measure_proportional_character()</code>	1670
11.144.4.54 <code>measure_vline()</code>	1670
11.144.4.55 <code>move_down()</code>	1671
11.144.4.56 <code>move_left()</code>	1671
11.144.4.57 <code>move_right()</code>	1671
11.144.4.58 <code>move_up()</code>	1671
11.144.4.59 <code>offset_line_starts()</code>	1671
11.144.4.60 <code>overstrike()</code>	1671
11.144.4.61 <code>position_style()</code>	1672
11.144.4.62 <code>position_to_line()</code>	1672
11.144.4.63 <code>position_to_linecol()</code>	1673
11.144.4.64 <code>position_to_xy()</code>	1673
11.144.4.65 <code>recalc_display()</code>	1673
11.144.4.66 <code>redisplay_range()</code>	1673
11.144.4.67 <code>reset_absolute_top_line_number()</code>	1674
11.144.4.68 <code>resize()</code>	1674
11.144.4.69 <code>rewind_lines()</code>	1674
11.144.4.70 <code>scroll()</code>	1674
11.144.4.71 <code>scroll_()</code>	1675
11.144.4.72 <code>scroll_timer_cb()</code>	1675
11.144.4.73 <code>scrollbar_align()</code> [1/2]	1675
11.144.4.74 <code>scrollbar_align()</code> [2/2]	1675
11.144.4.75 <code>scrollbar_size()</code> [1/2]	1675
11.144.4.76 <code>scrollbar_size()</code> [2/2]	1676
11.144.4.77 <code>scrollbar_width()</code> [1/2]	1676
11.144.4.78 <code>scrollbar_width()</code> [2/2]	1676
11.144.4.79 <code>secondary_selection_color()</code> [1/2]	1676
11.144.4.80 <code>secondary_selection_color()</code> [2/2]	1676
11.144.4.81 <code>shortcut()</code> [1/2]	1677
11.144.4.82 <code>shortcut()</code> [2/2]	1677
11.144.4.83 <code>show_cursor()</code>	1677
11.144.4.84 <code>show_insert_position()</code>	1677
11.144.4.85 <code>skip_lines()</code>	1677

11.144.4.86 spelling_underline_color() [1/2]	1678
11.144.4.87 spelling_underline_color() [2/2]	1678
11.144.4.88 string_width()	1678
11.144.4.89 style_buffer()	1678
11.144.4.90 textcolor() [1/2]	1679
11.144.4.91 textcolor() [2/2]	1679
11.144.4.92 textfont() [1/2]	1679
11.144.4.93 textfont() [2/2]	1679
11.144.4.94 textsize() [1/2]	1679
11.144.4.95 textsize() [2/2]	1679
11.144.4.96 update_h_scrollbar()	1680
11.144.4.97 update_line_starts()	1680
11.144.4.98 update_v_scrollbar()	1680
11.144.4.99 vline_length()	1680
11.144.4.100 word_end()	1680
11.144.4.101 word_start()	1681
11.144.4.102 wrap_mode()	1681
11.144.4.103 wrap_uses_character()	1681
11.144.4.104 wrapped_column()	1682
11.144.4.105 wrapped_line_counter()	1682
11.144.4.106 wrapped_row()	1684
11.144.4.107 x_to_col()	1684
11.144.4.108 xy_to_position()	1685
11.144.4.109 xy_to_rowcol()	1685
11.145 Fl_Text_Editor Class Reference	1685
11.145.1 Detailed Description	1703
11.145.2 Member Function Documentation	1703
11.145.2.1 add_key_binding()	1703
11.145.2.2 handle()	1703
11.145.2.3 insert_mode() [1/2]	1703
11.145.2.4 insert_mode() [2/2]	1703
11.145.2.5 kf_backspace()	1703
11.145.2.6 kf_c_s_move()	1703
11.145.2.7 kf_copy()	1704
11.145.2.8 kf_ctrl_move()	1704
11.145.2.9 kf_cut()	1704
11.145.2.10 kf_default()	1704
11.145.2.11 kf_delete()	1704
11.145.2.12 kf_down()	1704
11.145.2.13 kf_end()	1704
11.145.2.14 kf_enter()	1705
11.145.2.15 kf_home()	1705

11.145.2.16 kf_ignore()	1705
11.145.2.17 kf_insert()	1705
11.145.2.18 kf_left()	1705
11.145.2.19 kf_m_s_move()	1705
11.145.2.20 kf_meta_move()	1705
11.145.2.21 kf_move()	1706
11.145.2.22 kf_page_down()	1706
11.145.2.23 kf_page_up()	1706
11.145.2.24 kf_paste()	1706
11.145.2.25 kf_redo()	1706
11.145.2.26 kf_right()	1706
11.145.2.27 kf_select_all()	1707
11.145.2.28 kf_shift_move()	1707
11.145.2.29 kf_undo()	1707
11.145.2.30 kf_up()	1707
11.145.2.31 remove_key_binding()	1707
11.145.2.32 tab_nav() [1/2]	1707
11.145.2.33 tab_nav() [2/2]	1708
11.145.3 Member Data Documentation	1708
11.145.3.1 global_key_bindings	1708
11.146 FI_Text_Selection Class Reference	1708
11.146.1 Detailed Description	1709
11.146.2 Member Function Documentation	1710
11.146.2.1 end()	1710
11.146.2.2 includes()	1710
11.146.2.3 length()	1710
11.146.2.4 position()	1710
11.146.2.5 selected() [1/3]	1710
11.146.2.6 selected() [2/3]	1710
11.146.2.7 selected() [3/3]	1711
11.146.2.8 set()	1711
11.146.2.9 start()	1712
11.146.2.10 update()	1712
11.147 FI_Tile Class Reference	1712
11.147.1 Detailed Description	1721
11.147.2 Constructor & Destructor Documentation	1723
11.147.2.1 FI_Tile()	1723
11.147.3 Member Function Documentation	1723
11.147.3.1 cursor()	1723
11.147.3.2 drag_intersection()	1724
11.147.3.3 handle()	1724
11.147.3.4 init_size_range()	1724

11.147.3.5	move_intersection()	1725
11.147.3.6	on_insert()	1725
11.147.3.7	on_move()	1725
11.147.3.8	on_remove()	1725
11.147.3.9	position()	1725
11.147.3.10	request_grow_b()	1726
11.147.3.11	request_grow_l()	1726
11.147.3.12	request_grow_r()	1726
11.147.3.13	request_grow_t()	1726
11.147.3.14	request_shrink_b()	1727
11.147.3.15	request_shrink_l()	1727
11.147.3.16	request_shrink_r()	1727
11.147.3.17	request_shrink_t()	1728
11.147.3.18	resize()	1728
11.147.3.19	set_cursor()	1728
11.147.3.20	size_range() [1/2]	1729
11.147.3.21	size_range() [2/2]	1729
11.148	FI_Tiled_Image Class Reference	1729
11.148.1	Detailed Description	1732
11.148.2	Constructor & Destructor Documentation	1732
11.148.2.1	FI_Tiled_Image()	1732
11.148.3	Member Function Documentation	1732
11.148.3.1	color_average()	1732
11.148.3.2	copy()	1733
11.148.3.3	desaturate()	1733
11.148.3.4	draw()	1733
11.149	FI_Timeout Class Reference	1734
11.149.1	Detailed Description	1735
11.149.2	Member Function Documentation	1735
11.149.2.1	add_timeout()	1735
11.149.2.2	current()	1736
11.149.2.3	elapse_timeouts()	1736
11.149.2.4	get()	1736
11.149.2.5	has_timeout()	1737
11.149.2.6	insert()	1737
11.149.2.7	make_current()	1738
11.149.2.8	release()	1738
11.149.2.9	remove_next_timeout()	1738
11.149.2.10	remove_timeout()	1738
11.149.2.11	repeat_timeout()	1739
11.149.2.12	time_to_wait()	1739
11.149.3	Member Data Documentation	1740

11.149.3.1	current_timeout	1740
11.149.3.2	first_timeout	1740
11.149.3.3	free_timeout	1740
11.150	FI_Timer Class Reference	1740
11.150.1	Detailed Description	1747
11.150.2	Constructor & Destructor Documentation	1747
11.150.2.1	FI_Timer()	1747
11.150.3	Member Function Documentation	1747
11.150.3.1	direction() [1/2]	1747
11.150.3.2	direction() [2/2]	1748
11.150.3.3	draw()	1748
11.150.3.4	handle()	1748
11.150.3.5	suspended()	1749
11.151	FI_Toggle_Button Class Reference	1749
11.151.1	Detailed Description	1756
11.151.2	Constructor & Destructor Documentation	1756
11.151.2.1	FI_Toggle_Button()	1756
11.152	FI_Tooltip Class Reference	1757
11.152.1	Detailed Description	1758
11.152.2	Member Function Documentation	1758
11.152.2.1	color() [1/2]	1758
11.152.2.2	color() [2/2]	1759
11.152.2.3	current()	1759
11.152.2.4	delay() [1/2]	1759
11.152.2.5	delay() [2/2]	1759
11.152.2.6	disable()	1759
11.152.2.7	enable()	1759
11.152.2.8	enabled()	1759
11.152.2.9	enter_area()	1759
11.152.2.10	font() [1/2]	1760
11.152.2.11	font() [2/2]	1760
11.152.2.12	hidedelay() [1/2]	1760
11.152.2.13	hidedelay() [2/2]	1760
11.152.2.14	hoverdelay() [1/2]	1760
11.152.2.15	hoverdelay() [2/2]	1760
11.152.2.16	margin_height() [1/2]	1760
11.152.2.17	margin_height() [2/2]	1761
11.152.2.18	margin_width() [1/2]	1761
11.152.2.19	margin_width() [2/2]	1761
11.152.2.20	size() [1/2]	1761
11.152.2.21	size() [2/2]	1761
11.152.2.22	textcolor() [1/2]	1761

11.152.2.23	textcolor() [2/2]	1761
11.152.2.24	wrap_width() [1/2]	1761
11.152.2.25	wrap_width() [2/2]	1761
11.153	FI_Tree Class Reference	1762
11.153.1	Detailed Description	1776
11.153.2	Member Function Documentation	1780
11.153.2.1	add() [1/2]	1780
11.153.2.2	add() [2/2]	1781
11.153.2.3	calc_dimensions()	1781
11.153.2.4	calc_tree()	1782
11.153.2.5	callback_item() [1/2]	1782
11.153.2.6	callback_item() [2/2]	1782
11.153.2.7	callback_reason() [1/2]	1782
11.153.2.8	callback_reason() [2/2]	1783
11.153.2.9	clear()	1783
11.153.2.10	clear_children()	1783
11.153.2.11	close() [1/2]	1783
11.153.2.12	close() [2/2]	1783
11.153.2.13	closeicon() [1/2]	1784
11.153.2.14	closeicon() [2/2]	1784
11.153.2.15	connectorstyle()	1784
11.153.2.16	deselect() [1/2]	1784
11.153.2.17	deselect() [2/2]	1785
11.153.2.18	deselect_all()	1785
11.153.2.19	display()	1786
11.153.2.20	displayed()	1786
11.153.2.21	draw()	1786
11.153.2.22	extend_selection()	1786
11.153.2.23	extend_selection_dir()	1787
11.153.2.24	find_clicked()	1788
11.153.2.25	find_item()	1788
11.153.2.26	first()	1788
11.153.2.27	first_selected_item()	1789
11.153.2.28	first_visible()	1789
11.153.2.29	first_visible_item()	1789
11.153.2.30	get_selected_items()	1790
11.153.2.31	handle()	1790
11.153.2.32	hposition() [1/2]	1790
11.153.2.33	hposition() [2/2]	1790
11.153.2.34	insert()	1791
11.153.2.35	insert_above()	1791
11.153.2.36	is_close() [1/2]	1792

11.153.2.37 is_close() [2/2]	1792
11.153.2.38 is_hscroll_visible()	1793
11.153.2.39 is_open() [1/2]	1793
11.153.2.40 is_open() [2/2]	1793
11.153.2.41 is_scrollbar()	1794
11.153.2.42 is_selected() [1/2]	1794
11.153.2.43 is_selected() [2/2]	1794
11.153.2.44 is_vscroll_visible()	1795
11.153.2.45 item_clicked() [1/2]	1795
11.153.2.46 item_clicked() [2/2]	1795
11.153.2.47 item_draw_mode() [1/3]	1795
11.153.2.48 item_draw_mode() [2/3]	1795
11.153.2.49 item_draw_mode() [3/3]	1795
11.153.2.50 item_labelbgcolor() [1/2]	1796
11.153.2.51 item_labelbgcolor() [2/2]	1796
11.153.2.52 item_labelfgcolor()	1796
11.153.2.53 item_labelfont()	1796
11.153.2.54 item_labelsize()	1796
11.153.2.55 item_pathname()	1796
11.153.2.56 item_reselect_mode() [1/2]	1797
11.153.2.57 item_reselect_mode() [2/2]	1797
11.153.2.58 last()	1797
11.153.2.59 last_selected_item()	1797
11.153.2.60 last_visible()	1798
11.153.2.61 last_visible_item()	1798
11.153.2.62 load()	1798
11.153.2.63 next()	1798
11.153.2.64 next_item()	1799
11.153.2.65 next_selected_item()	1800
11.153.2.66 next_visible_item()	1800
11.153.2.67 open() [1/2]	1801
11.153.2.68 open() [2/2]	1801
11.153.2.69 open_toggle()	1802
11.153.2.70 openicon() [1/2]	1802
11.153.2.71 openicon() [2/2]	1802
11.153.2.72 prev()	1803
11.153.2.73 recalc_tree()	1803
11.153.2.74 remove()	1803
11.153.2.75 resize()	1803
11.153.2.76 root()	1804
11.153.2.77 root_label()	1804
11.153.2.78 scrollbar_size() [1/2]	1804

11.153.2.79 scrollbar_size() [2/2]	1804
11.153.2.80 select() [1/2]	1805
11.153.2.81 select() [2/2]	1805
11.153.2.82 select_all()	1806
11.153.2.83 select_only()	1806
11.153.2.84 select_toggle()	1806
11.153.2.85 selectbox() [1/2]	1807
11.153.2.86 selectbox() [2/2]	1807
11.153.2.87 selectmode() [1/2]	1807
11.153.2.88 selectmode() [2/2]	1807
11.153.2.89 set_item_focus()	1807
11.153.2.90 show_item() [1/2]	1808
11.153.2.91 show_item() [2/2]	1808
11.153.2.92 show_item_bottom()	1808
11.153.2.93 show_item_middle()	1808
11.153.2.94 show_item_top()	1809
11.153.2.95 show_self()	1809
11.153.2.96 showcollapse() [1/2]	1809
11.153.2.97 showcollapse() [2/2]	1809
11.153.2.98 showroot()	1809
11.153.2.99 sortorder()	1810
11.153.2.100 usericon() [1/2]	1810
11.153.2.101 usericon() [2/2]	1810
11.153.2.102 vposition() [1/2]	1810
11.153.2.103 vposition() [2/2]	1810
11.154 FI_Tree_Item Class Reference	1810
11.154.1 Detailed Description	1815
11.154.2 Constructor & Destructor Documentation	1816
11.154.2.1 FI_Tree_Item() [1/2]	1816
11.154.2.2 FI_Tree_Item() [2/2]	1816
11.154.3 Member Function Documentation	1816
11.154.3.1 activate()	1816
11.154.3.2 add() [1/4]	1816
11.154.3.3 add() [2/4]	1817
11.154.3.4 add() [3/4]	1817
11.154.3.5 add() [4/4]	1817
11.154.3.6 calc_item_height()	1817
11.154.3.7 child()	1818
11.154.3.8 deactivate()	1818
11.154.3.9 deparent()	1818
11.154.3.10 depth()	1818
11.154.3.11 deselect_all()	1818

11.154.3.12 draw()	1819
11.154.3.13 draw_horizontal_connector()	1819
11.154.3.14 draw_item_content()	1819
11.154.3.15 draw_vertical_connector()	1820
11.154.3.16 drawbgcolor()	1821
11.154.3.17 drawfgcolor()	1821
11.154.3.18 find_child() [1/2]	1821
11.154.3.19 find_child() [2/2]	1821
11.154.3.20 find_child_item() [1/2]	1821
11.154.3.21 find_child_item() [2/2]	1822
11.154.3.22 find_clicked()	1822
11.154.3.23 find_item()	1822
11.154.3.24 hide_widgets()	1822
11.154.3.25 insert()	1822
11.154.3.26 insert_above()	1823
11.154.3.27 is_visible_r()	1823
11.154.3.28 label()	1823
11.154.3.29 label_h()	1823
11.154.3.30 label_w()	1823
11.154.3.31 label_x()	1824
11.154.3.32 label_y()	1824
11.154.3.33 labelbgcolor() [1/2]	1824
11.154.3.34 labelbgcolor() [2/2]	1824
11.154.3.35 move() [1/2]	1824
11.154.3.36 move() [2/2]	1825
11.154.3.37 move_above()	1825
11.154.3.38 move_below()	1825
11.154.3.39 move_into()	1825
11.154.3.40 next()	1826
11.154.3.41 next_displayed()	1826
11.154.3.42 next_sibling()	1826
11.154.3.43 next_visible()	1826
11.154.3.44 parent()	1826
11.154.3.45 prefs()	1827
11.154.3.46 prev()	1827
11.154.3.47 prev_displayed()	1827
11.154.3.48 prev_sibling()	1827
11.154.3.49 prev_visible()	1827
11.154.3.50 recalc_tree()	1827
11.154.3.51 remove_child() [1/2]	1828
11.154.3.52 remove_child() [2/2]	1828
11.154.3.53 reparent()	1828

11.154.3.54	replace()	1828
11.154.3.55	replace_child()	1829
11.154.3.56	select()	1829
11.154.3.57	select_all()	1829
11.154.3.58	show_self()	1829
11.154.3.59	show_widgets()	1830
11.154.3.60	swap_children() [1/2]	1830
11.154.3.61	swap_children() [2/2]	1830
11.154.3.62	tree() [1/2]	1830
11.154.3.63	tree() [2/2]	1830
11.154.3.64	update_prev_next()	1831
11.154.3.65	userdeicon() [1/2]	1831
11.154.3.66	userdeicon() [2/2]	1831
11.154.3.67	usericon()	1831
11.154.3.68	visible_r()	1831
11.155	Fl_Tree_Item_Array Class Reference	1832
11.155.1	Detailed Description	1833
11.155.2	Constructor & Destructor Documentation	1833
11.155.2.1	Fl_Tree_Item_Array()	1833
11.155.3	Member Function Documentation	1833
11.155.3.1	add()	1833
11.155.3.2	clear()	1833
11.155.3.3	deparent()	1833
11.155.3.4	insert()	1833
11.155.3.5	manage_item_destroy()	1834
11.155.3.6	move()	1834
11.155.3.7	remove() [1/2]	1834
11.155.3.8	remove() [2/2]	1834
11.155.3.9	reparent()	1834
11.155.3.10	replace()	1835
11.156	Fl_Tree_Prefs Class Reference	1835
11.156.1	Detailed Description	1838
11.156.2	Member Function Documentation	1838
11.156.2.1	closedeicon()	1838
11.156.2.2	closeicon()	1838
11.156.2.3	item_draw_mode()	1838
11.156.2.4	item_labelbgcolor() [1/2]	1838
11.156.2.5	item_labelbgcolor() [2/2]	1838
11.156.2.6	marginbottom()	1838
11.156.2.7	opendeicon()	1838
11.156.2.8	openicon() [1/2]	1839
11.156.2.9	openicon() [2/2]	1839

11.156.2.10 selectmode()	1839
11.156.2.11 showcollapse()	1839
11.156.2.12 showroot()	1839
11.156.2.13 sortorder()	1839
11.156.2.14 userdeicon()	1840
11.157 FI_Valuator Class Reference	1840
11.157.1 Detailed Description	1848
11.157.2 Constructor & Destructor Documentation	1848
11.157.2.1 FI_Valuator()	1848
11.157.3 Member Function Documentation	1848
11.157.3.1 format()	1848
11.157.3.2 increment()	1849
11.157.3.3 maximum() [1/2]	1849
11.157.3.4 maximum() [2/2]	1849
11.157.3.5 minimum() [1/2]	1849
11.157.3.6 minimum() [2/2]	1849
11.157.3.7 precision()	1849
11.157.3.8 range()	1849
11.157.3.9 round()	1850
11.157.3.10 step()	1850
11.157.3.11 value() [1/2]	1850
11.157.3.12 value() [2/2]	1850
11.157.3.13 value_damage()	1850
11.158 FI_Value_Input Class Reference	1850
11.158.1 Detailed Description	1859
11.158.2 Constructor & Destructor Documentation	1859
11.158.2.1 FI_Value_Input()	1859
11.158.3 Member Function Documentation	1859
11.158.3.1 cursor_color() [1/2]	1859
11.158.3.2 cursor_color() [2/2]	1860
11.158.3.3 draw()	1860
11.158.3.4 handle()	1860
11.158.3.5 resize()	1861
11.158.3.6 shortcut() [1/2]	1861
11.158.3.7 shortcut() [2/2]	1861
11.158.3.8 soft()	1861
11.158.3.9 textcolor()	1861
11.158.3.10 textfont() [1/2]	1862
11.158.3.11 textfont() [2/2]	1862
11.158.3.12 textsize() [1/2]	1862
11.158.3.13 textsize() [2/2]	1862
11.159 FI_Value_Output Class Reference	1862

11.159.1 Detailed Description	1870
11.159.2 Constructor & Destructor Documentation	1870
11.159.2.1 FI_Value_Output()	1870
11.159.3 Member Function Documentation	1871
11.159.3.1 draw()	1871
11.159.3.2 handle()	1871
11.159.3.3 soft() [1/2]	1871
11.159.3.4 soft() [2/2]	1872
11.159.3.5 textcolor() [1/2]	1872
11.159.3.6 textcolor() [2/2]	1872
11.159.3.7 textfont() [1/2]	1872
11.159.3.8 textfont() [2/2]	1872
11.159.3.9 textsize()	1872
11.160 FI_Value_Slider Class Reference	1872
11.160.1 Detailed Description	1881
11.160.2 Constructor & Destructor Documentation	1882
11.160.2.1 FI_Value_Slider()	1882
11.160.3 Member Function Documentation	1882
11.160.3.1 draw()	1882
11.160.3.2 handle()	1882
11.160.3.3 value_height() [1/2]	1883
11.160.3.4 value_height() [2/2]	1883
11.160.3.5 value_width() [1/2]	1883
11.160.3.6 value_width() [2/2]	1883
11.161 FI_Widget Class Reference	1884
11.161.1 Detailed Description	1890
11.161.2 Member Enumeration Documentation	1891
11.161.2.1 anonymous enum	1891
11.161.3 Constructor & Destructor Documentation	1892
11.161.3.1 FI_Widget()	1892
11.161.3.2 ~FI_Widget()	1892
11.161.4 Member Function Documentation	1892
11.161.4.1 activate()	1892
11.161.4.2 active()	1892
11.161.4.3 active_r()	1893
11.161.4.4 align() [1/2]	1893
11.161.4.5 align() [2/2]	1893
11.161.4.6 argument() [1/2]	1893
11.161.4.7 argument() [2/2]	1894
11.161.4.8 as_gl_window()	1894
11.161.4.9 as_group()	1894
11.161.4.10 as_window()	1895

11.161.4.11 bind_deimage() [1/2]	1895
11.161.4.12 bind_deimage() [2/2]	1895
11.161.4.13 bind_image() [1/2]	1896
11.161.4.14 bind_image() [2/2]	1897
11.161.4.15 box() [1/2]	1897
11.161.4.16 box() [2/2]	1897
11.161.4.17 callback() [1/6]	1897
11.161.4.18 callback() [2/6]	1898
11.161.4.19 callback() [3/6]	1898
11.161.4.20 callback() [4/6]	1898
11.161.4.21 callback() [5/6]	1898
11.161.4.22 callback() [6/6]	1899
11.161.4.23 changed()	1899
11.161.4.24 clear_active()	1899
11.161.4.25 clear_changed()	1899
11.161.4.26 clear_damage()	1900
11.161.4.27 clear_output()	1900
11.161.4.28 clear_visible()	1900
11.161.4.29 clear_visible_focus()	1900
11.161.4.30 color() [1/3]	1900
11.161.4.31 color() [2/3]	1901
11.161.4.32 color() [3/3]	1901
11.161.4.33 color2() [1/2]	1901
11.161.4.34 color2() [2/2]	1901
11.161.4.35 contains()	1901
11.161.4.36 copy_label()	1902
11.161.4.37 copy_tooltip()	1902
11.161.4.38 damage() [1/3]	1902
11.161.4.39 damage() [2/3]	1903
11.161.4.40 damage() [3/3]	1904
11.161.4.41 deactivate()	1904
11.161.4.42 default_callback()	1904
11.161.4.43 deimage() [1/4]	1905
11.161.4.44 deimage() [2/4]	1905
11.161.4.45 deimage() [3/4]	1905
11.161.4.46 deimage() [4/4]	1905
11.161.4.47 deimage_bound()	1905
11.161.4.48 do_callback() [1/3]	1906
11.161.4.49 do_callback() [2/3]	1906
11.161.4.50 do_callback() [3/3]	1906
11.161.4.51 draw()	1907
11.161.4.52 draw_focus() [1/3]	1907

11.161.4.53 draw_focus()	[2/3]	1907
11.161.4.54 draw_focus()	[3/3]	1908
11.161.4.55 draw_label()	[1/3]	1908
11.161.4.56 draw_label()	[2/3]	1908
11.161.4.57 draw_label()	[3/3]	1909
11.161.4.58 h()	[1/2]	1909
11.161.4.59 h()	[2/2]	1909
11.161.4.60 handle()		1909
11.161.4.61 hide()		1910
11.161.4.62 horizontal_label_margin()	[1/2]	1910
11.161.4.63 horizontal_label_margin()	[2/2]	1910
11.161.4.64 image()	[1/4]	1910
11.161.4.65 image()	[2/4]	1910
11.161.4.66 image()	[3/4]	1910
11.161.4.67 image()	[4/4]	1911
11.161.4.68 image_bound()		1911
11.161.4.69 inside()		1911
11.161.4.70 is_label_copied()		1912
11.161.4.71 label()	[1/3]	1912
11.161.4.72 label()	[2/3]	1912
11.161.4.73 label()	[3/3]	1912
11.161.4.74 label_image_spacing()	[1/2]	1912
11.161.4.75 label_image_spacing()	[2/2]	1913
11.161.4.76 label_shortcut()		1913
11.161.4.77 labelcolor()	[1/2]	1913
11.161.4.78 labelcolor()	[2/2]	1913
11.161.4.79 labelfont()	[1/2]	1914
11.161.4.80 labelfont()	[2/2]	1914
11.161.4.81 labelsizes()	[1/2]	1914
11.161.4.82 labelsizes()	[2/2]	1914
11.161.4.83 labeltype()	[1/2]	1915
11.161.4.84 labeltype()	[2/2]	1915
11.161.4.85 measure_label()		1915
11.161.4.86 needs_keyboard()	[1/2]	1915
11.161.4.87 needs_keyboard()	[2/2]	1916
11.161.4.88 output()		1916
11.161.4.89 parent()	[1/2]	1916
11.161.4.90 parent()	[2/2]	1916
11.161.4.91 position()		1917
11.161.4.92 redraw()		1918
11.161.4.93 redraw_label()		1918
11.161.4.94 resize()		1918

11.161.4.95 selection_color() [1/2]	1918
11.161.4.96 selection_color() [2/2]	1919
11.161.4.97 set_active()	1919
11.161.4.98 set_changed()	1919
11.161.4.99 set_output()	1919
11.161.4.100 set_visible()	1919
11.161.4.101 set_visible_focus()	1919
11.161.4.102 shortcut_label() [1/2]	1920
11.161.4.103 shortcut_label() [2/2]	1920
11.161.4.104 show()	1920
11.161.4.105 size()	1920
11.161.4.106 take_focus()	1921
11.161.4.107 takesevents()	1921
11.161.4.108 test_shortcut() [1/2]	1921
11.161.4.109 test_shortcut() [2/2]	1921
11.161.4.110 tooltip() [1/2]	1922
11.161.4.111 tooltip() [2/2]	1922
11.161.4.112 top_window()	1922
11.161.4.113 top_window_offset()	1922
11.161.4.114 type() [1/2]	1923
11.161.4.115 type() [2/2]	1923
11.161.4.116 user_data()	1923
11.161.4.117 vertical_label_margin() [1/2]	1923
11.161.4.118 vertical_label_margin() [2/2]	1923
11.161.4.119 visible()	1923
11.161.4.120 visible_focus() [1/2]	1924
11.161.4.121 visible_focus() [2/2]	1924
11.161.4.122 visible_r()	1924
11.161.4.123 w() [1/2]	1924
11.161.4.124 w() [2/2]	1925
11.161.4.125 when() [1/2]	1925
11.161.4.126 when() [2/2]	1925
11.161.4.127 window()	1926
11.161.4.128 x() [1/2]	1926
11.161.4.129 x() [2/2]	1926
11.161.4.130 y() [1/2]	1926
11.161.4.131 y() [2/2]	1926
11.162 FI_Widget_Surface Class Reference	1927
11.162.1 Detailed Description	1928
11.162.2 Constructor & Destructor Documentation	1928
11.162.2.1 FI_Widget_Surface()	1928
11.162.3 Member Function Documentation	1928

11.162.3.1 draw()	1928
11.162.3.2 draw_decorated_window()	1929
11.162.3.3 origin() [1/2]	1929
11.162.3.4 origin() [2/2]	1929
11.162.3.5 print_window_part()	1929
11.162.3.6 printable_rect()	1930
11.162.3.7 translate()	1930
11.162.3.8 untranslate()	1930
11.163 FI_Widget_Tracker Class Reference	1930
11.163.1 Detailed Description	1931
11.163.2 Member Function Documentation	1931
11.163.2.1 deleted()	1931
11.163.2.2 exists()	1931
11.163.2.3 widget()	1932
11.164 FI_Window Class Reference	1932
11.164.1 Detailed Description	1944
11.164.2 Constructor & Destructor Documentation	1944
11.164.2.1 FI_Window() [1/2]	1944
11.164.2.2 FI_Window() [2/2]	1944
11.164.2.3 ~FI_Window()	1945
11.164.3 Member Function Documentation	1945
11.164.3.1 allow_expand_outside_parent()	1945
11.164.3.2 as_double_window()	1945
11.164.3.3 as_overlay_window()	1945
11.164.3.4 as_window() [1/2]	1945
11.164.3.5 as_window() [2/2]	1946
11.164.3.6 border()	1946
11.164.3.7 clear_border()	1946
11.164.3.8 clear_modal_states()	1946
11.164.3.9 current()	1947
11.164.3.10 cursor() [1/3]	1947
11.164.3.11 cursor() [2/3]	1947
11.164.3.12 cursor() [3/3]	1947
11.164.3.13 decorated_h()	1947
11.164.3.14 decorated_w()	1948
11.164.3.15 default_cursor() [1/2]	1948
11.164.3.16 default_cursor() [2/2]	1948
11.164.3.17 default_icon()	1948
11.164.3.18 default_icons() [1/2]	1949
11.164.3.19 default_icons() [2/2]	1949
11.164.3.20 default_size_range()	1949
11.164.3.21 default_xclass() [1/2]	1950

11.164.3.22 default_xclass() [2/2]	1950
11.164.3.23 draw()	1951
11.164.3.24 flush()	1951
11.164.3.25 force_position() [1/2]	1951
11.164.3.26 force_position() [2/2]	1951
11.164.3.27 free_icons()	1952
11.164.3.28 free_position()	1952
11.164.3.29 fullscreen()	1952
11.164.3.30 fullscreen_screens()	1952
11.164.3.31 get_size_range()	1952
11.164.3.32 handle()	1953
11.164.3.33 hide()	1953
11.164.3.34 hotspot()	1954
11.164.3.35 icon() [1/3]	1954
11.164.3.36 icon() [2/3]	1954
11.164.3.37 icon() [3/3]	1954
11.164.3.38 iconize()	1955
11.164.3.39 icons() [1/2]	1955
11.164.3.40 icons() [2/2]	1955
11.164.3.41 is_a_rescale()	1955
11.164.3.42 is_resizable()	1956
11.164.3.43 make_current()	1956
11.164.3.44 maximize()	1956
11.164.3.45 modal()	1956
11.164.3.46 os_id()	1957
11.164.3.47 resize()	1957
11.164.3.48 screen_num() [1/2]	1957
11.164.3.49 screen_num() [2/2]	1958
11.164.3.50 set_menu_window()	1958
11.164.3.51 set_modal()	1958
11.164.3.52 set_non_modal()	1958
11.164.3.53 set_tooltip_window()	1959
11.164.3.54 shape() [1/3]	1959
11.164.3.55 shape() [2/3]	1959
11.164.3.56 shape() [3/3]	1959
11.164.3.57 show() [1/2]	1960
11.164.3.58 show() [2/2]	1960
11.164.3.59 show_next_window_iconic() [1/2]	1961
11.164.3.60 show_next_window_iconic() [2/2]	1961
11.164.3.61 shown()	1961
11.164.3.62 size_range()	1961
11.164.3.63 un_maximize()	1962

11.164.3.64 wait_for_expose()	1962
11.164.3.65 xclass() [1/2]	1963
11.164.3.66 xclass() [2/2]	1963
11.164.4 Member Data Documentation	1963
11.164.4.1 current_	1963
11.165 FI_Wizard Class Reference	1964
11.165.1 Detailed Description	1972
11.165.2 Constructor & Destructor Documentation	1972
11.165.2.1 FI_Wizard()	1972
11.165.3 Member Function Documentation	1972
11.165.3.1 draw()	1972
11.165.3.2 next()	1973
11.166 FI_XBM_Image Class Reference	1973
11.166.1 Detailed Description	1975
11.166.2 Constructor & Destructor Documentation	1975
11.166.2.1 FI_XBM_Image()	1975
11.167 FI_XPM_Image Class Reference	1976
11.167.1 Detailed Description	1978
11.167.2 Constructor & Destructor Documentation	1978
11.167.2.1 FI_XPM_Image()	1978
11.168 FI_GIF_Image::GIF_FRAME Struct Reference	1979
11.169 FI_ICO_Image::IconDirEntry Struct Reference	1979
11.169.1 Detailed Description	1980
11.170 FI_Text_Editor::Key_Binding Struct Reference	1980
11.170.1 Detailed Description	1980
11.171 FI_Terminal::Margin Class Reference	1980
11.172 FI_Preferences::Name Class Reference	1980
11.172.1 Detailed Description	1981
11.172.2 Constructor & Destructor Documentation	1981
11.172.2.1 Name() [1/2]	1981
11.172.2.2 Name() [2/2]	1981
11.173 FI_Preferences::Node Class Reference	1981
11.174 FI_Paged_Device::page_format Struct Reference	1982
11.174.1 Detailed Description	1982
11.175 FI_Terminal::PartialUtf8Buf Class Reference	1983
11.176 FI_Terminal::RingBuffer Class Reference	1983
11.177 FI_Preferences::RootNode Class Reference	1984
11.178 FI_Scroll::ScrollInfo Struct Reference	1984
11.178.1 Detailed Description	1985
11.179 FI_Terminal::Selection Class Reference	1985
11.179.1 Member Function Documentation	1985
11.179.1.1 get_selection()	1985

11.180 FI_Tile::Size_Range Struct Reference	1986
11.181 FI_Text_Display::Style_Table_Entry Struct Reference	1986
11.181.1 Detailed Description	1986
11.182 FI_Terminal::Utf8Char Class Reference	1986
12 File Documentation	1989
12.1 Enumerations.H File Reference	1989
12.1.1 Detailed Description	2001
12.1.2 Macro Definition Documentation	2001
12.1.2.1 FL_ABI_VERSION	2001
12.1.2.2 FL_API_VERSION	2001
12.1.2.3 FL_BUTTON	2002
12.1.2.4 FL_IMAGE_LABEL	2002
12.1.2.5 FL_MAJOR_VERSION	2002
12.1.2.6 FL_MINOR_VERSION	2002
12.1.2.7 FL_MULTI_LABEL	2002
12.1.2.8 FL_PATCH_VERSION	2002
12.1.2.9 FL_SYMBOL_LABEL	2002
12.1.2.10 FL_VERSION	2003
12.1.3 Typedef Documentation	2003
12.1.3.1 FI_Contrast_Function	2003
12.1.3.2 FI_Fontsize	2003
12.1.4 Enumeration Type Documentation	2003
12.1.4.1 anonymous enum	2003
12.1.4.2 FI_Arrow_Type	2004
12.1.4.3 FI_Boxtype	2004
12.1.4.4 FI_Callback_Reason	2006
12.1.4.5 FI_Contrast_Mode	2007
12.1.4.6 FI_Cursor	2007
12.1.4.7 FI_Damage	2008
12.1.4.8 FI_Event	2008
12.1.4.9 FI_Labeltype	2011
12.1.4.10 FI_Orientation	2011
12.1.4.11 FI_When	2012
12.1.5 Function Documentation	2013
12.1.5.1 fl_box()	2013
12.1.5.2 fl_color_cube()	2013
12.1.5.3 fl_define_FL_EMBOSSED_LABEL()	2013
12.1.5.4 fl_define_FL_ENGRAVED_LABEL()	2013
12.1.5.5 fl_define_FL_ICON_LABEL()	2013
12.1.5.6 fl_define_FL_IMAGE_LABEL()	2013
12.1.5.7 fl_define_FL_MULTI_LABEL()	2013

12.1.5.8 fl_define_FL_SHADOW_LABEL()	2013
12.1.5.9 fl_down()	2014
12.1.5.10 fl_frame()	2014
12.1.5.11 fl_gray_ramp()	2014
12.1.6 Variable Documentation	2014
12.1.6.1 FL_ALIGN_LEFT	2014
12.1.6.2 FL_ALIGN_TOP	2014
12.1.6.3 FL_NORMAL_SIZE	2014
12.2 Enumerations.H	2014
12.3 filename.H File Reference	2023
12.3.1 Detailed Description	2024
12.4 filename.H	2024
12.5 FI.H File Reference	2026
12.5.1 Detailed Description	2027
12.6 FI.H	2027
12.7 FI_Adjuster.H	2034
12.8 FI_Anim_GIF_Image.H	2034
12.9 fl_ask.H File Reference	2036
12.9.1 Detailed Description	2037
12.9.2 Enumeration Type Documentation	2037
12.9.2.1 FI_Beep	2037
12.9.3 Function Documentation	2038
12.9.3.1 fl_message_position()	2038
12.10 fl_ask.H	2038
12.11 fl_attr.h File Reference	2039
12.11.1 Detailed Description	2040
12.11.2 Macro Definition Documentation	2040
12.11.2.1 __fl_attr	2040
12.11.2.2 FL_DEPRECATED	2040
12.11.2.3 FL_OVERRIDE	2040
12.12 fl_attr.h	2040
12.13 FI_Bitmap.H	2042
12.14 FI_BMP_Image.H	2043
12.15 FI_Box.H File Reference	2044
12.15.1 Detailed Description	2044
12.16 FI_Box.H	2044
12.17 FI_Browser.H	2044
12.18 FI_Browser_.H	2046
12.19 FI_Button.H	2048
12.20 FI_Cairo.H File Reference	2049
12.20.1 Detailed Description	2049
12.21 FI_Cairo.H	2049

12.22 FI_Cairo_Window.H File Reference	2050
12.22.1 Detailed Description	2050
12.23 FI_Cairo_Window.H	2051
12.24 fl_callback_macros.H File Reference	2051
12.24.1 Detailed Description	2052
12.24.2 Macro Definition Documentation	2052
12.24.2.1 FL_FUNCTION_CALLBACK_3	2052
12.24.2.2 FL_INLINE_CALLBACK_2	2052
12.24.2.3 FL_METHOD_CALLBACK_1	2053
12.25 fl_callback_macros.H	2054
12.26 fl_casts.H	2060
12.27 FI_Chart.H File Reference	2060
12.27.1 Detailed Description	2061
12.28 FI_Chart.H	2061
12.29 FI_Check_Browser.H	2062
12.30 FI_Check_Button.H	2063
12.31 FI_Choice.H	2064
12.32 FI_Clock.H	2064
12.33 FI_Color_Chooser.H File Reference	2065
12.33.1 Detailed Description	2066
12.34 FI_Color_Chooser.H	2066
12.35 FI_Copy_Surface.H	2067
12.36 FI_Counter.H	2068
12.37 FI_Device.H File Reference	2069
12.37.1 Detailed Description	2069
12.38 FI_Device.H	2069
12.39 FI_Dial.H	2070
12.40 FI_Double_Window.H	2071
12.41 fl_draw.H File Reference	2071
12.41.1 Detailed Description	2077
12.42 fl_draw.H	2077
12.43 FI_Export.H	2082
12.44 FI_File_Browser.H	2083
12.45 FI_File_Chooser.H	2084
12.46 FI_File_Icon.H	2086
12.47 FI_File_Input.H	2088
12.48 FI_Fill_Dial.H	2088
12.49 FI_Fill_Slider.H	2089
12.50 FI_Flex.H	2089
12.51 FI_Float_Input.H	2091
12.52 FI_FormsBitmap.H	2091
12.53 FI_FormsPixmap.H	2092

12.54 FI_Free.H	2092
12.55 FI_GIF_Image.H	2093
12.56 FI_Gl_Window.H	2094
12.57 FI_Graphics_Driver.H	2095
12.58 FI_Grid.H File Reference	2100
12.58.1 Detailed Description	2101
12.59 FI_Grid.H	2101
12.60 FI_Group.H File Reference	2104
12.60.1 Detailed Description	2104
12.61 FI_Group.H	2104
12.62 FI_Help_Dialog.H	2106
12.63 FI_Help_View.H	2107
12.64 FI_Hold_Browser.H	2110
12.65 FI_Hor_Fill_Slider.H	2110
12.66 FI_Hor_Nice_Slider.H	2111
12.67 FI_Hor_Slider.H	2111
12.68 FI_Hor_Value_Slider.H	2112
12.69 FI_ICO_Image.H	2112
12.70 FI_Image.H File Reference	2113
12.70.1 Detailed Description	2113
12.70.2 Enumeration Type Documentation	2113
12.70.2.1 FI_RGB_Scaling	2113
12.71 FI_Image.H	2113
12.72 FI_Image_Surface.H	2115
12.73 FI_Input.H	2116
12.74 FI_Input_.H	2117
12.75 FI_Input_Choice.H	2120
12.76 FI_Int_Input.H	2122
12.77 FI_JPEG_Image.H	2122
12.78 FI_Light_Button.H	2123
12.79 FI_Line_Dial.H	2123
12.80 FI_Menu.H	2124
12.81 FI_Menu_.H	2124
12.82 FI_Menu_Bar.H	2125
12.83 FI_Menu_Button.H	2126
12.84 FI_Menu_Item.H File Reference	2126
12.84.1 Enumeration Type Documentation	2127
12.84.1.1 anonymous enum	2127
12.85 FI_Menu_Item.H	2127
12.86 FI_Menu_Window.H	2130
12.87 fl_message.H	2130
12.88 FI_Multi_Browser.H	2131

12.89 FI_Multi_Label.H	2131
12.90 FI_Multiline_Input.H	2132
12.91 FI_Multiline_Output.H	2132
12.92 FI_Native_File_Chooser.H File Reference	2132
12.92.1 Detailed Description	2133
12.93 FI_Native_File_Chooser.H	2133
12.94 FI_Nice_Slider.H	2135
12.95 FI_Object.H	2135
12.96 FI_Output.H	2136
12.97 FI_Overlay_Window.H	2136
12.98 FI_Pack.H	2137
12.99 FI_Paged_Device.H File Reference	2137
12.99.1 Detailed Description	2138
12.100 FI_Paged_Device.H	2138
12.101 FI_PDF_File_Surface.H	2139
12.102 FI_Pixmap.H	2140
12.103 FI_Plugin.H	2141
12.104 FI_PNG_Image.H	2141
12.105 FI_PNM_Image.H	2142
12.106 FI_Positioner.H	2142
12.107 FI_PostScript.H File Reference	2143
12.107.1 Detailed Description	2143
12.107.2 Typedef Documentation	2143
12.107.2.1 FI_PostScript_Close_Command	2143
12.108 FI_PostScript.H	2144
12.109 FI_Preferences.H	2145
12.110 FI_Printer.H File Reference	2148
12.110.1 Detailed Description	2148
12.111 FI_Printer.H	2148
12.112 FI_Progress.H	2149
12.113 FI_Radio_Button.H	2149
12.114 FI_Radio_Light_Button.H	2150
12.115 FI_Radio_Round_Button.H	2150
12.116 FI_Rect.H	2151
12.117 FI_Repeat_Button.H	2152
12.118 FI_Return_Button.H	2152
12.119 FI_RGB_Image.H	2153
12.120 FI_Roller.H	2153
12.121 FI_Round_Button.H	2154
12.122 FI_Round_Clock.H	2154
12.123 FI_Scheme.H	2154
12.124 FI_Scheme_Choice.H	2155

12.125 FI_Scroll.H	2156
12.126 FI_Scrollbar.H	2157
12.127 FI_Secret_Input.H	2158
12.128 FI_Select_Browser.H	2158
12.129 FI_Shared_Image.H File Reference	2158
12.129.1 Detailed Description	2159
12.129.2 Typedef Documentation	2159
12.129.2.1 FI_Shared_Handler	2159
12.129.3 Function Documentation	2159
12.129.3.1 fl_register_images()	2159
12.130 FI_Shared_Image.H	2160
12.131 FI_Shortcut_Button.H	2161
12.132 fl_show_colormap.H File Reference	2162
12.132.1 Detailed Description	2162
12.133 fl_show_colormap.H	2162
12.134 fl_show_input.H	2162
12.135 FI_Simple_Counter.H	2162
12.136 FI_Single_Window.H	2163
12.137 FI_Slider.H	2163
12.138 FI_Spinner.H	2164
12.139 fl_string_functions.h File Reference	2166
12.139.1 Detailed Description	2166
12.140 fl_string_functions.h	2166
12.141 FI_SVG_File_Surface.H	2167
12.142 FI_SVG_Image.H	2167
12.143 FI_Sys_Menu_Bar.H File Reference	2168
12.143.1 Detailed Description	2168
12.144 FI_Sys_Menu_Bar.H	2168
12.145 FI_Table.H	2169
12.146 FI_Table_Row.H	2175
12.147 FI_Tabs.H	2177
12.148 FI_Terminal.H File Reference	2178
12.148.1 Detailed Description	2178
12.149 FI_Terminal.H	2178
12.150 FI_Text_Buffer.H	2188
12.151 FI_Text_Display.H	2192
12.152 FI_Text_Editor.H	2197
12.153 FI_Tile.H	2198
12.154 FI_Tiled_Image.H	2199
12.155 FI_Timer.H	2200
12.156 FI_Toggle_Button.H	2200
12.157 FI_Toggle_Light_Button.H	2201

12.158 FI_Toggle_Round_Button.H	2201
12.159 FI_Tooltip.H	2201
12.160 FI_Tree.H File Reference	2202
12.160.1 Detailed Description	2203
12.160.2 Enumeration Type Documentation	2203
12.160.2.1 FI_Tree_Reason	2203
12.161 FI_Tree.H	2203
12.162 FI_Tree_Item.H File Reference	2206
12.162.1 Detailed Description	2206
12.163 FI_Tree_Item.H	2206
12.164 FI_Tree_Item_Array.H File Reference	2210
12.164.1 Detailed Description	2210
12.165 FI_Tree_Item_Array.H	2210
12.166 FI_Tree_Prefs.H File Reference	2211
12.166.1 Detailed Description	2212
12.166.2 Enumeration Type Documentation	2212
12.166.2.1 FI_Tree_Connector	2212
12.166.2.2 FI_Tree_Item_Draw_Mode	2212
12.166.2.3 FI_Tree_Item_Reselect_Mode	2212
12.166.2.4 FI_Tree_Select	2213
12.166.2.5 FI_Tree_Sort	2213
12.167 FI_Tree_Prefs.H	2213
12.168 fl_types.h File Reference	2217
12.168.1 Detailed Description	2217
12.168.2 Typedef Documentation	2217
12.168.2.1 FI_Shortcut	2217
12.169 fl_types.h	2217
12.170 fl_utf8.h File Reference	2218
12.170.1 Detailed Description	2220
12.171 fl_utf8.h	2220
12.172 FI_Valuator.H	2222
12.173 FI_Value_Input.H	2223
12.174 FI_Value_Output.H	2224
12.175 FI_Value_Slider.H	2225
12.176 FI_Widget.H File Reference	2225
12.176.1 Detailed Description	2226
12.176.2 Macro Definition Documentation	2226
12.176.2.1 FL_RESERVED_TYPE	2226
12.177 FI_Widget.H	2226
12.178 FI_Widget_Surface.H	2231
12.179 FI_Window.H File Reference	2232
12.179.1 Detailed Description	2232

12.180	Fl_Window.H	2232
12.181	Fl_Wizard.H	2235
12.182	Fl_XBM_Image.H	2236
12.183	Fl_XPM_Image.H	2236
12.184	forms.H	2237
12.185	gl.h File Reference	2246
12.185.1	Detailed Description	2247
12.185.2	Function Documentation	2247
12.185.2.1	gl_color()	2247
12.185.2.2	gl_draw() [1/7]	2248
12.185.2.3	gl_draw() [2/7]	2248
12.185.2.4	gl_draw() [3/7]	2248
12.185.2.5	gl_draw() [4/7]	2248
12.185.2.6	gl_draw() [5/7]	2248
12.185.2.7	gl_draw() [6/7]	2249
12.185.2.8	gl_draw() [7/7]	2249
12.185.2.9	gl_font()	2249
12.185.2.10	gl_rect()	2249
12.185.2.11	gl_rectf()	2249
12.185.2.12	gl_texture_pile_height() [1/2]	2250
12.185.2.13	gl_texture_pile_height() [2/2]	2250
12.186	gl.h	2250
12.187	gl2opengl.h	2251
12.188	gl_draw.H	2252
12.189	glu.h	2252
12.190	glut.H	2252
12.191	mac.H File Reference	2259
12.191.1	Detailed Description	2259
12.192	mac.H	2259
12.193	math.h	2261
12.194	names.h File Reference	2262
12.194.1	Detailed Description	2262
12.195	names.h	2262
12.196	platform.H	2263
12.197	platform_types.h File Reference	2264
12.197.1	Detailed Description	2265
12.197.2	Typedef Documentation	2265
12.197.2.1	fl_intptr_t	2265
12.197.2.2	Fl_Offscreen	2265
12.197.2.3	Fl_Region	2265
12.197.2.4	Fl_Timestamp	2266
12.197.2.5	fl_uintptr_t	2266

12.197.2.6 GLContext	2266
12.198 platform_types.h	2266
12.199 wayland.H File Reference	2268
12.199.1 Detailed Description	2268
12.199.2 Function Documentation	2268
12.199.2.1 fl_wl_compositor()	2268
12.200 wayland.H	2268
12.201 win32.H File Reference	2269
12.201.1 Detailed Description	2269
12.202 win32.H	2269
12.203 x.H	2270
12.204 x11.H File Reference	2271
12.204.1 Detailed Description	2271
12.204.2 Function Documentation	2271
12.204.2.1 fl_x11_find()	2271
12.204.2.2 fl_x11_gc()	2271
12.204.2.3 fl_x11_use_display()	2271
12.204.2.4 fl_x11_xid()	2272
12.205 x11.H	2272
12.206 cgdebug.h	2273
12.207 fastarrow.h	2275
12.208 Fl.cxx File Reference	2276
12.208.1 Detailed Description	2277
12.208.2 Function Documentation	2277
12.208.2.1 fl_close_display()	2277
12.208.2.2 fl_find()	2277
12.208.2.3 fl_open_display()	2277
12.208.3 Variable Documentation	2277
12.208.3.1 fl_disable_wayland	2277
12.209 fl_arc.cxx File Reference	2278
12.209.1 Detailed Description	2278
12.210 fl_ask.cxx File Reference	2278
12.210.1 Detailed Description	2279
12.211 fl_boxtype.cxx File Reference	2280
12.211.1 Detailed Description	2281
12.211.2 Function Documentation	2281
12.211.2.1 fl_internal_boxtype()	2281
12.211.2.2 fl_rectbound()	2281
12.212 fl_cmap.h	2281
12.213 fl_color.cxx File Reference	2285
12.213.1 Detailed Description	2285
12.214 Fl_compose.cxx File Reference	2285

12.214.1 Detailed Description	2285
12.215 fl_contrast.cxx File Reference	2285
12.215.1 Detailed Description	2286
12.216 fl_curve.cxx File Reference	2286
12.216.1 Detailed Description	2286
12.217 FI_Double_Window.cxx File Reference	2286
12.217.1 Detailed Description	2286
12.218 FI_get_system_colors.cxx File Reference	2286
12.218.1 Detailed Description	2287
12.218.2 Function Documentation	2288
12.218.2.1 fl_parse_color()	2288
12.219 FI_Gl_Choice.H	2288
12.220 FI_Gl_Window_Driver.H	2289
12.221 FI_Graphics_Driver.cxx File Reference	2290
12.221.1 Detailed Description	2290
12.222 FI_Grid.cxx File Reference	2290
12.222.1 Detailed Description	2290
12.223 FI_Image_Reader.h	2291
12.224 FI_Int_Vector.H	2292
12.225 FI_Message.h	2293
12.226 FI_Native_File_Chooser_Kdialog.H	2295
12.227 FI_Native_File_Chooser_Zenity.H	2295
12.228 fl_oxy.h	2296
12.229 FI_Paged_Device.cxx File Reference	2296
12.229.1 Detailed Description	2296
12.230 fl_rect.cxx File Reference	2296
12.230.1 Detailed Description	2296
12.231 FI_Screen_Driver.H	2297
12.232 FI_String.H	2299
12.233 FI_Sys_Menu_Bar_Driver.H	2301
12.234 FI_System_Driver.H	2302
12.235 FI_Timeout.cxx File Reference	2304
12.236 FI_Timeout.h File Reference	2304
12.236.1 Detailed Description	2305
12.237 FI_Timeout.h	2305
12.238 fl_vertex.cxx File Reference	2306
12.238.1 Detailed Description	2306
12.239 FI_Window_Driver.H	2306
12.240 fl_write_png.cxx File Reference	2309
12.240.1 Detailed Description	2309
12.240.2 Function Documentation	2309
12.240.2.1 fl_write_png() [1/3]	2309

12.240.2.2 fl_write_png() [2/3]	2310
12.240.2.3 fl_write_png() [3/3]	2310
12.241 Fl_XColor.H	2311
12.242 flstring.h	2311
12.243 freeglut_teapot_data.h	2312
12.244 mediumarrow.h	2314
12.245 numeric-sort.c File Reference	2314
12.245.1 Function Documentation	2314
12.245.1.1 fl_casnumeric-sort()	2314
12.245.1.2 fl_numeric-sort()	2315
12.246 print_button.h	2316
12.247 print_panel.h	2316
12.248 slowarrow.h	2317
12.249 utf8_internal.h	2317
12.250 vsnprintf.c File Reference	2317
12.250.1 Detailed Description	2318
12.250.2 Function Documentation	2318
12.250.2.1 fl_vsnprintf()	2318
12.251 Xutf8.h	2318
12.252 case.h	2320
12.253 dingbats_.h	2341
12.254 spacing.h	2347
12.255 symbol_.h	2370
12.256 imKStoUCS.c	2383
12.257 armSCII_8.h	2387
12.258 ASCII.h	2388
12.259 big5.h	2388
12.260 big5_emacs.h	2436
12.261 cp1133.h	2438
12.262 cp1251.h	2439
12.263 cp1255.h	2440
12.264 cp1256.h	2442
12.265 cp936ext.h	2443
12.266 gb2312.h	2515
12.267 georgian_academy.h	2545
12.268 georgian_ps.h	2546
12.269 iso8859_1.h	2547
12.270 iso8859_10.h	2547
12.271 iso8859_11.h	2548
12.272 iso8859_13.h	2549
12.273 iso8859_14.h	2551
12.274 iso8859_15.h	2552

12.275 iso8859_16.h	2553
12.276 iso8859_2.h	2554
12.277 iso8859_3.h	2555
12.278 iso8859_4.h	2557
12.279 iso8859_5.h	2558
12.280 iso8859_6.h	2559
12.281 iso8859_7.h	2560
12.282 iso8859_8.h	2561
12.283 iso8859_9.h	2562
12.284 iso8859_9e.h	2563
12.285 jisx0201.h	2564
12.286 jisx0208.h	2565
12.287 jisx0212.h	2592
12.288 koi8_c.h	2617
12.289 koi8_r.h	2619
12.290 koi8_u.h	2620
12.291 ksc5601.h	2622
12.292 mulelao.h	2657
12.293 tatar_cyr.h	2658
12.294 tcvn.h	2659
12.295 tis620.h	2661
12.296 ucs2be.h	2662
12.297 utf8.h	2662
12.298 viscii.h	2664
12.299 mk_wcwidth.c	2665
12.300 ucs2fontmap.c	2669
12.301 utf8Utils.c	2673
12.302 Ximint.h	2676
12.303 Xlibint.h	2676

Index**2677**

Chapter 1

FLTK Programming Manual



FLTK 1.5.0 Programming Manual

By F. Costantini, M. Melcher, A. Schlosser, B. Spitzak and M. Sweet.

Copyright © 1998 - 2025 by Bill Spitzak and others.

This software and manual are provided under the terms of the GNU Library General Public License. Permission is granted to reproduce this manual or any portion for any purpose, provided this copyright and permission notice are preserved.

Preface Introduction to FLTK FLTK Basics Common Widgets and Attributes • Colors • Box Types • Labels and Label Types • Drawing Images Coordinates and Layout Widgets How Does Resizing Work? Designing a Simple Text Editor Drawing Things in FLTK • When Can You Draw Things in FLTK? • What Units Do FLTK Functions Use? • Drawing Functions • Drawing Images • Offscreen Drawing Handling Events • Fl::event_*() methods • Event Propagation	Adding and Extending Widgets Using OpenGL FLTK Runtime Options Advanced FLTK Unicode and UTF-8 Support Appendices: • Constants and Enumerations • GLUT Compatibility – class Fl_Glut_Window • Forms Compatibility • Operating System Issues • Migrating Code from FLTK 1.3 to 1.4 • Software License • Example Source Code • FAQ (Frequently Asked Questions)

1.1 Preface

This manual describes the Fast Light Tool Kit ("FLTK") version 1.5.0, a C++ Graphical User Interface ("GUI") toolkit for UNIX, Microsoft Windows, and Apple macOS.

Version 1.4.0 introduced support for a new windowing system under Linux/Unix: Wayland. FLTK applications under Linux/Unix run unchanged as Wayland or X11 clients depending on availability at run-time.

Each of the chapters in this manual is designed as a tutorial for using FLTK, while the appendices provide a convenient reference for all FLTK widgets, functions, and operating system interfaces.

This manual may be printed, modified, and/or used under the terms of the FLTK license provided in ↵ : [Software License](#).

1.1.1 Organization

This manual is organized into the following chapters and appendices:

- [Introduction to FLTK](#)
- [FLTK Basics](#)
- [Common Widgets and Attributes](#)

- [Designing a Simple Text Editor](#)
- [Drawing Things in FLTK](#)
- [Handling Events](#)
- [Adding and Extending Widgets](#)
- [Using OpenGL](#)
- [FLTK Runtime Options](#)
- [Advanced FLTK](#)
- [Unicode and UTF-8 Support](#)
- [Constants and Enumerations](#)
- [GLUT Compatibility](#)
- [Forms Compatibility](#)
- [Operating System Issues](#)
- [Migrating Code from FLTK 1.3 to 1.4](#)
- [Developer Information](#)
- [Software License](#)
- [Example Source Code](#)

1.1.2 Conventions

This manual was generated using Doxygen (see <https://www.doxygen.org/>) to process the source code itself, special comments in the code, and additional documentation files. In general, Doxygen recognizes and denotes the following entities as shown:

- classes, such as [Fl_Widget](#),
- methods, such as [Fl_Widget::callback\(Fl_Callback* cb, void* p\)](#),
- functions, such as [fl_draw\(const char *str, int x, int y\)](#),
- internal links, such as [Conventions](#),
- external links, such as <https://www.fltk.org/>.

Other code samples and commands are shown in `regular courier type`.

1.1.3 Abbreviations

The following abbreviations are used in this manual:

X11

The X Window System version 11.

Xlib

The X Window System interface library.

Windows, `WIN32`

The Microsoft Windows Application Programmer's Interface for Windows 2000, Windows XP, Windows Vista, Windows 7 and later Windows versions. FLTK uses the preprocessor definition `_WIN32` for the 32 bit and 64 bit Windows API.

macOS (aka Mac OS X), `APPLE`

The Apple desktop operating system macOS 10.0 and later. MacOS 8 and 9 support was dropped after FLTK 1.0.10. FLTK uses the preprocessor definition `__APPLE__` for macOS.

1.1.4 Copyrights and Trademarks

FLTK is Copyright © 1998 - 2025 by Bill Spitzak and others.

Use and distribution of FLTK is governed by the GNU Library General Public License with 4 exceptions, located in [Software License](#).

UNIX is a registered trademark of the X Open Group, Inc. Microsoft and Windows are registered trademarks of Microsoft Corporation. OpenGL is a registered trademark of Silicon Graphics, Inc. Apple, Macintosh, MacOS, macOS, and Mac OS X are registered trademarks of Apple Computer, Inc.

1.2 Introduction to FLTK

The Fast Light Tool Kit ("FLTK") is a cross-platform C++ GUI toolkit for UNIX®/Linux® (X11 and Wayland), Microsoft® Windows®, and Apple® macOS®. FLTK provides modern GUI functionality without bloat and supports 3D graphics via OpenGL® and its built-in GLUT emulation. It was originally developed by Mr. Bill Spitzak and is currently maintained by a small group of developers across the world with a central repository on GitHub.

1.2.1 History of FLTK

It has always been Bill's belief that the GUI API of all modern systems is much too high level. Toolkits (even FLTK) are *not* what should be provided and documented as part of an operating system. The system only has to provide arbitrary shaped but featureless windows, a powerful set of graphics drawing calls, and a simple *unalterable* method of delivering events to the owners of the windows. NeXT (if you ignored NextStep) provided this, but they chose to hide it and tried to push their own baroque toolkit instead.

Many of the ideas in FLTK were developed on a NeXT (but *not* using NextStep) in 1987 in a C toolkit Bill called "views". Here he came up with passing events downward in the tree and having the handle routine return a value indicating whether it used the event, and the table-driven menus. In general he was trying to prove that complex UI ideas could be entirely implemented in a user space toolkit, with no knowledge or support by the system.

After going to film school for a few years, Bill worked at Sun Microsystems on the (doomed) NeWS project. Here he found an even better and cleaner windowing system, and he reimplemented "views" atop that. NeWS did have an unnecessarily complex method of delivering events which hurt it. But the designers did admit that perhaps the user could write just as good of a button as they could, and officially exposed the lower level interface.

With the death of NeWS Bill realized that he would have to live with X. The biggest problem with X is the "window manager", which means that the toolkit can no longer control the window borders or drag the window around.

At Digital Domain Bill discovered another toolkit, "Forms". Forms was similar to his work, but provided many more widgets, since it was used in many real applications, rather than as theoretical work. He decided to use Forms, except he integrated his table-driven menus into it. Several very large programs were created using this version of Forms.

The need to switch to OpenGL and GLX, portability, and a desire to use C++ subclassing required a rewrite of Forms. This produced the first version of FLTK. The conversion to C++ required so many changes it made it impossible to recompile any Forms objects. Since it was incompatible anyway, Bill decided to incorporate his older ideas as much as possible by simplifying the lower level interface and the event passing mechanism.

Bill received permission to release it for free on the Internet, with the GNU general public license. Response from Internet users indicated that the Linux market dwarfed the SGI and high-speed GL market, so he rewrote it to use X for all drawing, greatly speeding it up on these machines.

Digital Domain has since withdrawn support for FLTK. While Bill is no longer able to actively develop it, he still contributes to FLTK in his free time and is a part of the FLTK development team.

FLTK was later ported to Windows and macOS. FLTK 1.4 added a "driver based" system of virtual device drivers that enabled porting to Wayland as well. Drawing features include Windows GDI+, Cairo (Wayland and X11), and improved text layout with Pango.

There have been experiments using this driver system to build FLTK based on SDL2, Android, and other graphics systems based solely on simple pixel drawing, but this experimental code is not included in FLTK 1.4. There are thoughts to enable more platforms in later FLTK versions.

FLTK 1.5 removed autotools (configure + Makefile) support in favor of exclusive use of CMake to generate the build files. FLTK 1.5 allows to use a basic set of C++ features from the C++11 standard.

1.2.2 Features

FLTK was designed to be statically linked. This was done by splitting it into many small objects and designing it so that functions that are not used do not have pointers to them in the parts that are used, and thus do not get linked in. This allows you to make an easy-to-install program or to modify FLTK to the exact requirements of your application without worrying about bloat. FLTK works fine as a shared library, though, and is included with several Linux distributions.

Here are some of the core features unique to FLTK:

Note: sizes given below are mostly from 32-bit systems and FLTK 1.1 or earlier, this list needs updates for current FLTK 1.5 (64-bit).

- `sizeof(Fl_Widget) == 64` to 92 (120 in FLTK 1.4 on 64-bit Linux).
- The "core" (the "hello" program compiled & linked with a static FLTK library using gcc on a 486 and then stripped) is 114K. (FLTK 1.4 on 64-bit Linux: 1.1 MB).
- The FLUID program (which includes every widget) is 538k. (FLTK 1.4 with more widgets on 64-bit Linux: 2.3 MB and 2.0 MB on 32-bit Windows).
- Written directly atop core libraries (Xlib, Wayland, Windows or Cocoa) for maximum speed and carefully optimized for code size and performance.
- Precise low-level compatibility between the X11/Wayland, Windows, and macOS versions - only about 10% of the code is different.
- Interactive user interface builder program FLUID. Its output is human-readable and editable C++ source code.
- Support for overlay hardware, with emulation if none is available. Note: hardware support has been removed in 1.4.0.
- Very small & fast portable 2-D drawing library to hide Xlib, Cairo, Windows, or macOS Quartz. Since FLTK 1.4 Wayland graphics use Cairo for drawing. This is optionally available for X11 drawing as well.
- OpenGL/Mesa drawing area widget.
- Support for OpenGL overlay hardware on both X11 and Windows, with emulation if none is available.
- Text widgets with cut & paste, undo, and support for Unicode text and international input methods.
- Compatibility header file for the GLUT library.
- Compatibility header file for the XForms library.

1.2.3 Licensing

FLTK comes with complete free source code. FLTK is available under the terms of the [GNU Library General Public License](https://www.gnu.org/licenses/old-licenses/lgpl-2.1.html) with exceptions that allow for static linking. Contrary to popular belief, it can be used in commercial software - even Bill Gates could use it!

1.2.4 What Does "FLTK" Mean?

FLTK was originally designed to be compatible with the Forms Library written for SGI machines. In that library all the functions and structures started with "fl_". This naming was extended to all new methods and widgets in the C++ library, and this prefix was taken as the name of the library. It is almost impossible to search for "FL" on the Internet, due to the fact that it is also the abbreviation for Florida. After much debating and searching for a new name for the toolkit, which was already in use by several people, Bill came up with "FLTK", including a bogus excuse that it stands for "The Fast Light Toolkit".

1.2.5 FLUID

FLTK comes bundled with FLUID. FLUID, short for Fast Light User Interface Designer, is a graphical editor capable of generating C++ source code and header files ready for compilation. These files ultimately create the graphical user interface for an application.

The FLUID User Handbook is available at <https://www.fltk.org/documentation.php>. It can also be compiled from the FLTK source repository using the `fluid_docs` target in the CMake build environment.

1.2.6 Building and Installing FLTK with CMake

Starting with version 1.4, the **recommended** FLTK build system was CMake. Since FLTK 1.5.0 CMake is the **only** available build system for FLTK.

CMake is a "Build System Generator" that can generate build environments for usage with Ninja, Make, and many more, for instance several IDE's. See file README.CMake.txt of the FLTK source tree for more information.

By default CMake will look for a C++ compiler named `CC`, `c++`, `g++`, or `gcc`. To use another compiler you can set the `CXX` and `CC` environment variables for the C++ and C compiler, respectively. The code to use depends on your shell, for instance:

```
CXX=clang++; export CXX
setenv CXX "clang++"

CC=clang; export CC
setenv CC "clang"
```

Then execute CMake like

```
cd /path/to/fltk
cmake . -B build [ -G <GENERATOR> ] [ options ... ]
```

Note that you can also specify the compiler and build tools on the CMake command line or in the CMake GUI.

When CMake is done you can just run CMake again to build the FLTK library, fluid, fltk-options (setup tool), and all test programs, depending on the configuration.

```
cmake --build build
```

To install the library, become root and type "cmake --install build". This will copy the "fluid" executable to "bindir", the header files to "includedir", and the library files to "libdir".

1.2.7 Building FLTK Under Microsoft Windows

NOTE: This documentation section is currently under review. More up-to-date information for this release may be available in the files "README.Windows.txt" and "README.CMake.txt" and you should read these files to determine if there are changes that may be applicable to your build environment.

FLTK 1.5 is **officially supported** on Windows 10 (11) and later. We don't intend to remove the ability to build and run FLTK 1.5 programs on older Windows versions but the FLTK Team can't build and test on Windows systems that are no longer supported by Microsoft. Therefore, such older systems (including Windows 10, starting in Oct. 2025, when Microsoft does no longer support Windows 10) may work with FLTK or not.

Note

FLTK 1.4 is known to work on Windows (2000,) 2003, XP, and later. The main prerequisite is that the OS version supports UTF-8. FLTK 1.4 is also known to work on recent versions of Windows such as Windows 7, Windows 8/8.1, Windows 10, and Windows 11, and has been reported to work in both 32-bit and 64-bit Windows versions. If you need to use FLTK on older versions than Windows 11 (or Windows 10) you may use FLTK 1.4.x.

Note

Libraries built by any one of the following build environments can not be mixed with object files from any of the other environments because they use incompatible C++ conventions internally.

FLTK currently supports the following development environments on the Windows platform:

1.2.7.1 Free and Commercial Microsoft Visual Studio Versions

Visual Studio 2015 Community or later versions use workspace and project files generated by CMake. Older versions and the commercial versions can be used as well, if they can open the project files generated by CMake. However, FLTK 1.5 and later requires newer C++ features (C++11 or higher) that may not be available in older versions of Visual Studio.

FLTK support of Visual Studio is limited to CMake's support for these Visual Studio versions. Be sure to get your service packs!

Since FLTK 1.5 the project files **MUST** be generated with CMake. Please read "README.txt", "README.CMake.txt", and "README.Windows.txt" for more information about this.

1.2.7.2 Using the Visual C++ DLL Library

The Visual Studio project files can be used to build a DLL version of the FLTK library if CMake option 'FLTK_BUILD_SHARED_LIBS=ON' is set. Because of name mangling differences between PC compilers (even between different versions of Visual Studio) you can only use the DLL that is generated with the same compiler version that you built it with.

When compiling an application or DLL that uses the FLTK DLL with Visual Studio, you need to define the `FL_DLL` preprocessor symbol to get the correct linkage commands embedded within the FLTK header files.

New since FLTK 1.4.0:

If you build your application project with CMake and use the CMake target 'fltk::fltk-shared' to link your application, then 'FL_DLL' is defined automatically for you (by CMake Compile Definitions). If you use your own (hand-made) Visual Studio project you still need to define FL_DLL to compile all source files that use FLTK headers.

1.2.7.3 GNU toolsets (Cygwin or MinGW) hosted on Windows

If using Cygwin with the Cygwin shell, or MinGW with the Msys shell, these build environments behave very much like a Unix or macOS build and the notes above can be applied. Build files are generated by CMake for the platform you are using. See README.CMake.txt for more info.

In general for a build using these tools, e.g. for the Msys shell with MinGW, it should suffice to "cd" into the directory where you have extracted the FLTK tarball and type:

```
cmake . -B build
cmake --build build
```

This will build the FLTK libraries and they can then be utilised directly from the build location. NOTE: this may be simpler than "installing" them in many cases as different tool chains on Windows have different ideas about where the files should be "installed" to.

For example, if you "install" the libraries using Msys/MinGW with the following command

```
cmake --install build
```

then Msys will "install" the libraries to where it thinks the path "/usr/local/" leads to. If you only ever build code from within the Msys environment this works well, but the actual "Windows path" these files are located in will be something like "C:\msys\1.0\local\lib", depending on where your Msys installation is rooted, which may not be useful to other tools.

If you want to install your built FLTK libraries in a non-standard location you may do:

```
cmake . -B build -D CMAKE_INSTALL_PATH="C:/FLTK"
cmake --build build
cmake --install build
```

Where the value passed to "CMAKE_INSTALL_PATH" is the path at which you would like FLTK to be installed. Depending on the build environment using standard Windows path syntax may work, or you need to use MinGW (or Cygwin) syntax.

1.2.8 Internet Resources

FLTK is available on the internet in a bunch of locations:

FLTK Source Repository on GitHub

<https://github.com/fltk/fltk>

WWW (fltk.org)

<https://www.fltk.org/>
<https://www.fltk.org/bugs.php> [for reporting bugs]
<https://www.fltk.org/software.php> [download source code]
<https://www.fltk.org/newsgroups.php> [newsgroup/forums]

User Forums and NNTP Newsgroups

<https://groups.google.com/forum/#!forum/fltkgeneral> [Google Groups interface]
<news://fltk.org:1024/> [NNTP interface]
<https://www.fltk.org/newsgroups.php> [web interface]

GitHub

<https://github.com/fltk/fltk> <https://github.com/fltk/fltk/discussions>
 [FLTK Discussions on GitHub] <https://github.com/fltk/fltk/discussions/categories/q-a>
 [Q&A: Questions and Answers] <https://github.com/fltk/fltk/releases> [Releases, i.e.
 Source Code and Documentation]

1.2.9 Reporting Bugs

To report a bug in FLTK, or for feature requests, please use <https://www.fltk.org/bugs.php> for information about where and how to post bugs, feature requests, or ask for help on using FLTK.

For general support and questions, please use the fltk.general newsgroup (see above, "NNTP Newsgroups") or the web interface to the newsgroups at <https://www.fltk.org/newsgroups.php>, or GitHub Discussions (see link above).

1.3 FLTK Basics

This chapter teaches you the basics of writing and compiling programs that use FLTK.

1.3.1 Writing Your First FLTK Program

Up to FLTK 1.3.x all FLTK programs were required to include the file `<FL/Fl.H>` as the first FLTK header file.

Since FLTK 1.4.0 this requirement was relaxed and `<FL/Fl.H>` needs only be included if the class `Fl` is used or if some other stuff like enumerations is used in the source code. Example code in this documentation may still include it "everywhere" even if it is no longer strictly required.

In addition the program must include a header file for each FLTK class it uses. Listing 1 shows a simple "Hello, World!" program that uses FLTK to display the window.

Listing 1 - "hello.cxx"

```
#include <FL/Fl.H>
#include <FL/Fl_Window.H>
#include <FL/Fl_Box.H>

int main(int argc, char **argv) {
    Fl_Window *window = new Fl_Window(340, 180);
    Fl_Box *box = new Fl_Box(20, 40, 300, 100, "Hello, World!");
    box->box(FL_UP_BOX);
    box->labelfont(FL_BOLD + FL_ITALIC);
    box->labelsize(36);
    box->labeltype(FL_SHADOW_LABEL);
    window->end();
    window->show(argc, argv);
    return Fl::run();
}
```

After including the required header files, the program then creates a window. All following widgets will automatically be children of this window.

```
Fl_Window *window = new Fl_Window(340, 180);
```

Then we create a box with the "Hello, World!" string in it. FLTK automatically adds the new box to `window`, the current grouping widget.

```
Fl_Box *box = new Fl_Box(20, 40, 300, 100, "Hello, World!");
```

Next, we set the type of box and the font, size, and style of the label:

```
box->box(FL_UP_BOX);
box->labelfont(FL_BOLD + FL_ITALIC);
box->labelsize(36);
box->labeltype(FL_SHADOW_LABEL);
```

We tell FLTK that we will not add any more widgets to window.

```
window->end();
```

Finally, we show the window and enter the FLTK event loop:

```
window->show(argc, argv);  
return Fl::run();
```

The resulting program will display the "Hello, World!" window:

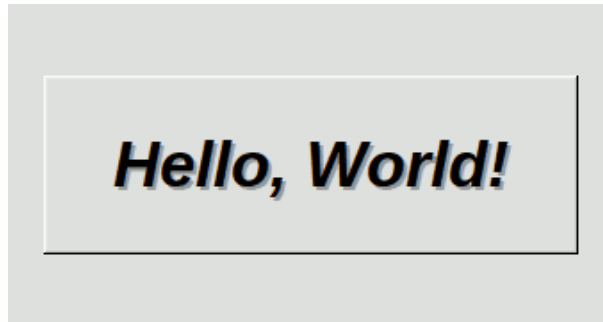


Figure 1.1 The Hello, World! Window

You can quit the program by closing the window or pressing the `ESCAPE` key.

1.3.1.1 Creating the Widgets

The widgets are created using the C++ `new` operator. For most widgets the arguments to the constructor are:

```
Fl_Widget(x, y, width, height, label)
```

The `x` and `y` parameters determine where the widget or window is placed on the screen. In FLTK the top left corner of the window or screen is the origin (i.e. `x = 0`, `y = 0`).

The `width` and `height` parameters determine the size of the widget or window. The maximum widget size is typically governed by the underlying window system or hardware.

[What Units Do FLTK Functions Use?](#) describes the unit FLTK employs for `x`, `y`, `width`, and `height`, and more generally, for all graphical quantities.

`label` is a pointer to a character string to label the widget with or `NULL`. If not specified the label defaults to `NULL`. The label string must be in static storage such as a string constant because FLTK does not make a copy of it - it just uses the pointer.

1.3.1.2 Creating Widget Hierarchies

Widgets are commonly ordered into functional groups, which in turn may be grouped again, creating a hierarchy of widgets. FLTK makes it easy to fill groups by automatically adding all widgets that are created between a `myGroup->begin()` and `myGroup->end()`. In this example, `myGroup` would be the *current* group.

Newly created groups and their derived widgets implicitly call `begin()` in the constructor, effectively adding all subsequently created widgets to itself until `end()` is called.

Calling `end()` on one group widget transfers the "current group" property to the **parent** of that widget. Calling `end()` on a top level window (which has no parent) sets the current group to `NULL`.

Setting the current group to `NULL` will stop automatic hierarchies. New widgets can now be added manually using `Fl_Group::add(...)` and `Fl_Group::insert(...)`.

1.3.1.3 Get/Set Methods

`box->box(FL_UP_BOX)` sets the type of box the [Fl_Box](#) draws, changing it from the default of `FL_NO_BOX`, which means that no box is drawn. In our "Hello, World!" example we use `FL_UP_BOX`, which means that a raised button border will be drawn around the widget. More details are available in the [Box Types](#) section.

You could examine the boxtype by doing `box->box()`. FLTK uses method name overloading to make short names for get/set methods. A "set" method is always of the form "void name(type)", and a "get" method is always of the form "type name() const".

1.3.1.4 Redrawing After Changing Attributes

Almost all of the get/set pairs are very fast, short inline functions and thus very efficient. However, *the "set" methods do not call `redraw()`* - you have to call it yourself. This greatly reduces code size and execution time. The only common exceptions are `value()` which calls `redraw()` and `label()` which calls `redraw_label()` if necessary.

1.3.1.5 Labels

All widgets support labels. In the case of window widgets, the label is used for the label in the title bar. Our example program calls the `labelfont()`, `labelsize()`, and `labeltype()` methods.

The `labelfont()` method sets the typeface and style that is used for the label, which for this example we are using `FL_BOLD` and `FL_ITALIC`.

The `labelsize()` method sets the height of the font in FLTK units.

The `labeltype()` method sets the type of label. FLTK supports normal, embossed, and shadowed labels internally, and more types can be added as desired.

A complete list of all label options can be found in the section on [Labels and Label Types](#).

1.3.1.6 Showing the Window

The `show()` method shows the widget or window. For windows you can also provide the command-line arguments to allow users to customize the appearance, size, and position of your windows.

1.3.1.7 The Main Event Loop

All FLTK applications (and most GUI applications in general) are based on a simple event processing model. User actions such as mouse movement, button clicks, and keyboard activity generate events that are sent to an application. The application may then ignore the events or respond to the user, typically by redrawing a button in the "down" position, adding the text to an input field, and so forth.

FLTK also supports idle, timer, and file pseudo-events that cause a function to be called when they occur. Idle functions are called when no user input is present and no timers or files need to be handled - in short, when the application is not doing anything. Idle callbacks are often used to update a 3D display or do other background processing.

Timer functions are called after a specific amount of time has expired. They can be used to pop up a progress dialog after a certain amount of time or do other things that need to happen at more-or-less regular intervals. FLTK timers are not 100% accurate, so they should not be used to measure time intervals, for example.

File functions are called when data is ready to read or write, or when an error condition occurs on a file. They are most often used to monitor network connections (sockets) for data-driven displays.

FLTK applications must periodically check ([Fl::check\(\)](#)) or wait ([Fl::wait\(\)](#)) for events or use the [Fl::run\(\)](#) method to enter a standard event processing loop. Calling [Fl::run\(\)](#) is equivalent to the following code:

```
while (Fl::wait());
```

[Fl::run\(\)](#) does not return until all of the windows under FLTK control are closed by the user or your program.

1.3.2 Naming Conventions

All public symbols in FLTK start with the characters 'F' and 'L':

- Functions are either `Fl::foo()` or `fl_foo()`.
- Class and type names are capitalized: `Fl_Foo`.
- [Constants and Enumerations](#) are uppercase: `FL_FOO`.
- All header files start with `<FL/...>`.

1.3.3 Header Files

The proper way to include FLTK header files is:

```
#include <FL/Fl_xyz.H>
```

Note

Case *is significant* on many operating systems, and the C standard uses the forward slash (/) to separate directories. *Do not use any of the following include lines:*

```
#include <FL\Fl_xyz.H>
#include <fl/fl_xyz.h>
#include <Fl/fl_xyz.h>
```

1.3.4 Compiling Programs that Use FLTK

Since FLTK 1.4 CMake is the recommended build system. The details below show the "old" methods and reference information in case you like to write your build configuration manually (e.g. Makefiles, Visual Studio, other IDE's ...).

CMake can simplify this task substantially. For now, refer to README.CMake.txt for further information.

Todo This section needs a major rework. Add a chapter "Building FLTK with CMake".

1.3.4.1 Compiling Programs with Standard Compilers

Under UNIX (and under Microsoft Windows when using the GNU development tools) you will probably need to tell the compiler where to find the header files. This is usually done using the `-I` option:

```
c++ -I/usr/local/include ...
```

Note

You need a C++ compiler to build FLTK. The commands given in this chapter are **examples** using `'c++'`. Please replace this command with the C++ compiler suitable for your system or use the `fltk-config` script as described below (this is recommended).

1.3.4.2 Compiling Programs with the 'fltk-config' Script

The `fltk-config` script included with FLTK can be used on systems with a Posix compliant shell, for instance Unix/Linux, macOS, Windows with MinGW, MSYS2, or Cygwin.

Note

`fltk-config` is not designed to work on Windows with Visual Studio compilers. If it works, then only by accident and this is undefined behavior.

```
fltk-config --help
```

displays all available options.

`fltk-config` can be used to get the compiler and the options that are required by your compiler to build a program using the FLTK library:

```
fltk-config --cc
fltk-config --cxx
```

return the C and C++ compiler commands used to build FLTK.

```
c++ `fltk-config --cxxflags` ...
```

can be used to include the required compiler flags in the command line.

Similarly, when linking your application you will need to tell the compiler to use the FLTK library:

```
c++ ... -L/usr/local/lib -lfltk -lXext -lX11 ... -lm -ldl
```

Aside from the "fltk" library, there are also the following libraries

- "fltk_forms" for the XForms compatibility classes (deprecated)
- "fltk_gl" for the OpenGL and GLUT classes
- "fltk_images" for the image file classes, [Fl_Help_Dialog](#) widget, and system icon support.

The libraries are named `fltk.lib`, `fltk_forms.lib`, `fltk_gl.lib`, and `fltk_images.lib` under Windows.

Note

The separate `fltk_cairo` library is no longer necessary since FLTK 1.4.0. However, this release of FLTK builds a dummy `fltk_cairo` library for backwards compatibility. You are advised to remove the usage of the `fltk_cairo` library from your build systems and tools. **The `fltk_cairo` library will be removed in a future release.**

As before, the `fltk-config` script can be used to get the options that are required by your linker:

```
c++ ... `fltk-config --ldflags`
```

The forms, GL, and images libraries are included with the "--use-foo" options, as follows:

```
c++ ... `fltk-config --use-forms --ldflags`
c++ ... `fltk-config --use-gl --ldflags`
c++ ... `fltk-config --use-images --ldflags`
c++ ... `fltk-config --use-cairo --ldflags`
c++ ... `fltk-config --use-forms --use-gl --use-images --ldflags`
```

The option `--use-cairo` may be used to build your program with Cairo libs if you use Cairo in your code. It does no longer include the `fltk_cairo` lib but all necessary Cairo compiler flags and Cairo libs, if and only if FLTK has been built with the optional Cairo support by configure or CMake.

Finally, you can use the `fltk-config` script to compile one or more source files as a FLTK program.

The following examples will create an executable named `filename` (or `filename.exe` under Windows) from a single source file:

```
fltk-config --compile filename.cxx
fltk-config --use-forms --compile filename.cpp
fltk-config --use-gl --compile filename.C
fltk-config --use-images --compile filename.cc
fltk-config --use-cairo --compile filename.cpp
fltk-config --use-forms --use-gl --use-images --compile filename.cpp
```

Note

'`fltk-config --compile`' accepts only a limited set of file extensions for C++ source files: `'.cpp'`, `'.cxx'`, `'.cc'`, and `'.C'` (capital 'C').

1.3.4.3 Compiling Multiple Source Files with 'fltk-config'

Before version 1.4.0 `fltk-config` accepted only a single source file and no additional compiler options or libraries. Since FLTK 1.4.0 it is possible to use additional compiler flags, more than one source file, and additional link libraries.

This is intended to be used for quick prototyping and not for production code development. It can be used to test compiler command options (like `-Wall` or `-Wextra`) or additional link libraries if these are required.

Building from more than one source file with flags and libraries can be achieved as follows:

```
fltk-config [USE-FLAGS] --compile MAIN [FLAGS] [SOURCES] [--link LFLAGS LIBS]
```

where

- arguments in `[ . . . ]` are optional
- `USE-FLAGS` are as described above, e.g. `--use-images`
- `MAIN` is the main C++ source file as documented above
- `FLAGS` are additional compiler flags
- `SOURCES` are additional source files or libraries
- `--link` is used to separate source files and flags from linker flags and libs
- `LFLAGS` are optional linker flags
- `LIBS` are additional libraries to link against

The final commandline is composed like this example:

```
$ fltk-config --compile main.cxx button.o -Wextra x1.a --link -L/usr/include/cairo/ -lcairo
```

```
g++ {fltk-flags} -o main -Wextra main.cxx button.o x1.a {fltk-libs} -L/usr/include/cairo/ -lcairo
```

where `{fltk-flags}` are the compiler flags generated by `fltk-config` as before and `{fltk-libs}` are the usual linker flags and libraries. All optional parameters are used as-is, i.e. there is no syntax checking or special parsing except: the order of flags and source files is preserved (from the commandline) but all flags (`-something`) are positioned before all sources, i.e. arguments w/o leading dash ('-'). All compiler flags and libraries generated from the library build follow all options and source files given on the commandline, and finally everything after `--link` is appended.

1.3.4.4 Compiling Programs with Makefiles

The previous sections described how to use `fltk-config` to build a program from the command line, and this is very convenient for small test programs. But `fltk-config` can also be used to set the compiler and linker options as variables within a Makefile that can be used to build larger programs.

```
CXX      = $(shell fltk-config --cxx)
DEBUG    = -g
CXXFLAGS = $(shell fltk-config --use-gl --use-images --cxxflags ) -I.
LDLFLAGS = $(shell fltk-config --use-gl --use-images --ldflags )
LDSTATIC = $(shell fltk-config --use-gl --use-images --ldstaticflags )
LINK      = $(CXX)

TARGET = cube
OBJS   = CubeMain.o CubeView.o CubeViewUI.o
SRCS   = CubeMain.cxx CubeView.cxx CubeViewUI.cxx

.SUFFIXES: .o .cxx

%.o: %.cxx
    $(CXX) $(CXXFLAGS) $(DEBUG) -c $<

all: $(TARGET)

$(TARGET): $(OBJS)
    $(LINK) -o $(TARGET) $(OBJS) $(LDSTATIC)

CubeMain.o:  CubeMain.cxx CubeViewUI.h
CubeView.o:  CubeView.cxx CubeView.h CubeViewUI.h
CubeViewUI.o: CubeViewUI.cxx CubeView.h

clean:
    rm -f *.o 2> /dev/null
    rm -f $(TARGET) 2> /dev/null
```

1.3.4.5 Compiling Programs with Microsoft Visual C++

In Visual C++ you will need to tell the compiler where to find the FLTK header files. This can be done by selecting "Settings" from the "Project" menu and then changing the "Preprocessor" settings under the "C/C++" tab.

You will also need to add the following libraries to the `Linker` settings:

- `fltk.lib` or `fltkd.lib`, the main FLTK library (postfix 'd' = Debug)
- all FLTK libraries your program requires (`fltk_gl`, `fltk_images`, ...)
- additional libraries like `libpng.lib`, `libjpeg.lib`, etc.
- the Windows Common Controls (`comctl32.lib`) and
- the GDIplus library if used to build FLTK (`gdiplus.lib`) and
- the Windows Socket (`ws2_32.lib`) libraries.

Note

There's a `Linker` setting "Additional Library Directories" or similar; the exact name depends on the Visual Studio version you're using. You can and **should** use this to simplify adding the libraries above. If you set this to the FLTK library path you can just use the library **names** and don't need to use the full paths to all libraries.

You must also define `_WIN32` if the compiler doesn't do this. Currently all known Windows compilers define `_WIN32` - unless you use Cygwin (that's correct, you must not define `_WIN32` if you use Cygwin).

More information can be found in `README.Windows.txt`.

You can build your Microsoft Windows applications as Console or Desktop applications. If you want to use the standard `C main()` function as the entry point, FLTK includes a `WinMain()` function that will call your `main()` function for you.

1.4 Common Widgets and Attributes

This chapter describes many of the widgets that are provided with FLTK and covers how to query and set the standard attributes.

1.4.1 Buttons

FLTK provides many types of buttons:

- `Fl_Button` - A standard push button.
- `Fl_Check_Button` - A button with a check box.
- `Fl_Light_Button` - A push button with a light.
- `Fl_Repeat_Button` - A push button that repeats when held.

- [Fl_Return_Button](#) - A push button that is activated by the `Enter` key.
- [Fl_Round_Button](#) - A button with a radio circle.

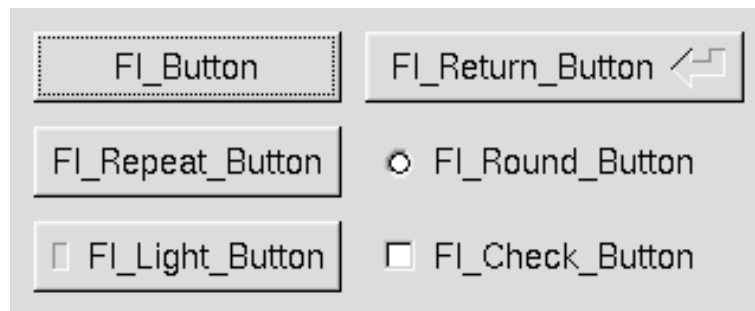


Figure 1.2 FLTK Button Widgets

All of these buttons just need the corresponding `<FL/Fl_xyz_Button.H>` header file. The constructor takes the bounding box of the button and optionally a label string:

```
Fl_Button *button = new Fl_Button(x, y, width, height, "label");
Fl_Light_Button *lbutton = new Fl_Light_Button(x, y, width, height);
Fl_Round_Button *rbutton = new Fl_Round_Button(x, y, width, height, "label");
```

Each button has an associated `type()` which allows it to behave as a push button, toggle button, or radio button:

```
button->type(FL_NORMAL_BUTTON);
lbutton->type(FL_TOGGLE_BUTTON);
rbutton->type(FL_RADIO_BUTTON);
```

For toggle and radio buttons, the `value()` method returns the current button state (0 = off, 1 = on). The `set()` and `clear()` methods can be used on toggle buttons to turn a toggle button on or off, respectively. Radio buttons can be turned on with the `setonly()` method; this will also turn off other radio buttons in the same group.

1.4.2 Text

FLTK provides several text widgets for displaying and receiving text:

- [Fl_Input](#) - A one-line text input field.
- [Fl_Output](#) - A one-line text output field.
- [Fl_Multiline_Input](#) - A multi-line text input field.
- [Fl_Multiline_Output](#) - A multi-line text output field.
- [Fl_Text_Display](#) - A multi-line text display widget.
- [Fl_Text_Editor](#) - A multi-line text editing widget.
- [Fl_Help_View](#) - A HTML text display widget.

The [Fl_Output](#) and [Fl_Multiline_Output](#) widgets allow the user to copy text from the output field but not change it.

The `value()` method is used to get or set the string that is displayed:

```
Fl_Input *input = new Fl_Input(x, y, width, height, "label");
input->value("Now is the time for all good men...");
```

The string is copied to the widget's own storage when you set the `value()` of the widget.

The [Fl_Text_Display](#) and [Fl_Text_Editor](#) widgets use an associated [Fl_Text_Buffer](#) class for the value, instead of a simple string.

1.4.3 Valuators

Unlike text widgets, valuators keep track of numbers instead of strings. FLTK provides the following valuators:

- [FL_Counter](#) - A widget with arrow buttons that shows the current value.
- [FL_Dial](#) - A round knob.
- [FL_Roller](#) - An SGI-like dolly widget.
- [FL_Scrollbar](#) - A standard scrollbar widget.
- [FL_Slider](#) - A scrollbar with a knob.
- [FL_Value_Slider](#) - A slider that shows the current value.

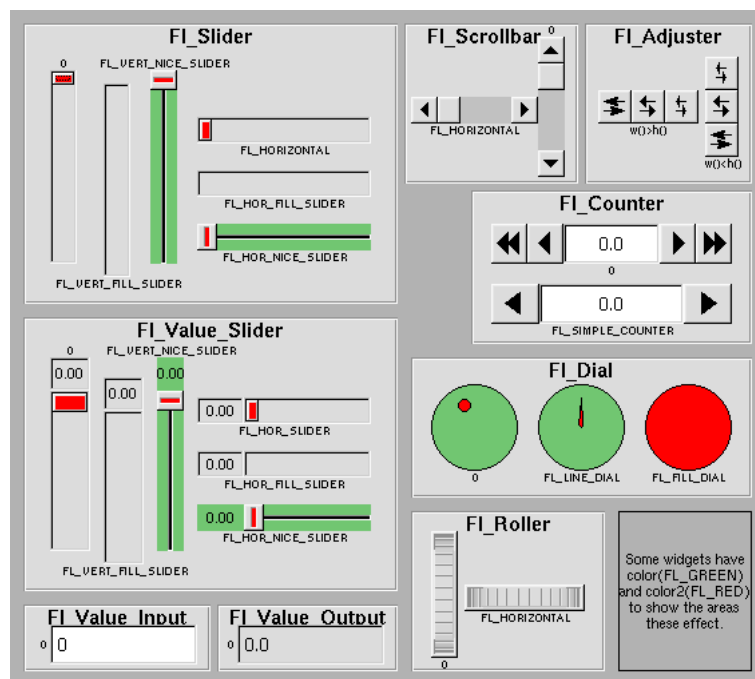


Figure 1.3 FLTK valuator widgets

The `value()` method gets and sets the current value of the widget. The `minimum()` and `maximum()` methods set the range of values that are reported by the widget.

1.4.4 Groups

The `FL_Group` widget class is used as a general purpose "container" widget. Besides grouping radio buttons, the groups are used to encapsulate windows, tabs, and scrolled windows. The following group classes are available with FLTK:

- `FL_Double_Window` - A double-buffered window on the screen.
- `FL_Flex` - A flexible container of one row or column of widgets. `FL_Flex` widgets may be nested, but see also `FL_Grid`.
- `FL_Gl_Window` - An OpenGL window on the screen.
- `FL_Grid` - A flexible container of one or more rows and columns of widgets arranged in a "grid" with auto-layout features. `FL_Grid` widgets can be nested with other `FL_Grid` or many `FL_Group` type widgets. Nesting with other self-resizing containers like `FL_Pack` and `FL_Tile` is not recommended but **may** work.
- `FL_Group` - The base container class; can be used to group any widgets together.
- `FL_Pack` - A collection of widgets that are packed into the group area.
- `FL_Scroll` - A scrolled window area.
- `FL_Tabs` - Displays child widgets as tabs.
- `FL_Tile` - A tiled window area.
- `FL_Window` - A window on the screen.
- `FL_Wizard` - Displays one group of widgets at a time.

1.4.5 Setting the Size and Position of Widgets

The size and position of widgets is usually set when you create them. You can access them with the `x()`, `y()`, `w()`, and `h()` methods.

You can change the size and position by using the `position()`, `resize()`, and `size()` methods:

```
button->position(x, y);
group->resize(x, y, width, height);
window->size(width, height);
```

If you change a widget's size or position after it is displayed you will have to call `redraw()` on the widget's parent.

1.4.6 Colors

FLTK stores the colors of widgets as a 32-bit unsigned number that is either an index into a color palette of 256 colors ($0 \leq \text{color} \leq 255$) or a 24-bit RGB color ($\text{color} > 255$). The color palette is *not* the X or Windows colormap, but instead is an internal table with fixed contents.

See the [Colors](#) section of [Drawing Things in FLTK](#) for implementation details.

There are symbols for naming some of the more common colors:

- FL_BLACK
- FL_RED
- FL_GREEN
- FL_YELLOW
- FL_BLUE
- FL_MAGENTA
- FL_CYAN
- FL_WHITE

Other symbols are used as the default colors for all FLTK widgets.

- FL_FOREGROUND_COLOR
- FL_BACKGROUND_COLOR
- FL_INACTIVE_COLOR
- FL_SELECTION_COLOR

The full list of named color values can be found in [FLTK Enumerations](#).

A color value can be created from its RGB components by using the `fl_rgb_color()` function, and decomposed again with `Fl::get_color()`:

```
Fl_Color c = fl_rgb_color(85, 170, 255);    // RGB to Fl_Color
Fl::get_color(c, r, g, b);                  // Fl_Color to RGB
```

The widget color is set using the `color()` method:

```
button->color(FL_RED);                      // set color using named value
```

Similarly, the label color is set using the `labelcolor()` method:

```
button->labelcolor(FL_WHITE);
```

The `Fl_Color` encoding maps to a 32-bit unsigned integer representing RGBI, so it is also possible to specify a color using a hex constant as a color map index:

```
button->color(0x000000ff);                  // colormap index #255 (FL_WHITE)
```

or specify a color using a hex constant for the RGB components:

```
button->color(0xff000000);                   // RGB: red
button->color(0x00ff0000);                   // RGB: green
button->color(0x0000ff00);                   // RGB: blue
button->color(0xffff0000);                   // RGB: white
```

Note

If TrueColor is not available, any RGB colors will be set to the nearest entry in the colormap.

1.4.7 Box Types

The type `Fl_Boxtype` stored and returned in `Fl_Widget::box()` is an enumeration defined in `Enumerations.H`.

These are the standard box types included with FLTK:

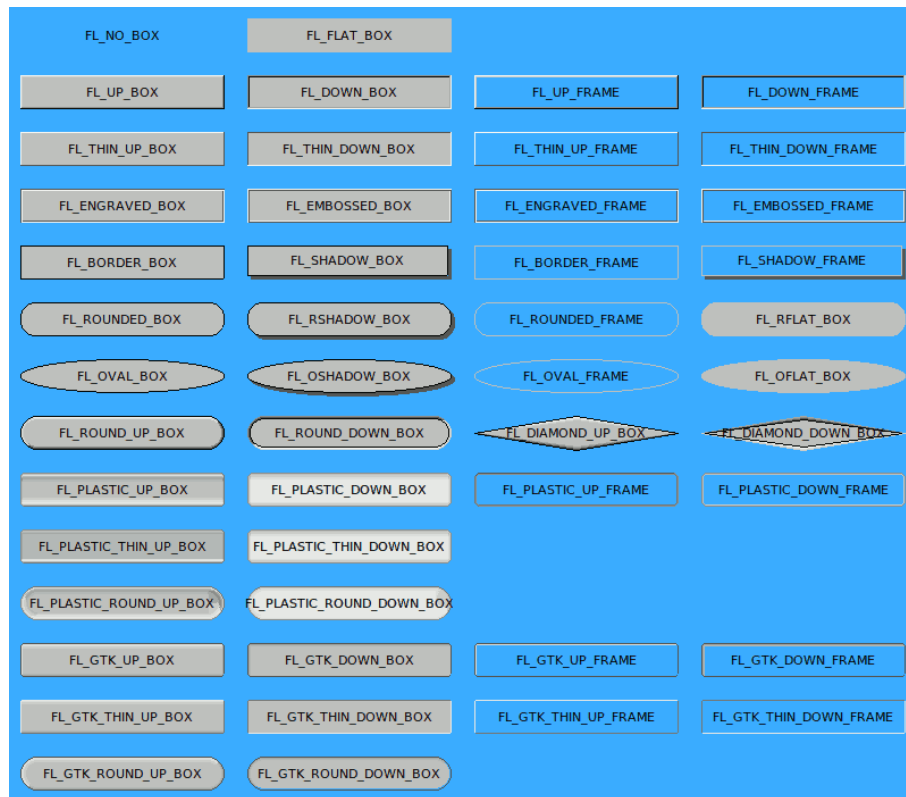


Figure 1.4 FLTK Standard Box Types

`FL_NO_BOX` means nothing is drawn at all, so whatever is already on the screen remains. The `FL_..._FRAME` types only draw their edges, leaving the interior unchanged. The blue color in the image above is the area that is not drawn by the frame types.

1.4.7.1 Making Your Own Boxtypes

You can define your own boxtypes by making a small function that draws the box and adding it to the table of boxtypes.

The Drawing Function

The drawing function is passed the bounding box and background color for the widget:

```
void xyz_draw(int x, int y, int w, int h, Fl_Color c) {
    ...
}
```

A simple drawing function might fill a rectangle with the given color and then draw a black outline:

```
void xyz_draw(int x, int y, int w, int h, Fl_Color c) {
    fl_color(c);
    fl_rectf(x, y, w, h);
    fl_color(FL_BLACK);
    fl_rect(x, y, w, h);
}
```

`Fl_Boxtype fl_down(Fl_Boxtype b)`

`fl_down()` returns the "pressed" or "down" version of a box. If no "down" version of a given box exists, the behavior of this function is undefined and some random box or frame is returned. See [Drawing Functions](#) for more details.

`Fl_Boxtype fl_frame(Fl_Boxtype b)`

`fl_frame()` returns the unfilled, frame-only version of a box. If no frame version of a given box exists, the behavior of this function is undefined and some random box or frame is returned. See [Drawing Functions](#) for more details.

`Fl_Boxtype fl_box(Fl_Boxtype b)`

`fl_box()` returns the filled version of a frame. If no filled version of a given frame exists, the behavior of this function is undefined and some random box or frame is returned. See [Drawing Functions](#) for more details.

Adding Your Box Type

The `Fl::set_boxtype()` method adds or replaces the specified box type:

```
#define XYZ_BOX FL_FREE_BOXTYPE
Fl::set_boxtype(XYZ_BOX, xyz_draw, 1, 1, 2, 2);
```

The last 4 arguments to `Fl::set_boxtype()` are the offsets for the `x`, `y`, `width`, and `height` values that should be subtracted when drawing the label inside the box.

A complete box design contains four box types in this order: a filled, neutral box (`UP_BOX`), a filled, depressed box (`DOWN_BOX`), and the same as outlines only (`UP_FRAME` and `DOWN_FRAME`). The function `fl_down(Fl_Boxtype)` expects the neutral design on a boxtype with a numerical value evenly dividable by two. `fl_frame(Fl_Boxtype)` expects the `UP_BOX` design at a value dividable by four.

1.4.8 Labels and Label Types

The `label()`, `align()`, `labelfont()`, `labelsize()`, `labeltype()`, `image()`, and `deimage()` methods control the labeling of widgets.

label()

The `label()` method sets the string that is displayed for the label. Symbols can be included with the label string by escaping them using the "@" symbol - "@@" displays a single at sign. These are the available symbols:



Figure 1.5 FLTK label symbols

The @ sign may also be followed by the following optional "formatting" characters, in this order:

- '#' forces square scaling, rather than distortion to the widget's shape.
- +[1-9] or -[1-9] tweaks the scaling a little bigger or smaller.
- '\$' flips the symbol horizontally, '%' flips it vertically.
- [0-9] - rotates by a multiple of 45 degrees. '5' and '6' do no rotation while the others point in the direction of that key on a numeric keypad. '0', followed by four more digits rotates the symbol by that amount in degrees.

Thus, to show a very large arrow pointing downward you would use the label string "@+92->".

Symbols and text can be combined in a label, however the symbol must be at the beginning and/or at the end of the text. If the text spans multiple lines, the symbol or symbols will scale up to match the height of all the lines.

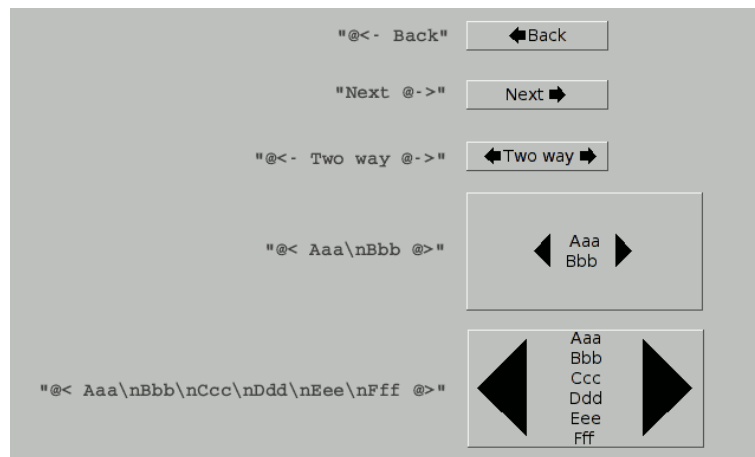


Figure 1.6 FLTK symbols and text

align()

The `align()` method positions the label. The following constants are defined and may be OR'd together as needed:

- `FL_ALIGN_CENTER` - center the label in the widget.
- `FL_ALIGN_TOP` - align the label at the top of the widget.
- `FL_ALIGN_BOTTOM` - align the label at the bottom of the widget.
- `FL_ALIGN_LEFT` - align the label to the left of the widget.
- `FL_ALIGN_RIGHT` - align the label to the right of the widget.
- `FL_ALIGN_LEFT_TOP` - The label appears to the left of the widget, aligned at the top. Outside labels only.
- `FL_ALIGN_RIGHT_TOP` - The label appears to the right of the widget, aligned at the top. Outside labels only.
- `FL_ALIGN_LEFT_BOTTOM` - The label appears to the left of the widget, aligned at the bottom. Outside labels only.
- `FL_ALIGN_RIGHT_BOTTOM` - The label appears to the right of the widget, aligned at the bottom. Outside labels only.
- `FL_ALIGN_INSIDE` - align the label inside the widget.
- `FL_ALIGN_CLIP` - clip the label to the widget's bounding box.
- `FL_ALIGN_WRAP` - wrap the label text as needed.
- `FL_ALIGN_TEXT_OVER_IMAGE` - show the label text over the image.
- `FL_ALIGN_IMAGE_OVER_TEXT` - show the label image over the text (default).
- `FL_ALIGN_IMAGE_NEXT_TO_TEXT` - The image will appear to the left of the text.
- `FL_ALIGN_TEXT_NEXT_TO_IMAGE` - The image will appear to the right of the text.
- `FL_ALIGN_IMAGE_BACKDROP` - The image will be used as a background for the widget.

labeltype()

The `labeltype()` method sets the type of the label. The following standard label types are included:

- `FL_NORMAL_LABEL` - draws the text.
- `FL_NO_LABEL` - does nothing.
- `FL_SHADOW_LABEL` - draws a drop shadow under the text.
- `FL_ENGRAVED_LABEL` - draws edges as though the text is engraved.
- `FL_EMBOSSED_LABEL` - draws edges as though the text is raised.
- `FL_ICON_LABEL` - draws the icon ([Fl_Image](#)) associated with the text.
- `FL_IMAGE_LABEL` - draws the image ([Fl_Image](#)) associated with the text.
- `FL_MULTI_LABEL` - draws multiple parts side by side, see [Fl_Multi_Label](#).

Note

Some of these labeltypes are no longer necessary for normal widgets. Widgets allow for an image and a text side by side, depending on the widget's `align()` flag. `FL_MULTI_LABEL` was designed to be used with [Fl_Menu_Item](#)'s to support icons or small images, typically left of the menu text.

As of this writing (FLTK 1.4.0) `Fl_Menu_Items` support only one label part (text **or** image), but using [Fl_Multi_Label](#) as the label can extend this to more than one part.

See also

class [Fl_Multi_Label](#), [Fl_Widget::align\(\)](#)

image() and deimage()

The `image()` and `deimage()` methods set an image that will be displayed with the widget. The `deimage()` method sets the image that is shown when the widget is inactive, while the `image()` method sets the image that is shown when the widget is active.

To make an image you use a subclass of [Fl_Image](#).

Making Your Own Label Types

Label types are actually indexes into a table of functions that draw them. The primary purpose of this is to use this to draw the labels in ways inaccessible through the `fl_font()` mechanism (e.g. `FL_ENGRAVED_LABEL`) or with program-generated letters or symbology.

Label Type Functions

To setup your own label type you will need to write two functions: one to draw and one to measure the label. The draw function is called with a pointer to a `Fl_Label` structure containing the label information, the bounding box for the label, and the label alignment:

```
void xyz_draw(const Fl_Label *label, int x, int y, int w, int h, Fl_Align align) {
    ...
}
```

The label should be drawn *inside* this bounding box, even if `FL_ALIGN_INSIDE` is not enabled. The function is not called if the label value is `NULL`.

The measure function is called with a pointer to a `Fl_Label` structure and references to the width and height:

```
void xyz_measure(const Fl_Label *label, int &w, int &h) {
    ...
}
```

The function should measure the size of the label and set `w` and `h` to the size it will occupy.

Adding Your Label Type

The `Fl::set_labeltype()` method creates a label type using your draw and measure functions:

```
#define XYZ_LABEL FL_FREE_LABELTYPE

Fl::set_labeltype(XYZ_LABEL, xyz_draw, xyz_measure);
```

The label type number `n` can be any integer value starting at the constant `FL_FREE_LABELTYPE`. Once you have added the label type you can use the `labeltype()` method to select your label type.

The `Fl::set_labeltype()` method can also be used to overload an existing label type such as `FL_NORMAL_LABEL`.

Making your own symbols

It is also possible to define your own drawings and add them to the symbol list, so they can be rendered as part of any label.

To create a new symbol, you implement a drawing function `void drawit(Fl_Color c)` which typically uses the functions described in [Drawing Complex Shapes](#) to generate a vector shape inside a two-by-two units sized box around the origin. This function is then linked into the symbols table using `fl_add_symbol()`:

```
int fl_add_symbol(const char *name, void (*drawit)(Fl_Color), int scalable)
```

`name` is the name of the symbol without the "@"; `scalable` must be set to 1 if the symbol is generated using scalable vector drawing functions.

```
int fl_draw_symbol(const char *name, int x, int y, int w, int h, Fl_Color col)
```

This function draws a named symbol fitting the given rectangle.

1.4.9 Callbacks

Callbacks are functions that are called when the value of a widget changes. A callback function is sent a `Fl_Widget` pointer of the widget that changed and a pointer to data that you provide:

```
void xyz_callback(Fl_Widget *w, void *data) {
    ...
}
```

The `callback()` method sets the callback function for a widget. You can optionally pass a pointer to some data needed for the callback:

```
int xyz_data;

button->callback(xyz_callback, &xyz_data);
```

Note

You cannot delete a widget inside a callback, as the widget may still be accessed by FLTK after your callback is completed. Instead, use the `Fl::delete_widget()` method to mark your widget for deletion when it is safe to do so.

Many programmers new to FLTK or C++ try to use a non-static class method instead of a static class method or function for their callback. Since callbacks are done outside a C++ class, the `this` pointer is not initialized for class methods.

To work around this problem, define a static method in your class that accepts a pointer to the class, and then have the static method call the class method(s) as needed. The data pointer you provide to the `callback()` method of the widget can be a pointer to the instance of your class.

```
class Foo {
    void my_callback(Fl_Widget *w);
    static void my_static_callback(Fl_Widget *w, void *f) { ((Foo *)f)->my_callback(w); }
    ...
}
...
w->callback(my_static_callback, (void *)this);
```

In an effort to make callbacks easier, more flexible, and type safe, FLTK provides three groups of macros that generate the code needed to call class methods directly with up to five custom parameters.

- `FL_FUNCTION_CALLBACK_#(WIDGET, FUNCTION, ...)` creates code for callbacks to functions and static class methods with up to five arguments. The `#` must be replaced by the number of callback arguments.
- `FL_METHOD_CALLBACK_#(WIDGET, CLASS, SELF, METH, ...)` creates code for callbacks to arbitrary public class methods
- `FL_INLINE_CALLBACK_#(WIDGET, ..., FUNCTION_BODY)` creates code for callback functions that are very close to (almost in the same line) the widget creation code, similar to lambda function in C++11. The last argument of this macro is the callback code.

The syntax is a bit unconventional, but the resulting code is flexible and needs no additional maintenance. It is also C++98 compatible. For example:

```
#include <FL/fl_callback_macros.H>
...
std::string *str = new std::string("FLTK");
Fl_Button *btn = new Fl_Button(10, 10, 100, 100);
FL_METHOD_CALLBACK_2(btn, std::string, str, insert, int, 2, const char*, "...");
...
Fl_Button *inline_cb_btn_2 = new Fl_Button(390, 60, 180, 25, "2 args");
FL_INLINE_CALLBACK_2( inline_cb_btn_2,
    const char *, text, "FLTK", int, number, 2,
    {
        fl_message("We received the message %s with %d!", text, number);
    }
);
```

See also

`Fl_Widget::callback(Fl_Callback*, void*)`, `FL_FUNCTION_CALLBACK_3`, `FL_METHOD_CALLBACK_1`, `FL_INLINE_CALLBACK_2`

1.4.10 When and Reason

Normally callbacks are performed only when the value of the widget changes. You can change this using the `Fl_Widget::when()` method:

```
button->when (FL_WHEN_NEVER);
button->when (FL_WHEN_CHANGED);
button->when (FL_WHEN_RELEASE);
button->when (FL_WHEN_RELEASE_ALWAYS);
button->when (FL_WHEN_ENTER_KEY);
button->when (FL_WHEN_ENTER_KEY_ALWAYS);
button->when (FL_WHEN_CHANGED | FL_WHEN_NOT_CHANGED);
```

Within the callback, you can query why the callback was called using `Fl::callback_reason()`. For example, setting `myInput->when (FL_WHEN_RELEASE | FL_WHEN_CHANGED)`

for a text input field may return `FL_REASON_LOST_FOCUS` or `FL_REASON_CHANGED` as a callback reason.

1.4.11 Shortcuts

Shortcuts are key sequences that activate widgets such as buttons or menu items. The `shortcut()` method sets the shortcut for a widget:

```
button->shortcut (FL_Enter);
button->shortcut (FL_SHIFT + 'b');
button->shortcut (FL_CTRL + 'b');
button->shortcut (FL_ALT + 'b');
button->shortcut (FL_CTRL + FL_ALT + 'b');
button->shortcut (0); // no shortcut
```

The shortcut value is the key event value - the ASCII value or one of the special keys described in `Fl::event_key()` Values combined with any modifiers like Shift, Alt, and Control.

1.5 Coordinates and Layout Widgets

This chapter describes the coordinate systems that apply when positioning widgets manually, and some of the basics of FLTK layout widgets that are used to position widgets automatically.

1.5.1 The Widget Coordinate System

All widgets have constructors with `x` and `y` parameters to let the programmer specify the desired initial position of the top left corner during explicit manual layout within `Fl_Window` and `Fl_Group` container widgets.

This position is always relative to the enclosing `Fl_Window`, which is usually, but not always, the top-level application window, or a free-floating pop-up dialog window. In some cases it could also be a subwindow embedded in a higher-level window, as shown in the figure below.

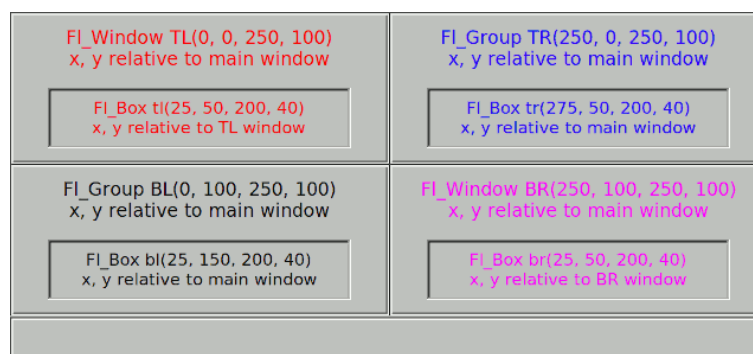


Figure 1.7 FLTK coordinate system

The positions of the TL and BR sub-windows and the TR and BL groups are all relative to the top-left corner of the main window. The positions of the boxes inside the TR and BL groups are also relative to the main window, but the boxes inside the TL and BR sub-windows are positioned relative to the enclosing sub-window.

In other words, the widget hierarchy and positions can be summarized as:

```
Fl_Window main window
  Fl_Window TL subwindow      # x, y relative to main window
    Fl_Box tl box             # x, y relative to TL subwindow
  Fl_Window BR subwindow      # x, y relative to main window
    Fl_Box br box             # x, y relative to BR subwindow
  Fl_Group TR group           # x, y relative to main window
    Fl_Box tr box             # x, y relative to main window
  Fl_Group BL group           # x, y relative to main window
    Fl_Box bl box             # x, y relative to main window
```

1.5.2 Layout and Container Widgets

There are four main groups of widgets derived from `Fl_Group` for a range of different purposes.

The first group are composite widgets that each contain a fixed set of components that work together for a specific purpose, rather than layout widgets as such, and are not discussed here.

The second group are basically containers offering the same manual layout features as `Fl_Group`, as described above, but which add one new capability. These widgets are `Fl_Scroll`, `Fl_Tabs` and `Fl_Wizard`.

The third group are layout managers that relocate and resize the child widgets added to them in order to satisfy a particular layout algorithm. These widgets are `Fl_Flex`, `Fl_Grid`, `Fl_Pack`, and `Fl_Tile`.

The final group consists of `Fl_Window` and its derivatives. Their special capability is that they can be top-level application windows and dialogs that interface with the operating system window manager, but can also be embedded within other windows and groups as shown in the example above. Note that the window manager may impose its own constraints on the position of top-level windows, and the `x` and `y` position parameters may be treated as hints, or even ignored. The `Fl_Window` class has an extra constructor that omits them.

Descriptions of layout and container widgets follow in alphabetical order.

1.5.2.1 The `Fl_Flex` Layout Widget

The `Fl_Flex` widget allows the layout of its direct children as a single row or column. If its `type()` is set to give the row or horizontal layout, the children are all resized to have the same height as the `Fl_Flex` and are moved next to each other. If set to give the column or vertical layout, the children are all resized to have the same width as the `Fl_Flex` and are then stacked below each other.

Widget positions (`x`, `y`) need not be given by the user because widgets are positioned inside the `Fl_Flex` container in the order of its children. Widget sizes can be set to (0, 0) as in `Fl_Pack` since they are calculated by `Fl_Flex`.

This is similar to `Fl_Pack` described below and `Fl_Flex` is designed to act as a drop-in replacement of `Fl_Pack` with some minor differences.

Other than `Fl_Pack` the `Fl_Flex` widget does **not** resize itself but resizes its children to fill the entire space of the `Fl_Flex` container. Single children of `Fl_Flex` can be set to fixed sizes to inhibit this resizing behavior. In this case the remaining space is distributed to all non-fixed widgets.

`Fl_Flex` widgets can be nested inside each other and with `Fl_Grid` in any combination.

The name `Fl_Flex` was inspired by the CSS 'flex' container.

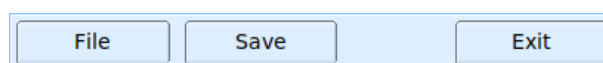


Figure 1.8 Simple `Fl_Flex` Layout

`Fl_Flex` was added in FLTK 1.4.0.

1.5.2.2 The `Fl_Grid` Layout Widget

`Fl_Grid` is the most flexible layout container in FLTK 1.4. It is based on a flexible grid of **cells** that can be assigned one widget per cell which is the *anchor* of the widget. Widgets can span multiple rows and columns and the cells can constitute a sparse matrix. Widgets can be aligned inside their cells in several ways (left, right, top, bottom) and can stretch horizontally, vertically, or both, i.e. fill the entire cell.

Widget positions (x, y) need not be given by the user because widgets are assigned to a particular grid cell by row and column number. Widget sizes can be given as their **minimal** sizes and will be resized appropriately depending on the free space.

Optional margins around all cells inside the widget border and gaps between rows and cells make the layout even more flexible.

The `Fl_Grid` widget should be designed with a grid (matrix) and its minimal size in mind. It is designed to **enlarge** cells and widgets in a flexible way when the `Fl_Grid` widget itself is created or resized.

Additional free space inside the `Fl_Grid` container is distributed to widgets by considering minimal row heights, column widths, sizes of widgets, and row and column *weights*. These weights are used to distribute the free space proportionally according to the row and column weights.

`Fl_Grid` widgets can be nested inside each other and with `Fl_Flex` and other subclasses of `Fl_Group` in any combination.

Note

We don't recommend to use `Fl_Pack` as child widgets although this **may** work as well.

The name `Fl_Grid` was inspired by the CSS 'grid' container but it has some properties in common with HTML `<table>` containers as well, for instance row and column spanning.

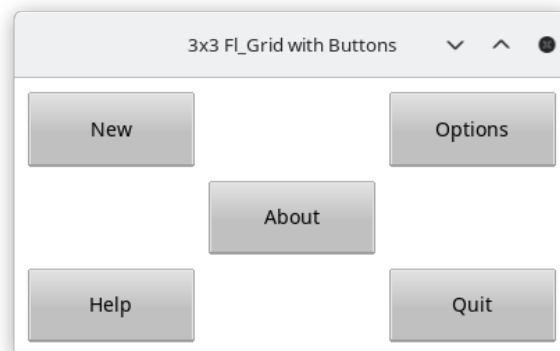


Figure 1.9 Simple `Fl_Grid` Layout

`Fl_Grid` was added in FLTK 1.4.0.

1.5.2.3 The FL_Pack Layout Widget

The `FL_Pack` widget allows the layout of its direct children as a single row, or column. If its `type()` is set to give the row or horizontal layout, the children are all resized to have the same height as the `FL_Pack` and are moved next to each other. If set to give the column or vertical layout, the children are all resized to have the same width as the `FL_Pack` and are then stacked below each other. The `FL_Pack` then resizes itself to shrink-wrap itself around all of the children.

`FL_Pack` widgets are often used inside an `FL_Scroll`, as shown in the diagram below, to avoid having to deal with tricky resize behavior when used with nested widgets.

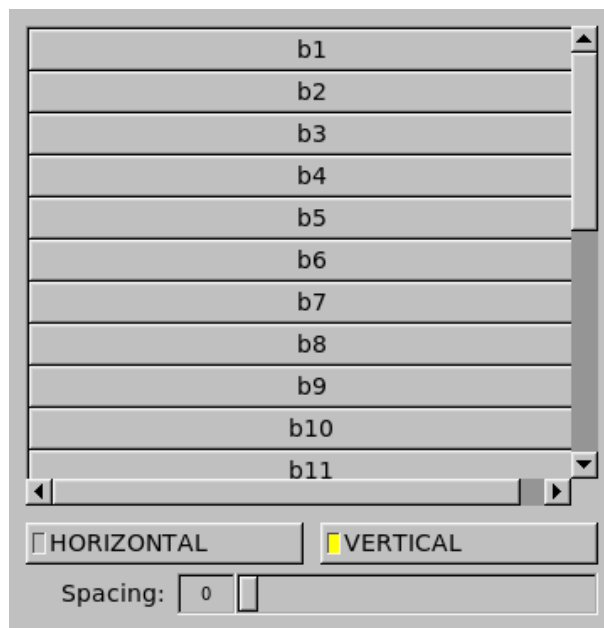


Figure 1.10 `FL_Pack` test program screenshot

Since FLTK 1.4.0 `FL_Flex` (described above) can in many cases be used as a drop-in replacement for `FL_Pack` if this "shrink-wrap" behavior is not required. Note that the `FL_Pack` layout algorithm can cause some issues because its widget size can change depending on its children and particularly because this is done late, i.e. during `draw()` and not as usual during `resize` of the window.

Note

We recommend that developers evaluate whether using `FL_Flex` or `FL_Grid` instead of `FL_Pack` can be a better solution with more predictable and reliable resizing behavior of the overall program layout.

1.5.2.4 The FL_Scroll Container Widget

The `FL_Scroll` container widget can hold an assortment of widgets that may extend beyond its own width and height, in which case horizontal and/or vertical scrollbars may appear automatically so that you can scroll and view the entire contents.

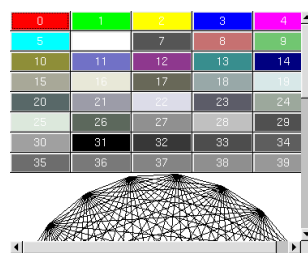


Figure 1.11 `FL_Scroll` container widget

1.5.2.5 The `Fl_Tabs` Container Widget

The `Fl_Tabs` widget provides a front-to-back stack of individual panels which usually contain `Fl_Group` widgets and their children. The user can switch between panels by clicking on the small tabs that protrude from the panels. The appearance of each tab is determined by the child widget's label and related attributes.

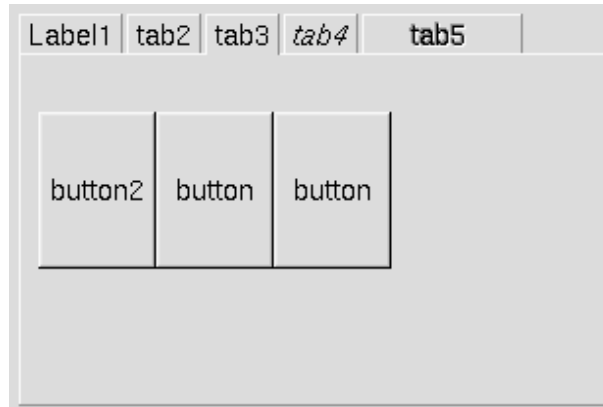


Figure 1.12 `Fl_Tabs` container widget

1.5.2.6 The `Fl_Tile` Layout Widget

The `Fl_Tile` widget allows the user to resize one or more of its children by dragging on the border between adjacent child widgets. However, the programmer must first explicitly layout the child widgets so that their borders exactly fill the width and height of the `Fl_Tile` without having any gaps between them, or at the edges. Some care is needed when initially positioning the children and setting the `resizable()` widget within the `Fl_Tile` to prevent squeezing a child to have a zero width or height. For more information see the `Fl_Tile` widget manual page, and [How Does Resizing Work?](#).

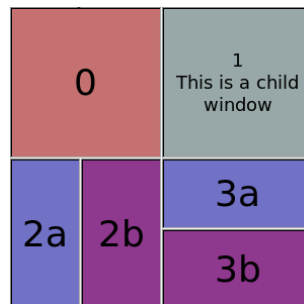


Figure 1.13 The `Fl_Tile` layout widget

1.5.2.7 The `Fl_Wizard` Container Widget

The `Fl_Wizard` widget derives from the `Fl_Tabs` class, but instead of having tabs that the user can click to select the corresponding panel, the programmer uses the `prev()`, `next()` or `value()` methods to show the appropriate panel. For example, the user might be able to click on "Next" and "Prev" navigation buttons or keys, as shown below.



Figure 1.14 `Fl_Wizard` container widget

1.6 How Does Resizing Work?

This chapter describes the basic mechanism behind the creation of resizable user interface elements in FLTK.

FLTK uses a simple, but very versatile system to resize even the most complex dialogs and interfaces. The resizing is implemented within the `Fl_Group` widget, and the exact resizing behavior of that group is determined by its `resizable()` attribute.

1.6.1 Resizing can be disabled

Summary:

```
group = new Fl_Group(xg, yg, wg, hg, "No Resizing");
child1 = new Fl_Box(xb, yb, wb, hb, "B"); // or other widget type
...
group->resizable((Fl_Widget*)0); // no resizing
group->end()
```

The `resizable` may be set to the NULL pointer, which means that the group will not resize. Note that this is the default behavior for `Fl_Window` and `Fl_Pack` derived widgets, and therefore the programmer must explicitly set the window's `resizable` attribute if they want to allow the window to be resized.

1.6.2 Resizing can be simple

Summary:

```
group = new Fl_Group(xg, yg, wg, hg, "Simple Resizing");
child1 = new Fl_Box(xb, yb, wb, hb, "B"); // or other widget type
...
group->resizable(group); // simple proportional resizing
group->end()
```

The `resizable` may be set to the group itself, which means that all widgets within the group will resize as the group itself is resized. This is the default behavior for `Fl_Group` widgets, and is shown in the diagram below.

If the group is stretched horizontally, the widths of the widgets within the group are adjusted proportionally. The same is true for vertical resizing.

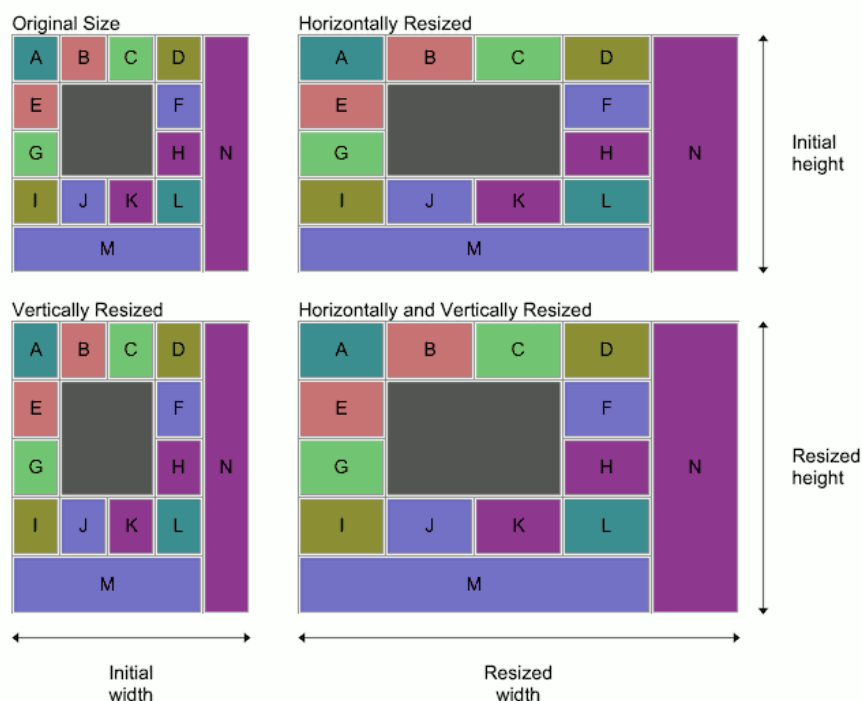


Figure 1.15 Proportional resizing example

1.6.3 Resizing can be complex

Summary:

```
group = new Fl_Group(xg, yg, wg, hg, "Complex Resizing");
child1 = new Fl_Box(xb, yb, wb, hb, "B"); // or other widget type
...
group->resizable(child1); // complex resizing
group->end()
```

It is when the group's `resizable` attribute is set to one of the group's child widgets, that things become really interesting.

In the diagram below, imagine vertical lines extending from the left and right sides of the yellow widget marked "resizable", and horizontal lines extending from the top and bottom sides. Exactly which widgets resize, and by how much, is determined by which ones lie completely or partially within this cross.

The widgets marked B, C, J, K and M clearly lie completely or partially within the vertical part of the cross; the widgets marked E, F, G, H and N lie completely or partially within the horizontal part of the cross; and the widgets marked A, D, I and L do not overlap with the cross at all. The resizing behavior is as follows:

- the width and height of the `resizable` widget increase to match the change in the width and height of the group widget as it is stretched;
- the widths of those widgets that overlap with the vertical part of the cross increase proportionally as the width of the group widget increases, but their heights remain unchanged, i.e. the widgets marked B, C, J, K and M;
- the heights of those widgets that overlap with the horizontal part of the cross increase proportionally as the height of the group widget increases, but their widths remain unchanged, i.e. the widgets marked E, F, G, H and N;
- the widths and heights of the remaining widgets stay the same, i.e. the widgets marked A, D, I and L stay the same size.

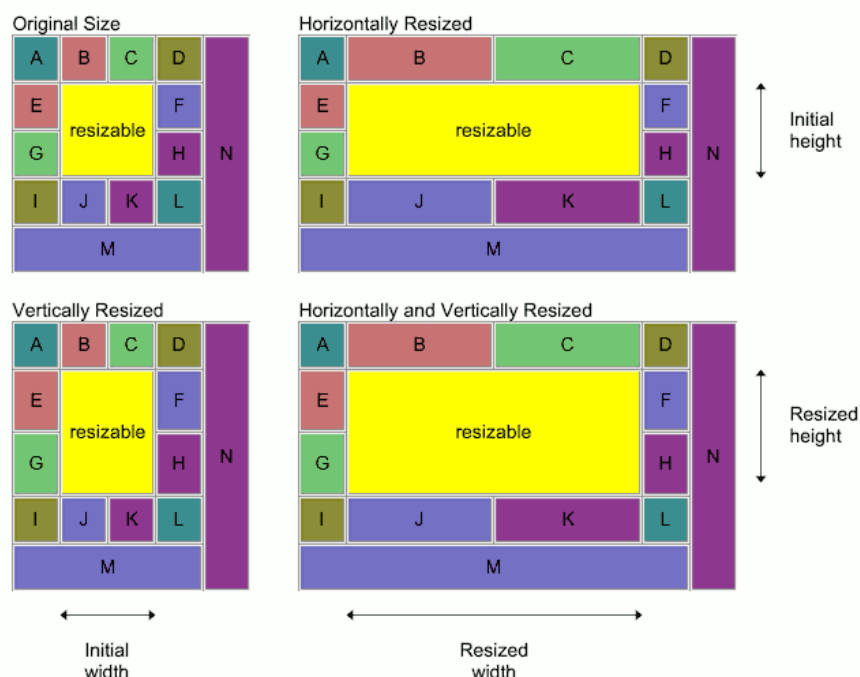


Figure 1.16 Complex resizing example

1.6.4 Practical examples

Why is this so powerful, you may ask. Well, every widget group can have a completely independent resizing strategy. By replacing one or more of the group's "normal" child widgets with another group widget where all of the above rules can be applied again, it is possible to create a hierarchy of group widgets with very complex layouts and resizing behavior.

Consider a simple dialog box, consisting of an icon box and a message area on the top and a button at the bottom right: which widget should be the `resizable` one?

Setting the `resizable` to be the icon box won't give us what we want:

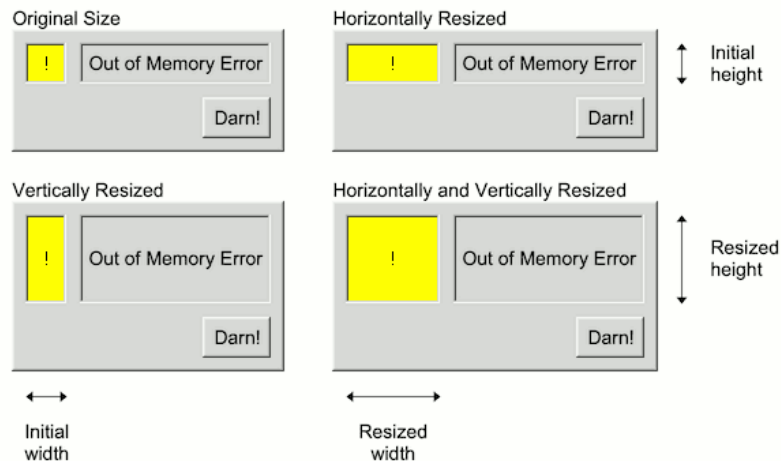


Figure 1.17 Resizing dialog example (a)

The message text area would be the logical choice so that the user can expand the dialog to see if there is more of an explanation below the short error message. This results in the behavior shown in the diagram below.

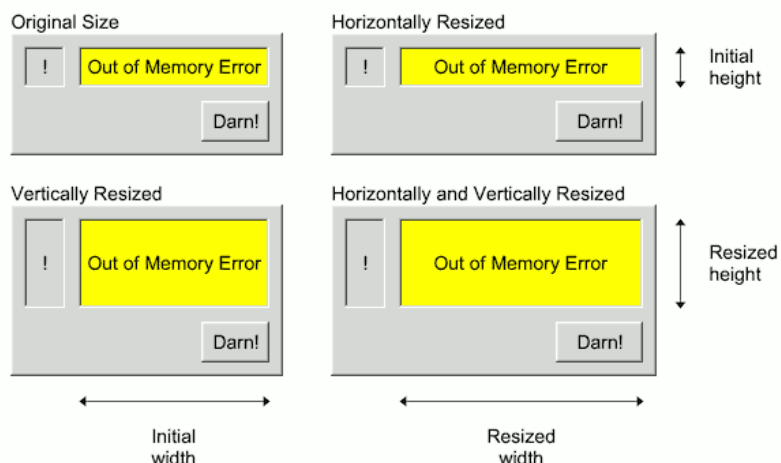


Figure 1.18 Resizing dialog example (b)

The result is close to what we want, but not quite: the text area will fully resize, the "!" icon box will resize vertically but not horizontally, which we can live with, but the "Darn!" button will - wait a minute - resize horizontally?

That's ugly. How do we stop that from happening? Simple: put it in its own group and set the `resizable` to an invisible box widget, as shown in the diagram below.

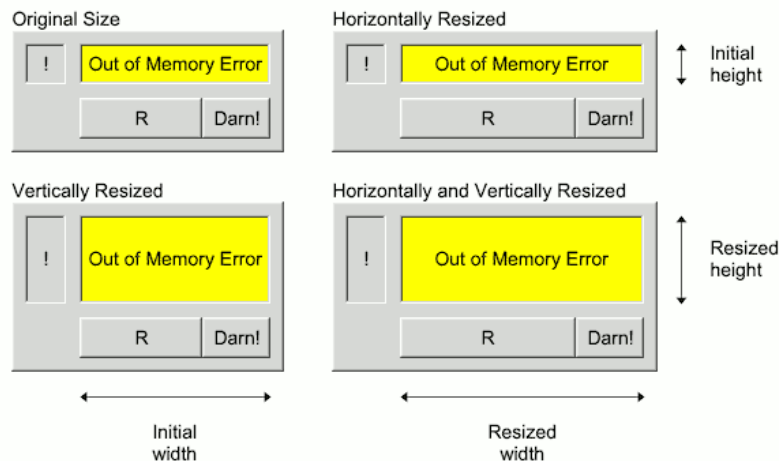


Figure 1.19 Resizing dialog example (c)

Now the invisible box, shown as "R", takes all of the horizontal resizing and the "Darn!" box will stay as it is. Here's the skeleton code:

```
dialog = new Fl_Window(300, 100);
icon = new Fl_Box(0, 0, 50, 50, "!");
text = new Fl_Box(50, 0, 250, 40, "Out of Memory Error");
btns = new Fl_Group(50, 50, 250, 50); // parent group
darn = new Fl_Button(200, 50, 100, 50, "Darn!");
R = new Fl_Box(50, 50, 150, 50); // "invisible" box "R"
R->hide(); // make sure it's invisible
btns->resizable(R); // make "R" parent group resizable
btns->end();
dialog->resizable(text);
dialog->end();
```

Imagine instead that you have a group that has a button, an input field, another button and a second input field, all next to each other, and you want the input fields to resize equally, but not the buttons. How could you achieve this?

Setting either of the input fields to be the `resizable` leaves the other one fixed, as shown below:

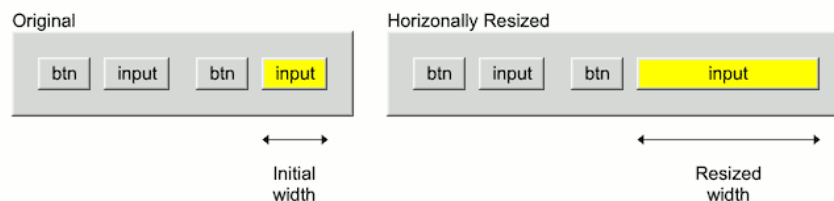


Figure 1.20 Resizing input fields example (b)

The answer is to leave the `resizable` of the group set to itself, and to create two equal size subgroups, each of which will resize equally. Add a button and input field to each subgroup, and set each subgroup's `resizable` to the input field, as shown below. Tada!

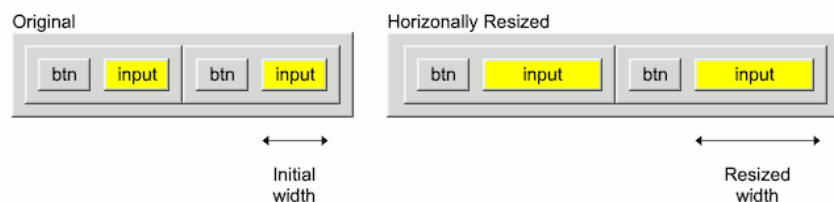


Figure 1.21 Resizing input fields example (b)

In FLTK it is possible to solve almost any layout and resizing problem by introducing an invisible box into a group, or an extra group into the widget hierarchy. It might take some thought to achieve exactly what you want and sometimes it is necessary to introduce parallel hierarchies in order to get widgets in different groups to resize together.

Imagine you have a group containing three widgets in a row, and you want the widget in the middle to stay the same size when the group is stretched and the ones on either side and the padding between them to resize symmetrically. As described earlier, the default resizing behavior for a group results in proportional resizing of the child widgets (and also of the margins and padding between them) as shown below, which is clearly not what you want.

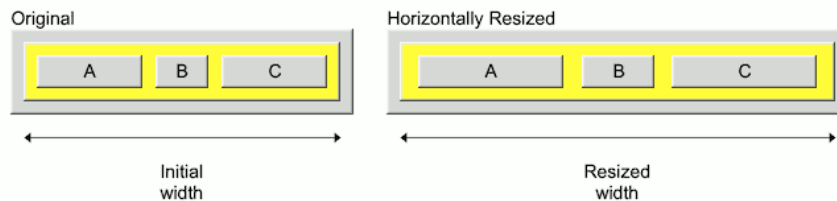


Figure 1.22 Resizing a row of widgets (a)

Simply adding a group around A and B and setting its `resizable` to A, as in the previous `btn-input` example, will mean that B stays the same size, but the other widgets won't resize symmetrically, so what else is needed? It isn't immediately obvious how to solve this problem, even for experienced FLTK users. This is possibly because users are generally advised to design widgets so that they don't overlap.

Albrecht Schlosser proposed an innovative technique that involves an invisible box that deliberately overlaps others to achieve the desired behavior. For the current example, this means inserting two new groups into the existing group and adding a hidden resizable widget.

The first group, shown in red below, extends from the left edge of the parent group to the middle of the gap between boxes B and C on the right. This first group contains boxes A and B, where A is the first group's `resizable` attribute.

The second group, shown in blue, extends from the right edge of the first group to the right edge of the parent group. This second group contains box C, where C is the second group's `resizable`.

The extra box widget is added to the parent group and is set as the group's `resizable`. The three `resizable` widgets are shown in yellow.

The clever bit is that this extra box widget is not horizontally aligned with any of the existing groups and widgets in the usual way, but instead overlaps the right and left parts of the two new groups by the same small amount, which means that its midpoint is aligned with the edge between the groups.

Note that, for clarity, the height of the original group has been increased to allow space for the additional annotation and to highlight the extra resizable box in the extra space at the bottom of the group. This is fine for the horizontal-only resizing shown here, but means that widgets A, B and C will never change height because the extra resizable box does not overlap them vertically. Only the padding below them will be resized.

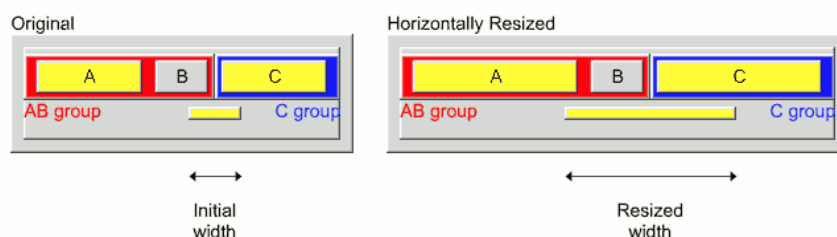


Figure 1.23 Resizing a row of widgets (b)

In a real application, you probably want to allow widgets A, B and C to resize vertically while the height of any padding or widgets above or below remains fixed, so the extra resizable box has to lie within the height of widgets A, B and C. Obviously after calling `hide()` on the box it is no longer visible, and may therefore be the same height as the other widgets, or a fraction of the height, as shown below.

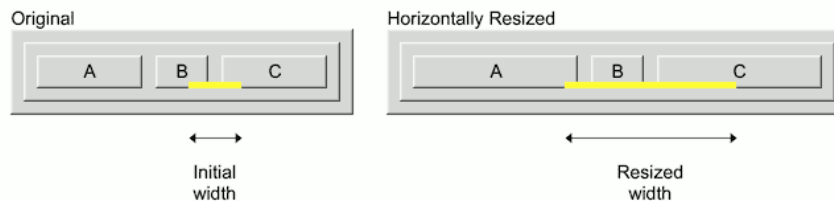


Figure 1.24 Resizing a row of widgets (c)

To summarize the key points of the new technique:

- The new resizable widget must overlap the widgets on each side by exactly the same amount.
- The width of the new resizable widget is not fixed, but should probably be a relatively small value to avoid potential problems.
- The total width of the two new groups must equal the width of the existing group and there can be no offsets or gaps between them because margins and gaps will affect the resizing behavior.
- The same principles apply to vertical resizing.

1.7 Designing a Simple Text Editor

This chapter guides you through the design of a simple FLTK-based text editor. The complete source code for our text editor can be found in the `test/editor.cxx` file.

The tutorial comprises multiple chapters, and you can activate the relevant code by adjusting the `TUTORIAL_↔CHAPTER` macro at the top of the source file to match the chapter number.

Each chapter builds on the previous one. The documentation, as well as the source code, can be read sequentially, maintaining a consistent program structure while introducing additional features step by step.

Note

The tutorial uses several global variables for brevity. Additionally, the order of code blocks is rather uncommon but helps to keep related features within a chapter.

1.7.1 Determining the Goals of the Text Editor

As our first step, we define what we want our text editor to do:

1. Edit a single text document.
2. Provide a menubar/menus for all functions.
3. Load from a file.
4. Save to a file.
5. Keep track of when the file has been changed.
6. Cut/copy/delete/paste menus.
7. Search and replace functionality.
8. Multiple views of the same text.
9. "C" language syntax highlighting.

1.7.2 Chapter 1: A Minimal App

Let's ensure that we can set up our build process to compile and verify our code as we add features. We begin by writing a minimal program with no other purpose than opening a window.

The code for that is barely longer than a "Hello, world" program and is marked in the source code as `TUTORIAL`↔

```
_CHAPTER = 1.
#include <FL/Fl_Double_Window.H>
#include <FL/Fl.H>

Fl_Double_Window *app_window = NULL;

void tut1_build_app_window() {
    app_window = new Fl_Double_Window(640, 480, "FLTK Editor");
}

int main (int argc, char **argv) {
    tut1_build_app_window();
    app_window->show(argc, argv);
    return Fl::run();
}
```

Passing `argc` and `argv` to `Fl_Double_Window::show()` allows FLTK to parse command line options, providing the user with the ability to change the color or graphical scheme of the editor at launch time.

`Fl::run()` will return when no more windows in the app are visible. In other words, if all windows in an app are closed, hidden, or deleted. Pressing "Escape" or clicking the "Close" button in the window frame will close our only window, prompting `Fl::run()` to return, effectively ending the app.

When building FLTK from source, the CMake environment includes the necessary rules to build the editor. You can find more information on how to write your own CMake files in the `README.CMake.txt` text in the top FLTK directory.

For Linux and macOS, FLTK comes with the `fltk-config` script that generates the compiler commands for you: `fltk-config --compile editor.cxx`

If the code compiles and links correctly, running the app will pop up an empty application window on the desktop screen. You can close the window and quit the app by pressing the 'Escape' key or by clicking the "Close" button in the window frame.

Congratulations, you've just built a minimal FLTK app.

1.7.3 Chapter 2: Adding a Menu Bar

In this chapter, we will handle the window title and add the main menu bar with a File menu and a Quit button.

We need to declare a variable to track changes in the text, and a buffer for the current filename.

`// remove `main()` from chapter 1, but keep the rest of the code, then add...`

```
#include <FL/Fl_Menu_Bar.H>
#include <FL/fl_ask.H>
#include <FL/filename.H>
#include <FL/fl_string_functions.h>

Fl_Menu_Bar *app_menu_bar = NULL;
bool text_changed = false;
char app_filename[FL_PATH_MAX] = "";
```

The window title is either "FLTK Editor" if the text is not saved in any file, or the filename, followed by an * if the text changed. Note that we have two ways to set the label of a widget. `label()` will link a static text, and `copy_label()` which will copy and manage the label text.

```
void update_title() {
    const char *fname = NULL;
    if (app_filename[0])
        fname = fl_filename_name(app_filename);
    if (fname) {
        char buf[FL_PATH_MAX + 3];
```

```

    buf[FL_PATH_MAX + 2] = '\0'; // ensure that the buffer is always terminated
    if (text_changed) {
        snprintf(buf, FL_PATH_MAX+2, "%s *", fname);
    } else {
        snprintf(buf, FL_PATH_MAX+2, "%s", fname);
    }
    app_window->copy_label(buf);
} else {
    app_window->label("FLTK Editor");
}
}

```

Now instead of writing directly to `text_changed`, we write a function that can set and clear the flag, and update the title accordingly.

```

void set_changed(bool v) {
    if (v != text_changed) {
        text_changed = v;
        update_title();
    }
}

```

Let's do the same for changing the filename. If the new filename is NULL, the window title will revert to "FLTK Editor".

```

void set_filename(const char *new_filename) {
    if (new_filename) {
        fl_strncpy(app_filename, new_filename, FL_PATH_MAX);
    } else {
        app_filename[0] = 0;
    }
    update_title();
}

```

But enough of managing window titles. The following code will add the first widget to our window. A menubar is created at the top and all across the main window.

```

void menu_quit_callback(Fl_Widget *, void *) { /* TODO */ }

void tut2_build_app_menu_bar() {
    app_window->begin();
    app_menu_bar = new Fl_Menu_Bar(0, 0, app_window->w(), 25);
    app_menu_bar->add("File/Quit Editor", FL_COMMAND+'q', menu_quit_callback);
    app_window->callback(menu_quit_callback);
    app_window->end();
}

int main (int argc, char **argv) {
    tut1_build_app_window();
    tut2_build_app_menu_bar();
    app_window->show(argc, argv);
    return Fl::run();
}

```

`begin()` tells FLTK to add all widgets created hereafter to our `app_window`. In this particular case, it is redundant because creating the window in the previous chapter already called `begin()` for us.

In the next line, we create the menu bar and add our first menu item to it. Menus can be constructed like file paths, with forward slashes '/' separating submenus from menu items.

Our basic callback is simple:

```

void menu_quit_callback(Fl_Widget *, void *) {
    Fl::hide_all_windows();
}

```

`Fl::hide_all_windows()` will make all windows invisible, causing `Fl::run()` to return and `main` to exit.

The next line, `app_window->callback(menu_quit_callback)` links the same `menu_quit_callback` to the `app_window` as well. Assigning the window callback removes the default "Escape" key handling and allows the `menu_quit_callback` to handle that keypress with a friendly dialog box instead of just quitting the app.

The `Fl_Widget*` parameter in the callback will either be `app_window` if called through the window callback, or `app_menu_bar` if called by one of the menu items.

One of our goals was to keep track of text changes. If we know the text changed and is unsaved, we should notify the user that she is about to lose her work. We achieve this by adding a dialog box in the Quit callback that queries if the user really wants to quit, even if text was changed:

```
void menu_quit_callback(Fl_Widget *, void *) {
    if (text_changed) {
        int c = fl_choice("Changes in your text have not been saved.\n"
                          "Do you want to quit the editor anyway?",
                          "Quit", "Cancel", NULL);
        if (c == 1) return;
    }
    Fl::hide_all_windows();
}
```

1.7.4 Chapter 3: Adding a Text Editor widget

FLTK comes with a pretty capable builtin text editing widget. We will use this `Fl_Text_Editor` widget here to allow users to edit their documents.

`Fl_Text_Editor` needs an `Fl_Text_Buffer` to do anything useful. What might seem like an unnecessary extra step is a great feature: we can assign one text buffer to multiple text editors. In a later chapter, we will use this feature to implement a split editor window.

```
#include <FL/Fl_Text_Buffer.H>
#include <FL/Fl_Text_Editor.H>

Fl_Text_Editor *app_editor = NULL;
Fl_Text_Editor *app_split_editor = NULL; // for later
Fl_Text_Buffer *app_text_buffer = NULL;

// ... callbacks go here

void tut3_build_main_editor() {
    app_window->begin();
    app_text_buffer = new Fl_Text_Buffer();
    app_text_buffer->add_modify_callback(text_changed_callback, NULL);
    app_editor = new Fl_Text_Editor(0, app_menu_bar->h(),
    app_window->w(), app_window->h() - app_menu_bar->h());
    app_editor->buffer(app_text_buffer);
    app_editor->textfont(FL_COURIER);
    app_window->resizable(app_editor);
    app_window->end();
}
```

By setting the `app_editor` to be the `resizable()` property of `app_window`, we make our application window resizable on the desktop, and we ensure that resizing the window will only resize the text editor vertically, but not our menu bar.

To keep track of changes to the document, we add a callback to the text editor that will be called whenever text is added or deleted. The text modify callback sets our `text_changed` flag if text was changed:

```
// insert before tut3_build_main_editor()
void text_changed_callback(int, int n_inserted, int n_deleted, int, const char*, void*) {
    if (n_inserted || n_deleted)
        set_changed(true);
}
```

To wrap this chapter up, we add a "File/New" menu and link it to a callback that clears the text buffer, clears the current filename, and marks the buffer as unchanged.

```
// insert before tut3_build_main_editor()
void menu_new_callback(Fl_Widget*, void*) {
    app_text_buffer->text("");
    set_changed(false);
}

// insert at the end of tut3_build_main_editor()
...
// find the Quit menu and insert the New menu there
int ix = app_menu_bar->find_index(menu_quit_callback);
app_menu_bar->insert(ix, "New", FL_COMMAND+'n', menu_new_callback);
...
```

1.7.5 Chapter 4: Reading and Writing Files

In this chapter, we will add support for loading and saving text files, so we need three more menu items in the File menu: Open, Save, and Save As.

```
#include <FL/Fl_Native_File_Chooser.H>
#include <FL/platform.H>
#include <errno.h>

// ... add callbacks here

void tut4_add_file_support() {
    int ix = app_menu_bar->find_index(menu_quit_callback);
    app_menu_bar->insert(ix, "Open", FL_COMMAND+'o', menu_open_callback, NULL, FL_MENU_DIVIDER);
    app_menu_bar->insert(ix+1, "Save", FL_COMMAND+'s', menu_save_callback);
    app_menu_bar->insert(ix+2, "Save as...", FL_COMMAND+'S', menu_save_as_callback, NULL, FL_MENU_DIVIDER);
}
```

Note

The menu shortcuts `FL_COMMAND+'s'` and `FL_COMMAND+'S'` look the same at a first glance, but the second shortcut is actually `Ctrl-Shift-S` due to the capital letter 'S'. Also, we use `FL_COMMAND` as our menu shortcut modifier key. `FL_COMMAND` translates to `FL_CTRL` on Windows and Linux, and to `FL_META` on macOS, better known as the cloverleaf, or simply "the Apple key".

We implement the Save As callback first, because we will want to call it from the Open callback later. The basic callback is only a few lines of code.

```
void menu_save_as_callback(Fl_Widget*, void*) {
    Fl_Native_File_Chooser file_chooser;
    file_chooser.title("Save File As...");
    file_chooser.type(Fl_Native_File_Chooser::BROWSE_SAVE_FILE);
    if (file_chooser.show() == 0) {
        app_text_buffer->savefile(file_chooser.filename());
        set_filename(file_chooser.filename());
        set_changed(false);
    }
}
```

However if the user has already set a file name including path information, it is the polite thing to preload the file chooser with that information. This little chunk of code will separate the file name from the path before we call `file_chooser.show()`:

```
// insert before `if (file_chooser.show())...`
if (app_filename[0]) {
    char temp_filename[FL_PATH_MAX];
    fl_strlcpy(temp_filename, app_filename, FL_PATH_MAX);
    const char *name = fl_filename_name(temp_filename);
    if (name) {
        file_chooser.preset_file(name);
        temp_filename[name - temp_filename] = 0;
        file_chooser.directory(temp_filename);
    }
}
```

Great. Now let's add code for our File/Save menu. If no filename was set yet, it falls back to our Save As callback. `Fl_Text_Editor::savefile()` writes the contents of our text widget into a UTF-8 encoded text file.

```
void menu_save_callback(Fl_Widget*, void*) {
    if (!app_filename[0]) {
        menu_save_as_callback(NULL, NULL);
    } else {
        app_text_buffer->savefile(file_chooser.filename());
        set_changed(false);
    }
}
```

Now that we have a save method available, we can improve the `menu_quit_callback` and offer the option to save the current modified text before quitting the app. Here is the new quit callback code that replaces the old callback:

```
void menu_quit_callback(Fl_Widget *, void *) {
    if (text_changed) {
        int r = fl_choice("The current file has not been saved.\n"
                        "Would you like to save it now?",
                        "Cancel", "Save", "Don't Save");
        if (r == 0) // cancel
            return;
        if (r == 1) { // save
```

```

        menu_save_callback(NULL, NULL);
        return;
    }
}
Fl::hide_all_windows();
}

```

On to loading a new file. Let's write the function to load a file from a given file name:

```

void load(const char *filename) {
    if (app_text_buffer->loadfile(filename) == 0) {
        set_filename(filename);
        set_changed(false);
    }
}

```

A friendly app should warn the user if file operations fail. This can be done in three lines of code, so let's add an alert dialog after every loadfile and savefile call. This is exemplary for load(), and the code is very similar for the two other locations.

```

void load(const char *filename) {
    if (app_text_buffer->loadfile(filename) == 0) {
        set_filename(filename);
        set_changed(false);
    } else {
        fl_alert("Failed to load file\n%s\n%s",
                filename,
                strerror(errno));
    }
}

```

If the user selects our pulldown "Load" menu, we first check if the current text was modified and provide a dialog box that offers to save the changes before loading a new text file:

```

void menu_open_callback(Fl_Widget*, void*) {
    if (text_changed) {
        int r = fl_choice("The current file has not been saved.\n"
                        "Would you like to save it now?",
                        "Cancel", "Save", "Don't Save");

        if (r == 2)
            return;
        if (r == 1)
            menu_save_callback();
    }
    ...
}

```

If the user did not cancel the operation, we pop up a file chooser for loading the file, using similar code as in Save As.

```

...
Fl_Native_File_Chooser file_chooser;
file_chooser.title("Open File...");
file_chooser.type(Fl_Native_File_Chooser::BROWSE_FILE);
...

```

Again, we preload the file chooser with the last used path and file name:

```

...
if (app_filename[0]) {
    char temp_filename[FL_PATH_MAX];
    fl_strlcpy(temp_filename, app_filename, FL_PATH_MAX);
    const char *name = fl_filename_name(temp_filename);
    if (name) {
        file_chooser.preset_file(name);
        temp_filename[name - temp_filename] = 0;
        file_chooser.directory(temp_filename);
    }
}
...

```

And finally, we pop up the file chooser. If the user cancels the file dialog, we do nothing and keep the current file. Otherwise, we call the load() function that we already wrote:

```

if (file_chooser.show() == 0)
    load(file_chooser.filename());
}

```

We really should support two more ways to load documents from a file. Let's modify the "show and run" part of main() to handle command line parameters and desktop drag'n'drop operations. For that, we refactor the last two lines of main() into a new function:

```

// ... new function here

```

```

int main (int argc, char **argv) {

```

```

tut1_build_app_window();
tut2_build_app_menu_bar();
tut3_build_main_editor();
tut4_add_file_support();
// ... refactor those into the new function
// app_window->show(argc, argv);
// return Fl::run();
return tut4_handle_commandline_and_run(argc, argv);
}

```

Our function to show the window and run the app has a few lines of boilerplate code. `Fl::args_to_utf8()` converts the command line argument from whatever the host system provides into Unicode. `Fl::args()` goes through the list of arguments and gives `args_handler()` a chance to handle each argument. It also makes sure that FLTK specific args are still forwarded to FLTK, so `"-scheme plastic"` and `"-background #aaccff"` will draw beautiful blue buttons in a plastic look.

`fl_open_callback()` lets FLTK know what to do if a user drops a text file onto our editor icon (Apple macOS). Here, we ask it to call the `load()` function that we wrote earlier.

```

// ... args_handler here

int tut4_handle_commandline_and_run(int &argc, char **argv) {
    int i = 0;
    Fl::args_to_utf8(argc, argv);
    Fl::args(argc, argv, i, args_handler);
    fl_open_callback(load);
    app_window->show(argc, argv);
    return Fl::run();
}

```

Last work item for this long chapter: what should our `args_handler` do? We could handle additional command line options here, but for now, all we want to handle is file names and paths. Let's make this easy: if the current arg does not start with a '-', we assume it is a file name, and we call `load()`:

```

int args_handler(int argc, char **argv, int &i) {
    if (argv[i] && argv[i][0] != '-') {
        load(argv[i]);
        i++;
        return 1;
    }
    return 0;
}

```

So this is our basic but quite functional text editor app in about 100 lines of code. The following chapters add some user convenience functions and show off some FLTK features including split editors and syntax highlighting.

1.7.6 Chapter 5: Cut, Copy, and Paste

The FLTK Text Editor widget comes with builtin cut, copy, and paste functionality, but as a courtesy, we should also offer these as menu items in the main menu.

In our feature list, we noted that we want to implement a split text editor. This requires that the callbacks know which text editor has the keyboard focus. Calling `Fl::focus()` may return `NULL` or other unknown widgets, so we add a little test in our callbacks:

```

void menu_cut_callback(Fl_Widget*, void* v) {
    Fl_Widget *e = Fl::focus();
    if (e && (e == app_editor || e == app_split_editor))
        Fl_Text_Editor::kf_cut(0, (Fl_Text_Editor*)e);
}

```

We can write very similar callbacks for undo, redo, copy, paste, and delete. Adding a new menu and the six menu items follows the same pattern as before. Using the Menu/Item notation will create an Edit menu for us:

```

void tut5_cut_copy_paste() {
    app_menu_bar->add("Edit/Undo", FL_COMMAND+'z', menu_undo_callback);
    app_menu_bar->add("Edit/Redo", FL_COMMAND+'Z', menu_redo_callback, NULL, FL_MENU_DIVIDER);
    app_menu_bar->add("Edit/Cut", FL_COMMAND+'x', menu_cut_callback);
    app_menu_bar->add("Edit/Copy", FL_COMMAND+'c', menu_copy_callback);
    app_menu_bar->add("Edit/Paste", FL_COMMAND+'v', menu_paste_callback);
    app_menu_bar->add("Edit/Delete", 0, menu_delete_callback);
}

```


1.7.7 Chapter 6: Find and Find Next

Corporate called. They want a dialog box for their users that can search for some word in the text file. We can add this functionality using a callback and a standard FLTK dialog box.

Here is some code to find a string in a text editor. The first four lines make sure that we start our search at the cursor position of the current editor window. The rest of the code searches the string and marks it if found.

```
void find_next(const char *needle) {
    Fl_Text_Editor *editor = app_editor;
    Fl_Widget *e = Fl::focus();
    if (e && e == app_split_editor)
        editor = app_split_editor;
    int pos = editor->insert_position();
    int found = app_text_buffer->search_forward(pos, needle, &pos);
    if (found) {
        app_text_buffer->select(pos, pos + (int)strlen(needle));
        editor->insert_position(pos + (int)strlen(needle));
        editor->show_insert_position();
    } else {
        fl_alert("No further occurrences of '%s' found!", needle);
    }
}
```

The callbacks are short, using the FLTK text field dialog box and the `find_next` function that we already implemented. The last searched text is saved in `last_find_text` to be reused by `menu_find_next_callback`. If no search text was set yet, or it was set to an empty text, "Find Next" will forward to `menu_find_callback` and pop up our "Find Text" dialog.

```
char last_find_text[1024] = "";

void menu_find_callback(Fl_Widget*, void* v) {
    const char *find_text = fl_input("Find in text:", last_find_text);
    if (find_text) {
        fl_strlcpy(last_find_text, find_text, sizeof(last_find_text));
        find_next(find_text);
    }
}

void menu_find_next_callback(Fl_Widget*, void* v) {
    if (last_find_text[0]) {
        find_next(last_find_text);
    } else {
        menu_find_callback(NULL, NULL);
    }
}
```

And of course we need to add two menu items to our main application menu.

```
...
app_menu_bar->add("Find/Find...", FL_COMMAND+'F', menu_find_callback);
app_menu_bar->add("Find/Find Next", FL_COMMAND+'g', menu_find_next_callback, NULL, FL_MENU_DIVIDER);
...
```

1.7.8 Chapter 7: Replace and Replace Next

To implement the next feature, we will need to implement our own "Find and Replace" dialog box. To make this dialog box useful, it needs the following elements:

- a text input field for the text that we want to find
- a text input field for the replacement text
- a button to find the next occurrence
- a button to replace the current text and find the next occurrence
- a button to close the dialog

This is rather complex functionality, so instead of adding more global variables, we will pack this dialog into a class, derived from `Fl_Window`.

Note

The tutorial uses `Fl_Double_Window` instead of `Fl_Window` throughout. Historically, on some platforms, `Fl_Window` renders faster, but has a tendency to flicker. In today's world, this has very little relevance and FLTK optimizes both window types. `Fl_Double_Window` is recommended unless there is a specific reason to use `Fl_Window`.

Let's implement the text replacement code first:

```
char last_replace_text[1024] = "";

void replace_selection(const char *new_text) {
    Fl_Text_Editor *editor = app_editor;
    Fl_Widget *e = Fl::focus();
    if (e && e == app_split_editor)
        editor = app_split_editor;
    int start, end;
    if (app_text_buffer->selection_position(&start, &end)) {
        app_text_buffer->remove_selection();
        app_text_buffer->insert(start, new_text);
        app_text_buffer->select(start, start + (int)strlen(new_text));
        editor->insert_position(start + (int)strlen(new_text));
        editor->show_insert_position();
    }
}
```

As before, the first four lines anticipate a split editor and find the editor that has focus. The code then deletes the currently selected text, replaces it with the new text, selects the new text, and finally sets the text cursor to the end of the new text.

The Replace_Dialog class

The `Replace_Dialog` class holds pointers to our active UI elements as well as all the callbacks for the dialog buttons.

```
class Replace_Dialog : public Fl_Double_Window {
    Fl_Input *find_text_input;
    Fl_Input *replace_text_input;
    Fl_Button *find_next_button;
    Fl_Button *replace_and_find_button;
    Fl_Button *close_button;
public:
    Replace_Dialog(const char *label);
    void show() FL_OVERRIDE;
private:
    static void find_next_callback(Fl_Widget*, void*);
    static void replace_and_find_callback(Fl_Widget*, void*);
    static void close_callback(Fl_Widget*, void*);
};

Replace_Dialog *replace_dialog = NULL;
```

The constructor creates the dialog and marks it as "non modal". This will make the dialog hover over the application window like a toolbox window until the user closes it, allowing multiple "find and replace" operations. So here is our constructor:

```
Replace_Dialog::Replace_Dialog(const char *label)
: Fl_Double_Window(430, 110, label)
{
    find_text_input = new Fl_Input(100, 10, 320, 25, "Find:");
    replace_text_input = new Fl_Input(100, 40, 320, 25, "Replace:");
    Fl_Flex* button_field = new Fl_Flex(100, 70, w()-100, 40);
    button_field->type(Fl_Flex::HORIZONTAL);
    button_field->margin(0, 5, 10, 10);
    button_field->gap(10);
    find_next_button = new Fl_Button(0, 0, 0, 0, "Next");
    find_next_button->callback(find_next_callback, this);
    replace_and_find_button = new Fl_Button(0, 0, 0, 0, "Replace");
    replace_and_find_button->callback(replace_and_find_callback, this);
    close_button = new Fl_Button(0, 0, 0, 0, "Close");
    close_button->callback(close_callback, this);
    button_field->end();
    set_non_modal();
}
```

All buttons are created inside an `Fl_Flex` group. They will be arranged automatically by `Fl_Flex`, so there is no need to set x and y coordinates or a width or height. `button_field` will lay out the buttons for us.

Note

There is no need to write a destructor or delete individual widgets. When we delete an instance of `ReplaceDialog`, all children are deleted for us.

The `show()` method overrides the window's `show` method. It adds some code to preload the values of the text fields for added convenience. It then pops up the dialog box by calling the original `Fl_Double_Window::show()`.

```
void ReplaceDialog::show() {
    find_text_input->value(last_find_text);
    replace_text_input->value(last_replace_text);
    Fl_Double_Window::show();
}
```

The buttons in the dialog need callbacks to be useful. If callbacks are defined within a class, they must be defined `static`, but a pointer to the class can be provided through the `user_data` field. We have done that in the constructor by adding `this` as the last argument when setting the callback, for example in `close_button->callback(close_callback, this);`.

The callback itself can then extract the `this` pointer with a static cast:

```
void ReplaceDialog::close_callback(Fl_Widget*, void* my_dialog) {
    ReplaceDialog *dlg = static_cast<ReplaceDialog*>(my_dialog);
    dlg->hide();
}
```

The callback for the Find button uses our already implemented `find_next` function:

```
void ReplaceDialog::find_next_callback(Fl_Widget*, void* my_dialog) {
    ReplaceDialog *dlg = static_cast<ReplaceDialog*>(my_dialog);
    fl_strlcpy(last_find_text, dlg->find_text_input->value(), sizeof(last_find_text));
    fl_strlcpy(last_replace_text, dlg->replace_text_input->value(), sizeof(last_replace_text));
    if (last_find_text[0])
        find_next(last_find_text);
}
```

The Replace button callback calls our newly implemented `replace_selection` function and then continues on to the `find_next_callback`:

```
void ReplaceDialog::replace_and_find_callback(Fl_Widget*, void* my_dialog) {
    ReplaceDialog *dlg = static_cast<ReplaceDialog*>(my_dialog);
    replace_selection(dlg->replace_text_input->value());
    find_next_callback(NULL, my_dialog);
}
```

This long chapter comes close to its end. We are missing menu items that pop up our dialog and that allow a quick "Replace and Find Next" functionality without popping up the dialog. The code is quite similar to the "Find" and "Find Next" code in the previous chapter:

```
void menu_replace_callback(Fl_Widget*, void*) {
    if (!replace_dialog)
        replace_dialog = new ReplaceDialog("Find and Replace");
    replace_dialog->show();
}

void menu_replace_next_callback(Fl_Widget*, void*) {
    if (!last_find_text[0]) {
        menu_replace_callback(NULL, NULL);
    } else {
        replace_selection(last_replace_text);
        find_next(last_find_text);
    }
}

void tut7_implement_replace() {
    app_menu_bar->add("Find/Replace...", FL_COMMAND+'r', menu_replace_callback);
    app_menu_bar->add("Find/Replace Next", FL_COMMAND+'t', menu_replace_next_callback);
}
```

1.7.9 Chapter 8: Editor Features

Chapter 7 was long an intense. Let's relax and implement something simple here. We want menus with check boxes that can toggle some text editor features on and off:

```
void tut8_editor_features() {
    app_menu_bar->add("Window/Line Numbers", FL_COMMAND+'l', menu_linenumbers_callback, NULL, FL_MENU_TOGGLE);
    app_menu_bar->add("Window/Word Wrap", 0, menu_wordwrap_callback, NULL, FL_MENU_TOGGLE);
}
```

```
}
```

The `Fl_Widget` parameter in callbacks always points to the widget that causes the callback. Menu items are not derived from widgets, so to find out which menu item caused a callback, we can do this:

```
void menu_linenumbers_callback(Fl_Widget* w, void*) {
    Fl_Menu_Bar* menu = static_cast<Fl_Menu_Bar*>(w);
    const Fl_Menu_Item* linenumber_item = menu->mvalue();
    if (linenumber_item->value()) {
        app_editor->linenumber_width(40);
    } else {
        app_editor->linenumber_width(0);
    }
    app_editor->redraw();
}
```

Setting the width enables the line numbers, setting it to 0 disables the line number display. When changing the value of a widget, FLTK will make sure that the widget is redrawn to reflect the new value. When changing other attributes such as colors or fonts, FLTK assumes that many attributes are changed at the same time and leaves it to the user to call `Fl_Widget::redraw()` when done. Here we call `app_editor->redraw()` to make sure that the change in the line number setting is also drawn on screen.

Let's not forget to update the line number display for a potential split editor widget as well:

```
// add before the end of menu_linenumbers_callback
if (app_split_editor) {
    if (linenumber_item->value()) {
        app_split_editor->linenumber_width(40);
    } else {
        app_split_editor->linenumber_width(0);
    }
    app_split_editor->redraw();
}
```

The word wrap feature is activated by calling `Fl_Text_Editor::wrap_mode()` with the parameters `Fl_Text_Display::WRAP_AT_BOUNDS` and 0. It's deactivated with `Fl_Text_Display::WRAP_NONE`. The implementation of the callback is the same as `menu_linenumbers_callback`.

1.7.10 Chapter 9: Split Editor

When editing long source code files, it can be really helpful to split the editor to view statements at the top of the text while adding features at the bottom of the text in a split text view.

FLTK can link multiple text editors to a single text buffer. Let's implement this now. This chapter will show you how to rearrange widgets in an existing widget tree.

Our initializer removes the main text editor from the widget tree and replaces it with an `Fl_Tile`. A tile can hold multiple widgets that can then be resized interactively by the user by clicking and dragging the divider between those widgets.

We start by replacing the editor widget with a tile group of the same size.

```
#include <FL/Fl_Tile.H>

Fl_Tile *app_tile = NULL;

void tut9_split_editor() {
    app_window->begin();
    app_tile = new Fl_Tile(app_editor->x(), app_editor->y(),
                          app_editor->w(), app_editor->h());
    app_window->remove(app_editor);
}
```

Next we add our existing editor as the first child of the tile and create another text editor `app_split_editor` as the second child of the tile, but it's hidden for now with a height of zero pixels.

Note

Creating the new `Fl_Tile` also calls `Fl_Tile::begin()`.

Adding `app_editor` to the tile would have also removed it from `app_window`, so `app_window->remove(app_editor)` in the code above is not really needed, but illustrates what we are doing.

```
app_tile->add(app_editor);
app_split_editor = new Fl_Text_Editor(app_tile->x(), app_tile->y()+app_tile->h(),
                                     app_tile->w(), 0);
app_split_editor->buffer(app_text_buffer);
app_split_editor->textfont(FL_COURIER);
app_split_editor->hide();
```

Now we clean up after ourselves and make sure that the resizables are all set correctly. Lastly, we add a menu item with a callback.

```
app_tile->end();
app_tile->size_range(0, 25, 25);
app_tile->size_range(1, 25, 25);
app_window->end();
app_window->resizable(app_tile);
app_tile->resizable(app_editor);
app_menu_bar->add("Window/Split", FL_COMMAND+'i', menu_split_callback, NULL, FL_MENU_TOGGLE);
}
```

Now with all widgets in place, the callback's job is to show and resize, or hide and resize the split editor. We can implement that like here:

```
void menu_split_callback(Fl_Widget* w, void*) {
    Fl_Menu_Bar* menu = static_cast<Fl_Menu_Bar*>(w);
    const Fl_Menu_Item* splitview_item = menu->mvalue();
    if (splitview_item->value()) {
        int h_split = app_tile->h()/2;
        app_editor->size(app_tile->w(), h_split);
        app_split_editor->resize(app_tile->x(), app_tile->y() + h_split,
                               app_tile->w(), app_tile->h() - h_split);
        app_split_editor->show();
    } else {
        app_editor->size(app_tile->w(), app_tile->h());
        app_split_editor->resize(app_tile->x(), app_tile->y()+app_tile->h(),
                               app_tile->w(), 0);
        app_split_editor->hide();
    }
    app_tile->resizable(app_editor);
    app_tile->init_sizes();
    app_tile->redraw();
}
```

1.7.11 Chapter 10: Syntax Highlighting

Chapter 10 adds a lot of code to implement "C" language syntax highlighting. Not all code is duplicated here in the documentation. Please check out `test/editor.cxx` for all the details.

The `Fl_Text_Editor` widget supports highlighting of text with different fonts, colors, and sizes. The implementation is based on the excellent `NEdit` text editor core, from <https://sourceforge.net/projects/nedit/>, which uses a parallel "style" buffer which tracks the font, color, and size of the text that is drawn.

Styles are defined using the `Fl_Text_Display::Style_Table_Entry` structure defined in `<FL/Fl_Text_Display.H>`:

```
:
struct Style_Table_Entry {
    Fl_Color color;
    Fl_Font font;
    int size;
    unsigned attr;
};
```

The `color` member sets the color for the text, the `font` member sets the FLTK font index to use, and the `size` member sets the pixel size of the text. The `attr` member is currently not used.

For our text editor we'll define 7 styles for plain code, comments, keywords, and preprocessor directives:

```
Fl_Text_Display::Style_Table_Entry styletable[] = { // Style table
```

```

{ FL_BLACK,      FL_COURIER,      FL_NORMAL_SIZE }, // A - Plain
{ FL_DARK_GREEN, FL_COURIER_ITALIC, FL_NORMAL_SIZE }, // B - Line comments
{ FL_DARK_GREEN, FL_COURIER_ITALIC, FL_NORMAL_SIZE }, // C - Block comments
{ FL_BLUE,       FL_COURIER,      FL_NORMAL_SIZE }, // D - Strings
{ FL_DARK_RED,   FL_COURIER,      FL_NORMAL_SIZE }, // E - Directives
{ FL_DARK_RED,   FL_COURIER_BOLD, FL_NORMAL_SIZE }, // F - Types
{ FL_BLUE,       FL_COURIER_BOLD, FL_NORMAL_SIZE }, // G - Keywords
};

```

You'll notice that the comments show a letter next to each style - each style in the style buffer is referenced using a character starting with the letter 'A'.

You call the `highlight_data()` method to associate the style data and buffer with the text editor widget:

```
Fl_Text_Buffer *app_style_buffer;
```

```

app_editor->highlight_data(app_style_buffer, styletable,
                          sizeof(styletable) / sizeof(styletable[0]),
                          'A', style_unfinished_cb, 0);

```

Finally, you need to add a callback to the main text buffer so that changes to the text buffer are mirrored in the style buffer:

```
app_text_buffer->add_modify_callback(style_update, app_editor);
```

The `style_update()` function, like the `change_cb()` function described earlier, is called whenever text is added or removed from the text buffer. It mirrors the changes in the style buffer and then updates the style data as necessary:

```

//
// 'style_update()' - Update the style buffer...
//

void
style_update(int      pos,          // I - Position of update
             int      nInserted,    // I - Number of inserted chars
             int      nDeleted,     // I - Number of deleted chars
             int      nRestyled,    // I - Number of restyled chars
             const char *deletedText, // I - Text that was deleted
             void      *cbArg) {    // I - Callback data
    int start,                      // Start of text
        end;                       // End of text
    char last,                     // Last style on line
        *style,                   // Style data
        *text;                   // Text data

    // If this is just a selection change, just unselect the style buffer...
    if (nInserted == 0 && nDeleted == 0) {
        app_style_buffer->unselect();
        return;
    }

    // Track changes in the text buffer...
    if (nInserted > 0) {
        // Insert characters into the style buffer...
        style = new char[nInserted + 1];
        memset(style, 'A', nInserted);
        style[nInserted] = '\0';

        app_style_buffer->replace(pos, pos + nDeleted, style);
        delete[] style;
    } else {
        // Just delete characters in the style buffer...
        app_style_buffer->remove(pos, pos + nDeleted);
    }

    // Select the area that was just updated to avoid unnecessary
    // callbacks...
    app_style_buffer->select(pos, pos + nInserted - nDeleted);

    // Re-parse the changed region; we do this by parsing from the
    // beginning of the line of the changed region to the end of
    // the line of the changed region... Then we check the last
    // style character and keep updating if we have a multi-line
    // comment character...
    start = app_text_buffer->line_start(pos);
    end   = app_text_buffer->line_end(pos + nInserted - nDeleted);
    text  = app_text_buffer->text_range(start, end);
    style = app_style_buffer->text_range(start, end);
    last  = style[end - start - 1];

    style_parse(text, style, end - start);

    app_style_buffer->replace(start, end, style);

```

```

((Fl_Text_Editor *)cbArg)->redisplay_range(start, end);

if (last != style[end - start - 1]) {
    // The last character on the line changed styles, so reparse the
    // remainder of the buffer...
    free(text);
    free(style);

    end = app_text_buffer->length();
    text = app_text_buffer->text_range(start, end);
    style = app_style_buffer->text_range(start, end);

    style_parse(text, style, end - start);

    app_style_buffer->replace(start, end, style);
    ((Fl_Text_Editor *)cbArg)->redisplay_range(start, end);
}

free(text);
free(style);
}

```

The `style_parse()` function scans a copy of the text in the buffer and generates the necessary style characters for display. It assumes that parsing begins at the start of a line:

```

//
// 'style_parse()' - Parse text and produce style data.
//

void
style_parse(const char *text,
            char *style,
            int length) {
    char current;
    int col;
    int last;
    char buf[255],
        *bufptr;
    const char *temp;

    for (current = *style, col = 0, last = 0; length > 0; length--, text++) {
        if (current == 'A') {
            // Check for directives, comments, strings, and keywords...
            if (col == 0 && *text == '#') {
                // Set style to directive
                current = 'E';
            } else if (strncmp(text, "//", 2) == 0) {
                current = 'B';
            } else if (strncmp(text, "/*", 2) == 0) {
                current = 'C';
            } else if (strncmp(text, "\\\"", 2) == 0) {
                // Quoted quote...
                *style++ = current;
                *style++ = current;
                text++;
                length--;
                col += 2;
                continue;
            } else if (*text == '\\') {
                current = 'D';
            } else if (!last && islower(*text)) {
                // Might be a keyword...
                for (temp = text, bufptr = buf;
                     islower(*temp) && bufptr < (buf + sizeof(buf) - 1);
                     *bufptr++ = *temp++);

                if (!islower(*temp)) {
                    *bufptr = '\\0';

                    bufptr = buf;

                    if (bsearch(&bufptr, code_types,
                               sizeof(code_types) / sizeof(code_types[0]),
                               sizeof(code_types[0]), compare_keywords)) {
                        while (text < temp) {
                            *style++ = 'F';
                            text++;
                            length--;
                            col++;
                        }

                        text--;
                        length++;
                        last = 1;
                        continue;
                    } else if (bsearch(&bufptr, code_keywords,
                                     sizeof(code_keywords) / sizeof(code_keywords[0]),

```

```

        sizeof(code_keywords[0]), compare_keywords)) {
    while (text < temp) {
        *style++ = 'G';
        text ++;
        length --;
        col ++;
    }

    text --;
    length ++;
    last = 1;
    continue;
}
}
}
} else if (current == 'C' && strcmp(text, "*/", 2) == 0) {
    // Close a C comment...
    *style++ = current;
    *style++ = current;
    text ++;
    length --;
    current = 'A';
    col += 2;
    continue;
} else if (current == 'D') {
    // Continuing in string...
    if (strcmp(text, "\\\"", 2) == 0) {
        // Quoted end quote...
        *style++ = current;
        *style++ = current;
        text ++;
        length --;
        col += 2;
        continue;
    } else if (*text == '\\') {
        // End quote...
        *style++ = current;
        col ++;
        current = 'A';
        continue;
    }
}

// Copy style info...
if (current == 'A' && (*text == '{' || *text == '}')) *style++ = 'G';
else *style++ = current;
col ++;

last = isalnum(*text) || *text == '.';

if (*text == '\\n') {
    // Reset column and possibly reset the style
    col = 0;
    if (current == 'B' || current == 'E') current = 'A';
}
}
}
}

```

1.8 Drawing Things in FLTK

This chapter covers the drawing functions that are provided with FLTK.

1.8.1 When Can You Draw Things in FLTK?

There are only certain places you can execute FLTK code that draws to the computer's display. Calling these functions at other places will result in undefined behavior!

- The most common place is inside the virtual `Fl_Widget::draw()` method. To write code here, you must subclass one of the existing `Fl_Widget` classes and implement your own version of `draw()`.
- You can also create custom `boxtypes` and `labeltypes`. These involve writing small procedures that can be called by existing `Fl_Widget::draw()` methods. These "types" are identified by an 8-bit index that is stored in the widget's `box()`, `labeltype()`, and possibly other properties.

- You can call `Fl_Window::make_current()` to do incremental update of a widget. Use `Fl_Widget::window()` to find the window.

In contrast, code that draws to other drawing surfaces than the display (i.e., instances of derived classes of the `Fl_Surface_Device` class, except `Fl_Display_Device`, such as `Fl_Printer` and `Fl_Copy_Surface`) can be executed at any time as follows:

1. Make your surface the new current drawing surface calling the `Fl_Surface_Device::push_current(Fl_Surface_Device*)` function.
2. Make a series of calls to any of the drawing functions described below; these will operate on the new current drawing surface;
3. Set the current drawing surface back to its previous state calling `Fl_Surface_Device::pop_current()`.

1.8.2 What Units Do FLTK Functions Use?

Before version 1.4 all graphical quantities used by FLTK were in pixel units: a window of width 500 units was 500 pixels wide, a line of length 10 units was 10 pixels long, lines of text written using a 14-point font were 14 pixels below each other. This organization is not sufficient to support GUI apps that can be drawn on screens of varying pixel density, especially on High-DPI screens, because widgets become very small and text becomes unreadable.

FLTK version 1.4 introduces a new feature, a screen-specific **scale factor** which is a float number with a typical value in the 1-2.5 range and is used as follows: any graphical element with an FLTK value of v units is drawn on the screen with $v * scale$ units. Thus, a window with width 500 units is $500 * scale$ pixels wide, a line of length 10 units is $10 * scale$ pixels long, lines of text written using a 14-point font are $14 * scale$ pixels below each other. Consider a system with two screens, one with regular DPI and one with a twice higher DPI. If the first screen's scale factor is set to 1 and that of the second screen to 2, the GUI of any FLTK app appears equally sized on the two screens.

FLTK uses several units to measure graphical elements:

- All quantities used by the public FLTK API to measure graphical elements (e.g., window widths, line lengths, font sizes, clipping regions, image widths and heights) are in **FLTK units** except if it's explicitly documented another unit is used. FLTK units are both platform- and DPI-independent. An example of FLTK API using another unit is `Fl_Gl_Window::pixel_w()`.
- Just before drawing to a screen, the library internally multiplies all quantities expressed in FLTK units by the current value of the scale factor for the screen in use and obtains quantities in **drawing units**. The current scale factor value, for an `Fl_Window` named *window*, is given by


```
int nscreen = window->screen_num(); // the screen where window is mapped
float s = Fl::screen_scale(nscreen); // this screen's scale factor
```

One drawing unit generally corresponds to one screen pixel ...

- ... but not on macOS and for retina displays, where one drawing unit corresponds to two pixels.
- ... and not with the Wayland platform, where one drawing unit may correspond to 1, 2, or 3 pixels according to the current value of the Wayland-defined, integer-valued scale factor.

At application start time, FLTK attempts to detect the adequate scale factor value for each screen of the system. Here is how that's done under the [X11](#), [Windows](#), and [Wayland](#) platforms. If the resulting scale factor is not satisfactory, and also under the macOS platform, it's possible to set the `FLTK_SCALING_FACTOR` environmental variable to the desired numerical value (e.g., 1.75) and any FLTK app will start scaled with that value. Furthermore, it's possible to change the scale factor value of any screen at run time with `ctrl+/-/0/` keystrokes which enlarge, shrink, and reset, respectively, all FLTK windows on a screen and their content. Under macOS, the corresponding GUI scaling shortcuts are `cmd+/-/0/`.

GUI rescaling involves also image drawing: the screen area covered by the drawn image contains a number of pixels that grows with the scale factor. When FLTK draws images, it maps the image data (the size of these data is given by `Fl_Image::data_w()` and `Fl_Image::data_h()`) to the screen area whose size (in FLTK units) is given by `Fl_Image::w()` and `Fl_Image::h()`. How exactly such mapping is performed depends on the image type, the platform and some hardware features. The most common case for `Fl_RGB_Image`'s is that FLTK uses a scaled drawing system feature that directly maps image data to screen pixels. An important feature of FLTK for image drawing is the `Fl_Image::scale()` member function, new in FLTK version 1.4. This function controls the image drawing size (in FLTK units, given by `Fl_Image::w()` and `Fl_Image::h()`) independently from the size of the image data (given by `Fl_Image::data_w()` and `Fl_Image::data_h()`). An image with large enough data size can thus be drawn at the full resolution of the screen even when the screen area covered by the image grows following the GUI scale factor.

The `Fl_Image_Surface` class is intended to create an `Fl_RGB_Image` from a series of FLTK drawing operations. The `Fl_Image_Surface` constructor allows to control whether the size in pixels of the resulting image matches the FLTK units used when performing drawing operations, or matches the number of pixels corresponding to these FLTK units given the current value of the scale factor. The first result is obtained with `new Fl_Image_Surface(w, h)`, the second with `new Fl_Image_Surface(w, h, 1)`.

When drawing to `Fl_Printer` or `Fl_PostScript_File_Device`, the drawing unit is initially one point, that is, 1/72 of an inch. This unit is changed by calls to `Fl_Paged_Device::scale()`.

1.8.3 Drawing Functions

To use the drawing functions you must first include the `<FL/fl_draw.H>` header file. FLTK provides the following types of drawing functions:

- [Boxes](#)
- [Clipping](#)
- [Colors](#)
- [Color Contrast](#)
- [Line Dashes and Thickness](#)
- [Drawing Fast Shapes](#)
- [Drawing Complex Shapes](#)
- [Drawing Text](#)
- [Fonts](#)
- [Character Encoding](#)
- [Drawing Overlays](#)
- [Drawing Images](#)
- [Direct Image Drawing](#)
- [Direct Image Reading](#)
- [Image Classes](#)
- [Offscreen Drawing](#)

1.8.3.1 Boxes

FLTK provides three functions that can be used to draw boxes for buttons and other UI controls. Each function uses the supplied upper-lefthand corner and width and height to determine where to draw the box.

```
void fl_draw_box(FL_Boxtype b, int x, int y, int w, int h, FL_Color c)
```

The `fl_draw_box()` function draws a standard boxtype `b` in the specified color `c`.

```
void fl_frame(const char *s, int x, int y, int w, int h)
void fl_frame2(const char *s, int x, int y, int w, int h)
```

The `fl_frame()` and `fl_frame2()` functions draw a series of line segments around the given box. The string `s` must contain groups of 4 letters which specify one of 24 standard grayscale values, where 'A' is black and 'X' is white. The results of calling these functions with a string that is not a multiple of 4 characters in length are undefined.

The only difference between `fl_frame()` and `fl_frame2()` is the order of the line segments:

- For `fl_frame()` the order of each set of 4 characters is: top, left, bottom, right.
- For `fl_frame2()` the order of each set of 4 characters is: bottom, right, top, left.

Note that `fl_frame(FL_Boxtype b)` is described in the [Box Types](#) section.

1.8.3.2 Clipping

You can limit all your drawing to a rectangular region by calling `fl_push_clip()`, and put the drawings back by using `fl_pop_clip()`. This rectangle is measured in [FLTK units](#) and is unaffected by the current transformation matrix.

In addition, the system may provide clipping when updating windows which may be more complex than a simple rectangle.

```
void fl_push_clip(int x, int y, int w, int h)
void fl_clip(int x, int y, int w, int h)
```

Intersect the current clip region with a rectangle and push this new region onto the stack.

The `fl_clip()` version is deprecated and will be removed from future releases.

```
void fl_push_no_clip()
```

Pushes an empty clip region on the stack so nothing will be clipped.

```
void fl_pop_clip()
```

Restore the previous clip region.

Note: You must call `fl_pop_clip()` once for every time you call `fl_push_clip()`. If you return to FLTK with the clip stack not empty unpredictable results occur.

```
int fl_not_clipped(int x, int y, int w, int h)
```

Returns non-zero if any of the rectangle intersects the current clip region. If this returns 0 you don't have to draw the object.

Note: Under X this returns 2 if the rectangle is partially clipped, and 1 if it is entirely inside the clip region.

```
int fl_clip_box(int x, int y, int w, int h, int &X, int &Y, int &W, int &H)
```

Intersect the rectangle `x, y, w, h` with the current clip region and returns the bounding box of the result in `X, Y, W, H`. Returns non-zero if the resulting rectangle is different than the original. This can be used to limit the necessary drawing to a rectangle. `W` and `H` are set to zero if the rectangle is completely outside the region.

```
void fl_clip_region(Fl_Region r)
Fl_Region fl_clip_region()
```

Replace the top of the clip stack with a clipping region of any shape. `Fl_Region` is an operating system specific type. The second form returns the current clipping region.

1.8.3.3 Colors

FLTK manages colors as 32-bit unsigned integers, encoded as RGBA. When the "RGB" bytes are non-zero, the value is treated as RGB. If these bytes are zero, the "I" byte will be used as an index into the colormap. Colors with both "RGB" set and an "I" > 0 are reserved for special use.

Values from 0 to 255, i.e. the "I" index value, represent colors from the FLTK standard colormap and are allocated as needed on screens without TrueColor support. The **FL_Color** enumeration type defines the standard colors and color cube for the first 256 colors. All of these are named with symbols in [<FL/Enumerations.H>](#). Example:

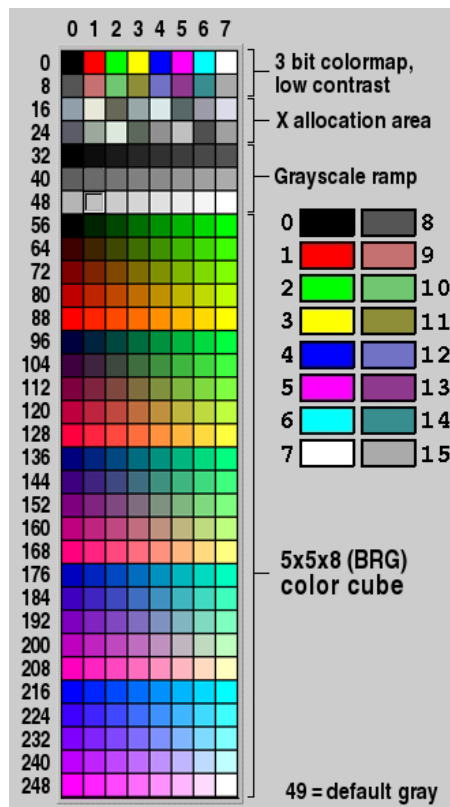


Figure 1.25 FLTK default colormap (FI_Color 0x00 - 0xff)

Color values greater than 255 are treated as 24-bit RGB values. These are mapped to the closest color supported by the screen, either from one of the 256 colors in the FLTK colormap or a direct RGB value on TrueColor screens.

```
FI_Color fl_rgb_color(uchar r, uchar g, uchar b)
FI_Color fl_rgb_color(uchar grayscale)
```

Generate FI_Color out of specified 8-bit RGB values or one 8-bit grayscale value.

```
void fl_color(FI_Color c)
void fl_color(int c)
```

Sets the color for all subsequent drawing operations. Please use the first form: the second form is only provided for back compatibility.

For colormapped displays, a color cell will be allocated out of `fl_colormap` the first time you use a color. If the colormap fills up then a least-squares algorithm is used to find the closest color.

```
FI_Color fl_color()
```

Returns the last color that was set using `fl_color()`. This can be used for state save/restore.

```
void fl_color(uchar r, uchar g, uchar b)
```

Set the color for all subsequent drawing operations. The closest possible match to the RGB color is used. The RGB color is used directly on TrueColor displays. For colormap visuals the nearest index in the gray ramp or color cube is used.

```
unsigned Fl::get_color(Fl_Color i)
void Fl::get_color(Fl_Color i, uchar &red, uchar &green, uchar &blue)
```

Generate RGB values from a colormap index value *i*. The first returns the RGB as a 32-bit unsigned integer, and the second decomposes the RGB into three 8-bit values.

```
Fl::get_system_colors()
Fl::foreground()
Fl::background()
Fl::background2()
```

The first gets color values from the user preferences or the system, and the other routines are used to apply those values.

```
Fl::own_colormap()
Fl::free_color(Fl_Color i, int overlay)
Fl::set_color(Fl_Color i, unsigned c)
```

`Fl::own_colormap()` is used to install a local colormap [X11 only].

`Fl::free_color()` and `Fl::set_color()` are used to remove and replace entries from the colormap.

There are two predefined graphical interfaces for choosing colors. The function `fl_show_colormap()` shows a table of colors and returns an `Fl_Color` index value. The `Fl_Color_Chooser` widget provides a standard RGB color chooser.

As the `Fl_Color` encoding maps to a 32-bit unsigned integer representing RGBI, it is also possible to specify a color using a hex constant as a color map index:

```
// COLOR MAP INDEX
color(0x000000II)
  ----- |
  |       |
  |       | Color map index (8 bits)
  |       | Must be zero
button->color(0x000000ff);           // colormap index #255 (FL_WHITE)
```

or specify a color using a hex constant for the RGB components:

```
// RGB COLOR ASSIGNMENTS
color(0xRRGGBB00)
  | | | |
  | | | Must be zero
  | | Blue (8 bits)
  | Green (8 bits)
  Red (8 bits)
button->color(0xff000000);           // RGB: red
button->color(0x00ff0000);           // RGB: green
button->color(0x0000ff00);           // RGB: blue
button->color(0xffffffff00);         // RGB: white
```

Note

If TrueColor is not available, any RGB colors will be set to the nearest entry in the colormap.

1.8.3.4 Color Contrast

Although these are not real "drawing" functions, creating readable contrast is essential in a good GUI design. FLTK tries to help with this by providing [fl_contrast\(\)](#) and related functions.

The basic function is `Fl_Color fl_contrast(Fl_Color fg, Fl_Color bg, int context, int size);`

The parameters `context` and `size` are optional and reserved for future use (since FLTK 1.4.0).

The return value can be used to substitute the foreground color `fg` used for drawing (usually the "text" or "label" color) on a particular background color `bg` with either black (`FL_BLACK`) or white (`FL_WHITE`). This is useful if the background color is not known or can be changed by the user or a system "theme".

FLTK calculates the contrast between `fg` and `bg` and returns the same color (`fg`) if the contrast is considered sufficient or one of `FL_BLACK` or `FL_WHITE` if the contrast of the given foreground color would be insufficient. Then either `FL_BLACK` or `FL_WHITE` is chosen, whichever has the higher contrast with the background color.

Example, may be used in a widget's `draw()` method:

```
Fl_Color bg = color();           // background color of the widget
Fl_Color fg = FL_BLUE;           // the chosen foreground (drawing) color
fl_color(fl_contrast(fg, bg));    // set the drawing color
fl_rect(...);                    // draw a rectangle with sufficient contrast
```

FLTK 1.4.0 introduced a new contrast algorithm which is superior to the one used up to FLTK 1.3.x. You can use

```
fl_contrast_mode(FL_CONTRAST_LEGACY);
```

early in your program to select the old behavior if you really need strict backwards compatibility. This is discouraged because the new algorithm is much better with regard to human contrast perception. The default mode since FLTK 1.4.0 is

```
fl_contrast_mode(FL_CONTRAST_CIELAB);
```

For more info please see the linked documentation of these functions.

Additionally the old and new contrast calculations can be fine tuned with the new function (since 1.4.0)

```
fl_contrast_level(int level);
```

This is not recommended but can be useful for some border cases. Please refer to the documentation of [fl_contrast_level\(\)](#).

Finally, developers can define their own contrast calculation function with

```
void fl_contrast_function(Fl_Contrast_Function *f);
```

Please see the documentation for details.

1.8.3.5 Line Dashes and Thickness

FLTK supports drawing of lines with different styles and widths.

```
void fl_line_style(int style, int width, char* dashes)
```

FL_SOLID		FL_CAP_FLAT	
FL_DASH		FL_CAP_ROUND	
FL_DOT		FL_CAP_SQUARE	
FL_DASHDOT		FL_JOIN_MITER	
FL_DASHDOTDOT		FL_JOIN_ROUND	
		FL_JOIN_BEVEL	

Figure 1.26 fl_line_style() styles

Set how to draw lines (the "pen"). If you change this it is your responsibility to set it back to the default with `fl_line_style(0)`.

`style` is a bitmask which is a bitwise-OR of the following values. If you don't specify a dash type you will get a solid line. If you don't specify a cap or join type you will get a system-defined default of whatever value is fastest.

- `FL_SOLID` solid line
- `FL_DASH` 75% dashed line
- `FL_DOT` 50% pixel dotted
- `FL_DASHDOT` dash / dot pattern
- `FL_DASHDOTDOT` dash / two dot pattern
- `FL_CAP_FLAT` end is flat
- `FL_CAP_ROUND` end is round
- `FL_CAP_SQUARE` extends past end point 1/2 line width
- `FL_JOIN_MITER` line join extends to a point
- `FL_JOIN_ROUND` line join is rounded
- `FL_JOIN_BEVEL` line join is flat (tidied)

`width` is the number of [FLTK units](#) thick to draw the lines. Zero results in the system-defined default, which on both X and Windows is somewhat different and nicer than 1.

`dashes` is a pointer to an array of dash lengths, measured in [FLTK units](#). The first location is how long to draw a solid portion, the next is how long to draw the gap, then the solid, etc. It is terminated with a zero-length entry. A `NULL` pointer or a zero-length array results in a solid line. Odd array sizes are not supported and result in undefined behavior.

Note

- Full functionality is not available under Windows 95, 98, and Me due to the reduced drawing functionality these operating systems provide.
- Because of how line styles are implemented on Windows systems, you *must* set the line style *after* setting the drawing color. If you set the color after the line style you will lose the line style settings!
- The dashes array does not work under Windows 95, 98, or Me, since those operating systems do not support complex line styles.

1.8.3.6 Drawing Fast Shapes

These functions are used to draw almost all the FLTK widgets. They draw on exact pixel boundaries and are as fast as possible. Their behavior is duplicated exactly on all platforms FLTK is ported. It is undefined whether these are affected by the [transformation matrix](#), so you should only call these while the matrix is set to the identity matrix (the default).

```
void fl_point(int x, int y)
```

Draw a single pixel at the given coordinates.

```
void fl_rectf(int x, int y, int w, int h)
void fl_rectf(int x, int y, int w, int h, FL_Color c)
```

Color a rectangle that exactly fills the given bounding box.

```
void fl_rectf(int x, int y, int w, int h, uchar r, uchar g, uchar b)
```

Color a rectangle with "exactly" the passed `r, g, b` color. On screens with less than 24 bits of color this is done by drawing a solid-colored block using [fl_draw_image\(\)](#) so that the correct color shade is produced.

```
void fl_rect(int x, int y, int w, int h)
void fl_rect(int x, int y, int w, int h, FL_Color c)
```

Draw a 1-pixel border *inside* this bounding box.

```
void fl_rounded_rect(int x, int y, int w, int h, int radius) void fl_rounded_rectf(int x, int y, int w, int h, int radius)
```

Draw an outlined or filled rectangle with rounded corners.

```
void fl_line(int x, int y, int x1, int y1)
void fl_line(int x, int y, int x1, int y1, int x2, int y2)
```

Draw one or two lines between the given points.

```
void fl_loop(int x, int y, int x1, int y1, int x2, int y2)
void fl_loop(int x, int y, int x1, int y1, int x2, int y2, int x3, int y3)
```

Outline a 3 or 4-sided polygon with lines.

```
void fl_polygon(int x, int y, int x1, int y1, int x2, int y2)
void fl_polygon(int x, int y, int x1, int y1, int x2, int y2, int x3, int y3)
```

Fill a 3 or 4-sided polygon. The polygon must be convex.

```
void fl_xyline(int x, int y, int x1)
void fl_xyline(int x, int y, int x1, int y2)
void fl_xyline(int x, int y, int x1, int y2, int x3)
```

Draw horizontal and vertical lines. A horizontal line is drawn first, then a vertical, then a horizontal.

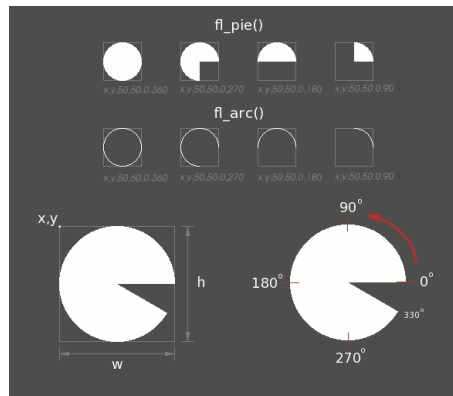
```
void fl_yxline(int x, int y, int y1)
void fl_yxline(int x, int y, int y1, int x2)
void fl_yxline(int x, int y, int y1, int x2, int y3)
```

Draw vertical and horizontal lines. A vertical line is drawn first, then a horizontal, then a vertical.

```
void fl_arc(int x, int y, int w, int h, double a1, double a2)
void fl_pie(int x, int y, int w, int h, double a1, double a2)
```

Draw ellipse sections using integer coordinates. These functions match the rather limited circle drawing code provided by X and Windows. The advantage over using `fl_arc()` with floating point coordinates is that they are faster because they often use the hardware, and they draw much nicer small circles, since the small sizes are often hard-coded bitmaps.

If a complete circle is drawn it will fit inside the passed bounding box. The two angles are measured in degrees counter-clockwise from 3'o'clock and are the starting and ending angle of the arc, `a2` must be greater or equal to `a1`.

Figure 1.27 `fl_pie()` and `fl_arc()`

`fl_arc()` draws a series of lines to approximate the arc. Notice that the integer version of `fl_arc()` has a different number of arguments to the other `fl_arc()` function described later in this chapter.

`fl_pie()` draws a filled-in pie slice. This slice may extend outside the line drawn by `fl_arc()`; to avoid this use `w-1` and `h-1`.

```
void fl_scroll(int X, int Y, int W, int H, int dx, int dy, void (draw_area)(void, int,int,int,int), void* data)
```

Scroll a rectangle and draw the newly exposed portions. The contents of the rectangular area is first shifted by `dx` and `dy` FLTK units. The callback is then called for every newly exposed rectangular area,

1.8.3.7 Drawing Complex Shapes

The complex drawing functions let you draw arbitrary shapes with 2-D linear transformations. The functionality matches that found in the Adobe® PostScript™ language. The exact pixels that are filled are less defined than for the fast drawing functions so that FLTK can take advantage of drawing hardware. On both X and Windows the transformed vertices are rounded to integers before drawing the line segments: this severely limits the accuracy of these functions for complex graphics, so use OpenGL when greater accuracy and/or performance is required.

```
void fl_load_matrix(double a,double b,double c,double d,double x,double y) void fl_load_identity()
```

Set the current transformation.

```
void fl_push_matrix()
void fl_pop_matrix()
```

Save and restore the current transformation. The maximum depth of the stack is 32 entries.

```
void fl_scale(double x,double y)
void fl_scale(double x)
void fl_translate(double x,double y)
void fl_rotate(double d)
void fl_mult_matrix(double a,double b,double c,double d,double x,double y)
```

Concatenate another transformation onto the current one. The rotation angle is in degrees (not radians) and is counter-clockwise.

```
double fl_transform_x(double x, double y)
double fl_transform_y(double x, double y)
double fl_transform_dx(double x, double y)
double fl_transform_dy(double x, double y)
void fl_transformed_vertex(double xf, double yf)
```

Transform a coordinate or a distance using the current transformation matrix. After transforming a coordinate pair, it can be added to the vertex list without any further translations using `fl_transformed_vertex()`.

```
void fl_begin_points()
void fl_end_points()
```

Start and end drawing a list of points. Points are added to the list with `fl_vertex()`.

```
void fl_begin_line()
void fl_end_line()
```

Start and end drawing lines.

```
void fl_begin_loop()
void fl_end_loop()
```

Start and end drawing a closed sequence of lines.

```
void fl_begin_polygon()
void fl_end_polygon()
```

Start and end drawing a convex filled polygon.

```
void fl_begin_complex_polygon()
void fl_gap()
void fl_end_complex_polygon()
```

Start and end drawing a complex filled polygon. This polygon may be concave, may have holes in it, or may be several disconnected pieces. Call `fl_gap()` to separate loops of the path. It is unnecessary but harmless to call `fl_gap()` before the first vertex, after the last one, or several times in a row.

`fl_gap()` should only be called between `fl_begin_complex_polygon()` and `fl_end_complex_polygon()`. To outline the polygon, use `fl_begin_loop()` and replace each `fl_gap()` with a `fl_end_loop();fl_begin_loop()` pair.

Note: For portability, you should only draw polygons that appear the same whether "even/odd" or "non-zero" winding rules are used to fill them. Holes should be drawn in the opposite direction of the outside loop.

void `fl_vertex(double x,double y)`

Add a single vertex to the current path.

void `fl_curve(double X0, double Y0, double X1, double Y1, double X2, double Y2, double X3, double Y3)`

Add a series of points on a Bézier curve to the path. The curve ends (and two of the points are) at `X0, Y0` and `X3, Y3`.

void `fl_arc(double x, double y, double r, double start, double end)`

Add a series of points to the current path on the arc of a circle; you can get elliptical paths by using `scale` and `rotate` before calling `fl_arc()`. The center of the circle is given by `x` and `y`, and `r` is its radius. `fl_arc()` takes `start` and `end` angles that are measured in degrees counter-clockwise from 3 o'clock. If `end` is less than `start` then it draws the arc in a clockwise direction.

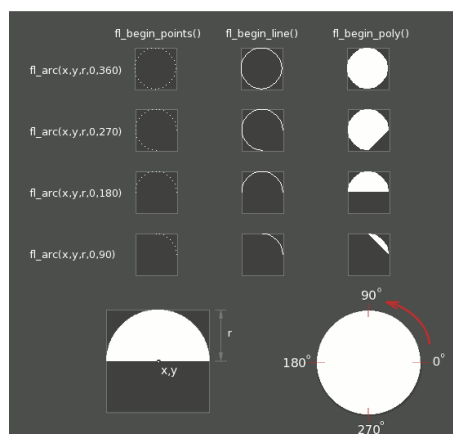


Figure 1.28 `fl_arc(x,y,r,a1,a2)`

void `fl_circle(double x, double y, double r)`

`fl_circle(x, y, r)` is equivalent to `fl_arc(x, y, r, 0, 360)` but may be faster. It must be the *only* thing in the path: if you want a circle as part of a complex polygon you must use `fl_arc()`.

Note: `fl_circle()` draws incorrectly if the transformation is both rotated and non-square scaled.

1.8.3.8 Drawing Text

All text is drawn in the [current font](#). It is undefined whether this location or the characters are modified by the current transformation.

```
void fl_draw(const char *, int x, int y)
void fl_draw(const char *, int n, int x, int y)
```

Draw a nul-terminated string or an array of `n` bytes starting at the given location. In both cases, the text must be UTF-8 encoded. Text is aligned to the left and to the baseline of the font. To align to the bottom, subtract `fl_descent()` from `y`. To align to the top, subtract `fl_descent()` and add `fl_height()`. This version of `fl_draw()` provides direct access to the text drawing function of the underlying OS. It does not apply any special handling to control characters.

```
void fl_rtl_draw(const char *str, int n, int x, int y)
```

Draw a UTF-8 string of length `n` bytes right to left starting at the given `x, y` location.

```
void fl_draw(const char* str, int x, int y, int w, int h, FL_Align align, FL_Image* img, int draw_symbols, int spacing)
```

Fancy string drawing function which is used to draw all the labels. The string is formatted and aligned inside the passed box. Handles `\t` and `\n`, expands all other control characters to `^X`, and aligns inside or against the edges of the box described by `x, y, w` and `h`. See [FL_Widget::align\(\)](#) for values for `align`. The value `FL_ALIGN_INSIDE` is ignored, as this function always prints inside the box. Parameter `spacing` controls the space between text and image.

If `img` is provided and is not `NULL`, the image is drawn above or below the text as specified by the `align` value.

The `draw_symbols` argument specifies whether or not to look for symbol names starting with the "@" character.

```
void fl_measure(const char *str, int& w, int& h, int draw_symbols)
```

Measure how wide and tall the string will be when printed by the `fl_draw(...align)` function. This includes leading/trailing white space in the string, kerning, etc.

If the incoming `w` is non-zero it will wrap to that width.

This will probably give unexpected values unless you have called `fl_font()` explicitly in your own code. Refer to the full documentation for `fl_measure()` for details on usage and how to avoid common pitfalls.

See also

`fl_text_extents()` – measure the 'inked' area of a string
`fl_width()` – measure the width of a string or single character
`fl_height()` – measure the height of the [current font](#)
`fl_descent()` – the height of the descender for the [current font](#)

int `fl_height()`

Recommended minimum line spacing for the [current font](#). You can also just use the value of `size` passed to `fl_font()`.

See also

`fl_text_extents()`, `fl_measure()`, `fl_width()`, `fl_descent()`

int `fl_descent()`

Recommended distance above the bottom of a `fl_height()` tall box to draw the text at so it looks centered vertically in that box.

double `fl_width(const char* txt)`
double `fl_width(const char* txt, int n)`
double `fl_width(unsigned int unicode_char)`

Return the width of a nul-terminated string, a sequence of `n` characters, or a single character in the [current font](#).

See also

`fl_measure()`, `fl_text_extents()`, `fl_height()`, `fl_descent()`

void `fl_text_extents(const char* txt, int& dx, int& dy, int& w, int& h)`

Determines the minimum dimensions of a nul-terminated string, ie. the 'inked area'.

Given a string "txt" drawn using `fl_draw(txt, x, y)` you would determine its extents in [FLTK units](#) on the display using `fl_text_extents(txt, dx, dy, wo, ho)` such that a bounding box that exactly fits around the inked area of the text could be drawn with `fl_rect(x+dx, y+dy, wo, ho)`.

Refer to the full documentation for [fl_text_extents\(\)](#) for details on usage.

See also

[fl_measure\(\)](#), [fl_width\(\)](#), [fl_height\(\)](#), [fl_descent\(\)](#)

```
const char* fl_shortcut_label(int shortcut)
```

Unparse a shortcut value as used by [Fl_Button](#) or [Fl_Menu_Item](#) into a human-readable string like "Alt+N". This only works if the shortcut is a character key or a numbered function key. If the shortcut is zero an empty string is returned. The return value points at a static buffer that is overwritten with each call.

1.8.3.9 Fonts

FLTK supports a set of standard fonts based on the Times, Helvetica/Arial, Courier, and Symbol typefaces, as well as custom fonts that your application may load. Each font is accessed by an index into a font table.

Initially only the first 16 faces are filled in. There are symbolic names for them: `FL_HELVETICA`, `FL_TIMES`, `FL_COURIER`, and modifier values `FL_BOLD` and `FL_ITALIC` which can be added to these, and `FL_SYMBOL` and `FL_ZAPF_DINGBATS`. Faces greater than 255 cannot be used in [Fl_Widget](#) labels, since [Fl_Widget](#) stores the index as a byte.

One important thing to note about 'current font' is that there are so many paths through the GUI event handling code as widgets are partially or completely hidden, exposed and then re-drawn and therefore you can not guarantee that 'current font' contains the same value that you set on the other side of the event loop. Your value may have been superseded when a widget was redrawn. You are strongly advised to set the font explicitly before you draw any text or query the width and height of text strings, etc.

```
void fl_font(int face, int size)
```

Set the current font, which is then used by the routines described above. You may call this outside a draw context if necessary to call [fl_width\(\)](#), but on X this will open the display.

The font is identified by a *face* and a *size*. The size of the font is measured in [FLTK units](#) and not "points". Lines should be spaced *size* FLTK units apart or more.

```
int fl_font()
int fl_size()
```

Returns the face and size set by the most recent call to `fl_font(a, b)`. This can be used to save/restore the font.

1.8.3.10 Character Encoding

FLTK 1.3 and later versions expect all text in Unicode UTF-8 encoding. UTF-8 is ASCII compatible for the first 128 characters. International characters are encoded in multibyte sequences.

FLTK expects individual characters, characters that are not part of a string, in UCS-4 encoding, which is also ASCII compatible, but requires 4 bytes to store a Unicode character.

FLTK can draw accurately any Unicode-supported script for which the system contains relevant fonts. Under X11 platforms, this requires to build the library with the `FLTK_USE_PANGO` CMake option turned On (or with `configure --enable-pango`).

Plain text drawing starting at a user-given coordinate is well supported by FLTK, including for right-to-left scripts. Further text-related operations (i.e., selection, formatting, input, and editing) are functional with left-to-right scripts only.

For more information about character encodings, see the chapter on [Unicode and UTF-8 Support](#).

1.8.3.11 Drawing Overlays

These functions allow you to draw interactive selection rectangles without using the overlay hardware. FLTK will XOR a single rectangle outline over a window.

```
void fl_overlay_rect(int x, int y, int w, int h)
void fl_overlay_clear()
```

`fl_overlay_rect()` draws a selection rectangle, erasing any previous rectangle by XOR'ing it first. `fl_overlay_clear()` will erase the rectangle without drawing a new one.

Using these functions is tricky. You should make a widget with both a `handle()` and `draw()` method. `draw()` should call `fl_overlay_clear()` before doing anything else. Your `handle()` method should call `window()->make_current()` and then `fl_overlay_rect()` after `FL_DRAG` events, and should call `fl_overlay_clear()` after a `FL_RELEASE` event.

1.8.4 Drawing Images

To draw images, you can either do it directly from data in your memory, or you can create a [Fl_Image](#) object. The advantage of drawing directly is that it is more intuitive, and it is faster if the image data changes more often than it is redrawn. The advantage of using the object is that FLTK will cache translated forms of the image (on X it uses a server pixmap) and thus redrawing is *much* faster.

1.8.4.1 Direct Image Drawing

The behavior when drawing images when the current transformation matrix is not the identity is not defined, so you should only draw images when the matrix is set to the identity.

```
void fl_draw_image(const uchar *buf,int X,int Y,int W,int H,int D,int L)
void fl_draw_image_mono(const uchar *buf,int X,int Y,int W,int H,int D,int L)
```

Draw an 8-bit per color RGB or luminance image. The pointer points at the "r" data of the top-left pixel. Color data must be in *r, g, b* order. The top left corner is given by *X* and *Y* and the size of the image is given by *W* and *H*. *D* is the delta to add to the pointer between pixels, it may be any value greater or equal to 3, or it can be negative to flip the image horizontally. *L* is the delta to add to the pointer between lines (if 0 is passed it uses *W*D*). and may be larger than *W*D* to crop data, or negative to flip the image vertically.

It is highly recommended that you put the following code before the first `show()` of *any* window in your program to get rid of the dithering if possible:

```
Fl::visual (FL_RGB);
```

Gray scale (1-channel) images may be drawn. This is done if `abs(D)` is less than 3, or by calling `fl_draw_image_mono()`. Only one 8-bit sample is used for each pixel, and on screens with different numbers of bits for red, green, and blue only gray colors are used. Setting *D* greater than 1 will let you display one channel of a color image.

Note: The X version does not support all possible visuals. If FLTK cannot draw the image in the current visual it will abort. FLTK supports any visual of 8 bits or less, and all common TrueColor visuals up to 32 bits.

```
typedef void (*Fl_Draw_Image_Cb)(void *data,int x,int y,int w,uchar *buf)
void fl_draw_image(Fl_Draw_Image_Cb cb,void *data,int X,int Y,int W,int H,int D)
void fl_draw_image_mono(Fl_Draw_Image_Cb cb,void *data,int X,int Y,int W,int H,int D)
```

Call the passed function to provide each scan line of the image. This lets you generate the image as it is being drawn, or do arbitrary decompression of stored data, provided it can be decompressed to individual scan lines easily.

The callback is called with the `void*` user data pointer which can be used to point at a structure of information about the image, and the *x*, *y*, and *w* of the scan line desired from the image. 0,0 is the upper-left corner of the image, *not* *X*, *Y*. A pointer to a buffer to put the data into is passed. You must copy *w* pixels from scanline *y*, starting at pixel *x*, to this buffer.

Due to cropping, less than the whole image may be requested. So *x* may be greater than zero, the first *y* may be greater than zero, and *w* may be less than *W*. The buffer is long enough to store the entire *W*D* pixels, this is for convenience with some decompression schemes where you must decompress the entire line at once: decompress it into the buffer, and then if *x* is not zero, copy the data over so the *x*'th pixel is at the start of the buffer.

You can assume the `y`'s will be consecutive, except the first one may be greater than zero.

If `D` is 4 or more, you must fill in the unused bytes with zero.

```
int fl_draw_pixmap(char* const* data, int x, int y, FL_Color bg)
int fl_draw_pixmap(const char* const* cdata, int x, int y, FL_Color bg)
```

Draws XPM image data, with the top-left corner at the given position. The image is dithered on 8-bit displays so you won't lose color space for programs displaying both images and pixmaps. This function returns zero if there was any error decoding the XPM data.

To use an XPM, do:

```
#include "foo.xpm"
...
fl_draw_pixmap(foo, X, Y);
```

Transparent colors are replaced by the optional `FL_Color` argument. To draw with true transparency you must use the `FL_Pixmap` class.

```
int fl_measure_pixmap(char* const* data, int &w, int &h)
int fl_measure_pixmap(const char* const* cdata, int &w, int &h)
```

An XPM image contains the dimensions in its data. This function finds and returns the width and height. The return value is non-zero if the dimensions were parsed ok and zero if there was any problem.

1.8.4.2 Direct Image Reading

FLTK provides a single function for reading from the current window or off-screen buffer into a RGB(A) image buffer.

```
uchar* fl_read_image(uchar *p, int X, int Y, int W, int H, int alpha)
```

Read a RGB(A) image from the current window or off-screen buffer. The `p` argument points to a buffer that can hold the image and must be at least `W*H*3` bytes when reading RGB images and `W*H*4` bytes when reading RGBA images. If `NULL`, `fl_read_image()` will create an array of the proper size which can be freed using `delete[]`.

The `alpha` parameter controls whether an alpha channel is created and the value that is placed in the alpha channel. If 0, no alpha channel is generated.

1.8.4.3 Image Classes

FLTK provides a base image class called `FL_Image` which supports creating, copying, and drawing images of various kinds, along with some basic color operations. Images can be used as labels for widgets using the `image()` and `deimage()` methods or drawn directly. Images can be drawn scaled to any size, independently from the size of the image's data (see `FL_Image::scale()`).

The `FL_Image` class does almost nothing by itself, but is instead supported by three basic image types:

- `FL_Bitmap`
- `FL_Pixmap`
- `FL_RGB_Image`

The `FL_Bitmap` class encapsulates a mono-color bitmap image. The `draw()` method draws the image using the current drawing color.

The `FL_Pixmap` class encapsulates a colormapped image. The `draw()` method draws the image using the colors in the file, and masks off any transparent colors automatically.

The `FL_RGB_Image` class encapsulates a full-color (or grayscale) image with 1 to 4 color components. Images with an even number of components are assumed to contain an alpha channel that is used for transparency. The transparency provided by the `draw()` method is either a 24-bit blend against the existing window contents or a "screen door" transparency mask, depending on the platform and screen color depth.

char `fl_can_do_alpha_blending()`

`fl_can_do_alpha_blending()` will return 1, if your platform supports true alpha blending for RGBA images, or 0, if FLTK will use screen door transparency.

FLTK also provides several image classes based on the three standard image types for common file formats:

- `FL_GIF_Image`
- `FL_Anim_GIF_Image`
- `FL_JPEG_Image`
- `FL_PNG_Image`
- `FL_PNM_Image`
- `FL_XBM_Image`
- `FL_XPM_Image`
- `FL_SVG_Image`
- `FL_BMP_Image`
- `FL_ICO_Image`

Each of these image classes loads a named file of the corresponding format. The `FL_Shared_Image` class can be used to load any type of image file - the class examines the file and constructs an image of the appropriate type.

Finally, FLTK provides a special image class called `FL_Tiled_Image` to tile another image object in the specified area. This class can be used to tile a background image in a `FL_Group` widget, for example.

```
virtual FL_Image* FL_Image::copy()
virtual FL_Image* FL_Image::copy(int W, int H) const
```

The `copy()` method creates a copy of the image. The second form specifies the new size of the image - the image is resized using the nearest-neighbor algorithm (this is the default).

Note

As of FLTK 1.3.3 the image resizing algorithm can be changed. See [FI_Image::RGB_scaling\(FI_RGB_Scaling method\)](#)

virtual void [FI_Image::draw\(int x, int y, int w, int h, int ox, int oy\)](#)

The `draw()` method draws the image object. `x, y, w, h` indicates the destination rectangle. `ox, oy, w, h` is the source rectangle. This source rectangle is copied to the destination. The source rectangle may extend outside the image, i.e. `ox` and `oy` may be negative and `w` and `h` may be bigger than the image, and this area is left unchanged.

Note

See exceptions for [FI_Tiled_Image::draw\(\)](#) regarding arguments `ox`, `oy`, `w`, and `h`.

virtual void [FI_Image::draw\(int x, int y\)](#)

Draws the image with the upper-left corner at `x, y`. This is the same as doing `img->draw(x, y, img->w(), img->h(), 0, 0)` where `img` is a pointer to any [FI_Image](#) type.

1.8.5 Offscreen Drawing

Sometimes it can be very useful to generate a complex drawing in memory first and copy it to the screen at a later point in time. This technique can significantly reduce the amount of repeated drawing. Offscreen drawing functions are declared in [<FL/fl_draw.H>](#).

[FI_Double_Window](#) uses offscreen rendering to avoid flickering on systems that don't support double-buffering natively.

FLTK can draw into an offscreen buffer at any time. There is no need to wait for an [FI_Widget::draw\(\)](#) to occur.

Note

In FLTK 1.3.x and earlier versions all offscreen drawing functions described below were implemented as macros and created certain temporary variables to save context information. You needed to create local scope blocks with curly braces `{ ... }` if you used offscreen functions more than once in a function or method. This is no longer necessary since offscreen drawing is now implemented in real functions (no macros).

Example:

```
FI_Offscreen oscr = fl_create_offscreen(120, 120);
fl_begin_offscreen(oscr);
fl_color(FL_WHITE);
fl_rectf(0, 0, 120, 120);
fl_end_offscreen();
// other code here
fl_begin_offscreen(oscr);
fl_color(FL_BLACK);
fl_rectf(10, 10, 100, 100);
fl_end_offscreen();
// other code here
fl_delete_offscreen(oscr);
```

FI_Offscreen [fl_create_offscreen\(int w, int h\)](#)

Create an RGB offscreen buffer containing as many pixels as in a screen area of size `w,h` [FLTK units](#).

```
void fl_delete_offscreen(FL_Offscreen)
```

Delete a previously created offscreen buffer. All drawings are lost.

```
void fl_begin_offscreen(FL_Offscreen)
```

Send all subsequent drawing commands to this offscreen buffer.

```
void fl_end_offscreen()
```

Quit sending drawing commands to this offscreen buffer.

```
void fl_copy_offscreen(int x, int y, int w, int h, FL_Offscreen osrc, int srcx, int srcy)
```

Copy a rectangular area of the size `w*h` from `srcx,srcy` in the offscreen buffer into the current drawing surface at `x,y`.

```
void fl_rescale_offscreen(FL_Offscreen &osrc)
```

Adapts the offscreen's size in pixels to a changed value of the scale factor while keeping the offscreen's graphical content.

1.9 Handling Events

This chapter discusses the FLTK event model and how to handle events in your program or widget.

1.9.1 The FLTK Event Model

Every time a user moves the mouse pointer, clicks a button, or presses a key, an event is generated and sent to your application. Events can also come from other programs like the window manager.

Events are identified by the integer argument passed to a `handle()` method that overrides the `FL_Widget::handle()` virtual method. Other information about the most recent event is stored in static locations and acquired by calling the [FL::event_*\(\) methods](#). This static information remains valid until the next event is read from the window system, so it is ok to look at it outside of the `handle()` method, for instance in a callback.

Event numbers can be converted to their actual names using the `fl_eventnames[]` array defined in `#include <FL/names.h>`; see next chapter for details.

In the next chapter, the `MyClass::handle()` example shows how to override the `FL_Widget::handle()` method to accept and process specific events.

1.9.2 Mouse Events

Classic mice provide two or three buttons:

- a primary (left) button, typically used to point and click on objects
- a secondary (right) button, typically used to open special menus etc.
- a middle button for other special events.

The primary and secondary mouse buttons can usually be inverted by system setup functions for left handed usage. FLTK always sends the mouse button number according to the "logical" mouse button, i.e. such a setup is transparent for application programmers.

The middle button was later replaced by a scroll wheel to allow scrolling text or other objects in a window or widget vertically. Pushing the scroll wheel down acts like pushing the middle mouse button.

If the scroll wheel is used to scroll while the Shift key is held, the scrolling direction is horizontal rather than vertical.

Some mice may even have a "scroll ball" or similar (touch) feature. These mice support horizontal and vertical scrolling simultaneously.

Newer mice add even more buttons, called "side buttons":

- a "back" button, typically used to "go back", e.g. in browsers
- a "forward" button, typically used to "go forward", e.g. in browsers

FLTK handles up to five buttons since version 1.3.10 across all supported platforms. Note that Windows doesn't support more than five buttons whereas other platforms may support more. FLTK ignores other buttons as long as they send "mouse button" events.

Some gaming mice with five or more buttons may send their button clicks as keyboard events which will be handled like normal text input by FLTK. This has not been tested though.

1.9.2.1 FL_PUSH

A mouse button has gone down with the mouse pointing at this widget. You can find out what button by calling `Fl::event_button()`. You find out the mouse position by calling `Fl::event_x()` and `Fl::event_y()`.

A widget indicates that it "wants" the mouse click by returning non-zero from its `handle()` method, as in the `MyClass::handle()` example. It will then become the `Fl::pushed()` widget and will get `FL_DRAG` and the matching `FL_RELEASE` events. If `handle()` returns zero then FLTK will try sending the `FL_PUSH` to another widget.

1.9.2.2 FL_DRAG

The mouse has moved with at least one button held down. The current button state is in `Fl::event_state()`. The mouse position is in `Fl::event_x()` and `Fl::event_y()`.

In order to receive `FL_DRAG` events, the widget must return non-zero when handling `FL_PUSH`.

Moving the mouse keeps sending `FL_DRAG` events as long as at least one button is held down. If any buttons are pushed and released during the drag operation FLTK sends `FL_PUSH` and `FL_RELEASE` events even while dragging. The current button status can be found in `Fl::event_state()` and in `Fl::event_buttons()`.

When the last button has been released FLTK sends `FL_MOVE` events again. Note: The button released last need not necessarily be the one that started the drag operation.

Since FLTK 1.4.0 dragging is supported for all five supported mouse buttons, for instance the user can drag the mouse while the "back" button is held down. Warning: This may or may not be compatible with other applications for drag-and-drop operations **between** applications and lead to unexpected behavior.

1.9.2.3 FL_RELEASE

A mouse button has been released. You can find out what button by calling `Fl::event_button()`.

In order to receive the `FL_RELEASE` event, the widget must return non-zero when handling `FL_PUSH`.

1.9.2.4 FL_MOVE

The mouse has moved without any mouse buttons held down. This event is sent to the `Fl::belowmouse()` widget.

In order to receive `FL_MOVE` events, the widget must return non-zero when handling `FL_ENTER`.

1.9.2.5 FL_MOUSEWHEEL

The user has moved the mouse wheel. The `Fl::event_dx()` and `Fl::event_dy()` methods can be used to find the amount to scroll horizontally (dx) and vertically (dy). On mice with a single scroll wheel holding the Shift key on the keyboard while scrolling sends horizontal scroll events.

1.9.3 Focus Events

1.9.3.1 FL_ENTER

The mouse has been moved to point at this widget. This can be used for highlighting feedback. If a widget wants to highlight or otherwise track the mouse, it indicates this by returning non-zero from its `handle()` method. It then becomes the `Fl::belowmouse()` widget and will receive `FL_MOVE` and `FL_LEAVE` events.

1.9.3.2 FL_LEAVE

The mouse has moved out of the widget.

In order to receive the `FL_LEAVE` event, the widget must return non-zero when handling `FL_ENTER`.

1.9.3.3 FL_FOCUS

This indicates an *attempt* to give a widget the keyboard focus.

If a widget wants the focus, it should change itself to display the fact that it has the focus, and return non-zero from its `handle()` method. It then becomes the `Fl::focus()` widget and gets `FL_KEYDOWN`, `FL_KEYUP`, and `FL_UNFOCUS` events.

The focus will change either because the window manager changed which window gets the focus, or because the user tried to navigate using tab, arrows, or other keys. You can check `Fl::event_key()` to figure out why it moved. For navigation it will be the key pressed and for interaction with the window manager it will be zero.

1.9.3.4 FL_UNFOCUS

This event is sent to the previous `Fl::focus()` widget when another widget gets the focus or the window loses focus.

1.9.4 Keyboard Events

1.9.4.1 FL_KEYBOARD, FL_KEYDOWN, FL_KEYUP

A key was pressed (FL_KEYDOWN) or released (FL_KEYUP). FL_KEYBOARD is a synonym for FL_KEYDOWN, and both names are used interchangeably in this documentation.

The key can be found in `Fl::event_key()`. The text that the key should insert can be found with `Fl::event_text()` and its length is in `Fl::event_length()`.

If you use the key, then `handle()` should return 1. If you return zero then FLTK assumes you ignored the key and will then attempt to send it to a parent widget. If none of them want it, it will change the event into a FL_SHORTCUT event. FL_KEYBOARD events are also generated by the character palette/map.

To receive FL_KEYBOARD events you must also respond to the FL_FOCUS and FL_UNFOCUS events by returning 1. This way FLTK knows whether to bother sending your widget keyboard events. (Some widgets don't need them, e.g. `Fl_Box`.)

If you are writing a text-editing widget you may also want to call the `Fl::compose()` function to translate individual keystrokes into characters.

FL_KEYUP events are sent to the widget that currently has focus. This is not necessarily the same widget that received the corresponding FL_KEYDOWN event because focus may have changed between events.

Todo Add details on how to detect repeating keys, since on some X servers a repeating key will generate both FL_KEYUP and FL_KEYDOWN, such that to tell if a key is held, you need `Fl::event_key(int)` to detect if the key is being held down during FL_KEYUP or not.

1.9.4.2 FL_SHORTCUT

If the `Fl::focus()` widget is zero or ignores an FL_KEYBOARD event then FLTK tries sending this event to every widget it can, until one of them returns non-zero. FL_SHORTCUT is first sent to the `Fl::belowmouse()` widget, then its parents and siblings, and eventually to every widget in the window, trying to find an object that returns non-zero. FLTK tries really hard to not to ignore any keystrokes!

You can also make "global" shortcuts by using `Fl::add_handler()`. A global shortcut will work no matter what windows are displayed or which one has the focus.

Since version 1.4, FLTK has 3 default global shortcuts (`Ctrl+/-/0/` [`Cmd+/-/0/` under macOS]) that change the value of the GUI scaling factor. `Ctrl+` zooms-in all app windows of the focussed display (all displays under macOS); `Ctrl-` zooms-out these windows; `Ctrl 0` restores the initial value of the scaling factor. If any window of the display is fullscreen or maximized, scaling shortcuts have no effect. It's possible to deactivate FLTK's default scaling shortcuts with function `Fl::keyboard_screen_scaling()`.

Option `Fl::OPTION_SIMPLE_ZOOM_SHORTCUT` can facilitate the typing necessary to trigger the zoom-in operation with those keyboard layouts where character '+' is located in the shifted position of its key: when this option is On it's not necessary to press also the Shift key to zoom-in.

These scaling shortcuts are installed when the FLTK library opens the display. They have a lower priority than any shortcut defined in any menu and than any user-provided event handler (see `Fl::add_handler()`) installed after FLTK opened the display. Therefore, if a menu item of an FLTK app is given FL_COMMAND+'+' as shortcut, that item's callback rather than FLTK's default zooming-in routine is triggered when `Ctrl+` (`Cmd+` under macOS) is pressed.

FLTK sends the FL_ZOOM_EVENT when the scaling factor value changes, to which a callback can be associated with `Fl::add_handler()`. By default, FLTK displays the new scaling factor value in a yellow, transient window. This can be changed with option `Fl::OPTION_SHOW_SCALING`.

1.9.5 Widget Events

1.9.5.1 FL_DEACTIVATE

This widget is no longer active, due to `deactivate()` being called on it or one of its parents. Please note that although `active()` may still return true for this widget after receiving this event, it is only truly active if `active()` is true for both it and all of its parents. (You can use `active_r()` to check this).

1.9.5.2 FL_ACTIVATE

This widget is now active, due to `activate()` being called on it or one of its parents.

1.9.5.3 FL_HIDE

This widget is no longer visible, due to `hide()` being called on it or one of its parents, or due to a parent window being minimized. Please note that although `visible()` may still return true for this widget after receiving this event, it is only truly visible if `visible()` is true for both it and all of its parents. (You can use `visible_r()` to check this).

1.9.5.4 FL_SHOW

This widget is visible again, due to `show()` being called on it or one of its parents, or due to a parent window being restored. *A child `Fl_Window` will respond to this by actually creating the window if not done already, so if you subclass a window, be sure to pass `FL_SHOW` to the base class `handle()` method!*

Note

The events in this chapter ("Widget Events"), i.e. `FL_ACTIVATE`, `FL_DEACTIVATE`, `FL_SHOW`, and `FL_HIDE`, are the only events deactivated and invisible widgets can usually get, depending on their states. Under certain circumstances, there may also be `FL_LEAVE` or `FL_UNFOCUS` events delivered to deactivated or hidden widgets.

1.9.6 Clipboard Events

1.9.6.1 FL_PASTE

You should get this event some time after you call `Fl::paste()`. The contents of `Fl::event_text()` is the text to insert and the number of characters is in `Fl::event_length()`.

1.9.6.2 FL_SELECTIONCLEAR

The `Fl::selection_owner()` will get this event before the selection is moved to another widget. This indicates that some other widget or program has claimed the selection. Motif programs used this to clear the selection indication. Most modern programs ignore this.

1.9.7 Drag and Drop Events

FLTK supports drag and drop of text and files from any application on the desktop to an FLTK widget. Text is transferred using UTF-8 encoding.

See [Fl::dnd\(\)](#) for drag and drop from an FLTK widget.

The drag and drop data is available in [Fl::event_text\(\)](#) at the concluding `FL_PASTE`. On some platforms, the event text is also available for the `FL_DND_*` events, however application must not depend on that behavior because it depends on the protocol used on each platform.

`FL_DND_*` events cannot be used in widgets derived from [Fl_Group](#) or [Fl_Window](#).

1.9.7.1 Dropped filenames

Files are received as a list of full path and file names.

- On some X11 platforms, files are received as a URL-encoded UTF-8 string, that is, non-ASCII bytes (and a few others such as space and %) are replaced by the 3 bytes "%XY" where XY are the byte's hexadecimal value. The [fl_decode_uri\(\)](#) function can be used to transform in-place the received string into a proper UTF-8 string. On these platforms, strings corresponding to dropped files are further prepended by `file://` (or other prefixes such as `computer://`).
- Other X11 situations put all dropped filenames in a single line, separated by spaces.
- On non-X11 platforms, including Wayland, files dropped are received one pathname per line, with no `'\n'` after the last pathname.

1.9.7.2 FL_DND_ENTER

The mouse has been moved to point at this widget. A widget that is interested in receiving drag'n'drop data must return 1 to receive `FL_DND_DRAG`, `FL_DND_LEAVE` and `FL_DND_RELEASE` events.

1.9.7.3 FL_DND_DRAG

The mouse has been moved inside a widget while dragging data. A widget that is interested in receiving drag'n'drop data should indicate the possible drop position.

1.9.7.4 FL_DND_LEAVE

The mouse has moved out of the widget.

1.9.7.5 FL_DND_RELEASE

The user has released the mouse button dropping data into the widget. When the receiving widget's `handle()` method gets the `FL_DND_RELEASE` event, it should return 1 to accept the dragged data. Processing of this event must not use code that would make unrelated events be sent to the application (opening a dialog window for example) or that would communicate with the dragging process. The next event received by the `handle()` method will then be an `FL_PASTE` event. The `handle()` method should process this `FL_PASTE` event rapidly to prevent the dragging process from failing with a timeout error.

1.9.8 Other events

1.9.8.1 FL_SCREEN_CONFIGURATION_CHANGED

Sent whenever the screen configuration changes (a screen is added/removed, a screen resolution is changed, screens are moved). Use [Fl::add_handler\(\)](#) to be notified of this event.

1.9.8.2 FL_FULLSCREEN

The application window has been changed from normal to fullscreen, or from fullscreen to normal. If you are using a X window manager which supports Extended Window Manager Hints, this event will not be delivered until the change has actually happened.

1.9.9 Fl::event_*() methods

FLTK keeps the information about the most recent event in static storage. This information is good until the next event is processed. Thus it is valid inside `handle()` and `callback()` methods.

These are all trivial inline functions and thus very fast and small:

- [Fl::event_button\(\)](#)
- [Fl::event_clicks\(\)](#)
- [Fl::event_dx\(\)](#)
- [Fl::event_dy\(\)](#)
- [Fl::event_inside\(\)](#)
- [Fl::event_is_click\(\)](#)
- [Fl::event_key\(\)](#)
- [Fl::event_length\(\)](#)
- [Fl::event_state\(\)](#)
- [Fl::event_text\(\)](#)
- [Fl::event_x\(\)](#)
- [Fl::event_x_root\(\)](#)
- [Fl::event_y\(\)](#)
- [Fl::event_y_root\(\)](#)
- [Fl::get_key\(\)](#)
- [Fl::get_mouse\(\)](#)
- [Fl::test_shortcut\(\)](#)

1.9.10 Event Propagation

Widgets receive events via the virtual `handle()` function. The argument indicates the type of event that can be handled. The widget must indicate if it handled the event by returning 1. FLTK will then remove the event and wait for further events from the host. If the widget's handle function returns 0, FLTK may redistribute the event based on a few rules.

Most events are sent directly to the `handle()` method of the `Fl_Window` that the window system says they belong to. The window (actually the `Fl_Group` that `Fl_Window` is a subclass of) is responsible for sending the events on to any child widgets. To make the `Fl_Group` code somewhat easier, FLTK sends some events (`FL_DRAG`, `FL_RELEASE`, `FL_KEYBOARD`, `FL_SHORTCUT`, `FL_UNFOCUS`, and `FL_LEAVE`) directly to leaf widgets. These procedures control those leaf widgets:

- `Fl::add_handler()`
- `Fl::belowmouse()`
- `Fl::focus()`
- `Fl::grab()`
- `Fl::modal()`
- `Fl::pushed()`
- `Fl::release()` (deprecated, see `Fl::grab(0)`)
- `Fl_Widget::take_focus()`

FLTK propagates events along the widget hierarchy depending on the kind of event and the status of the UI. Some events are injected directly into the widgets, others may be resent as new events to a different group of receivers.

Mouse click events are first sent to the window that caused them. The window then forwards the event down the hierarchy until it reaches the widget that is below the click position. If that widget uses the given event, the widget is marked "pushed" and will receive all following mouse motion (`FL_DRAG`) events until the mouse button is released.

Mouse motion (`FL_MOVE`) events are sent to the `Fl::belowmouse()` widget, i.e. the widget that returned 1 on the last `FL_ENTER` event.

Mouse wheel events are sent to the window that caused the event. The window propagates the event down the tree, first to the widget that is below the mouse pointer, and if that does not succeed, to all other widgets in the group. This ensures that scroll widgets work as expected with the widget furthest down in the hierarchy getting the first opportunity to use the wheel event, but also giving scroll bars, that are not directly below the mouse a chance.

Keyboard events are sent directly to the widget that has keyboard focus. If the focused widget rejects the event, it is resent as a shortcut event, first to the top-most window, then to the widget below the mouse pointer, propagating up the hierarchy to all its parents. Those send the event also to all widgets that are not below the mouse pointer. Now if that did not work out, the shortcut is sent to all registered shortcut handlers.

If we are still unsuccessful, the event handler flips the case of the shortcut letter and starts over. Finally, if the key is "escape", FLTK sends a close event to the top-most window.

All other events are pretty much sent right away to the window that created the event.

Widgets can "grab" events. The grabbing window gets all events exclusively, but usually by the same rules as described above.

Windows can also request exclusivity in event handling by making the window modal.

1.9.11 FLTK Compose-Character Sequences

The character composition done by `Fl_Input` widget requires that you call the `Fl::compose()` function if you are writing your own text editor widget.

Currently, all characters made by single key strokes with or without modifier keys, or by system-defined character compose sequences (that can involve dead keys or a compose key) can be input. You should call `Fl::compose()` in case any enhancements to this processing are done in the future. The interface has been designed to handle arbitrary UTF-8 encoded text.

The following methods are provided for character composition:

- `Fl::compose()`
- `Fl::compose_reset()`

Under Mac OS X, FLTK "previews" partially composed sequences.

1.10 Adding and Extending Widgets

This chapter describes how to add your own widgets or extend existing widgets in FLTK.

1.10.1 Subclassing

New widgets are created by *subclassing* an existing FLTK widget, typically `Fl_Widget` for controls and `Fl_Group` for composite widgets.

A control widget typically interacts with the user to receive and/or display a value of some sort.

A composite widget holds a list of child widgets and handles moving, sizing, showing, or hiding them as needed. `Fl_Group` is the main composite widget class in FLTK, and all of the other composite widgets (`Fl_Pack`, `Fl_Scroll`, `Fl_Tabs`, `Fl_Tile`, `Fl_Window`, `Fl_Flex`, `Fl_Grid`, etc.) are subclasses of it.

You can also subclass other existing widgets to provide a different look or user-interface. For example, the button widgets are all subclasses of `Fl_Button` since they all interact with the user via a mouse button click. The only difference is the code that draws the face of the button.

1.10.2 Making a Subclass of `Fl_Widget`

Your subclasses can directly descend from `Fl_Widget` or any subclass of `Fl_Widget`. `Fl_Widget` has only four virtual methods, and overriding some or all of these may be necessary.

1.10.3 The Constructor

The constructor should have the following arguments:

```
MyClass(int x, int y, int w, int h, const char *label = 0);
```

This will allow the class to be used in FLUID without problems.

The constructor must call the constructor for the base class and pass the same arguments:

```
MyClass::MyClass(int x, int y, int w, int h, const char *label)
: FL_Widget(x, y, w, h, label) {
    // do initialization stuff...
}
```

`FL_Widget`'s protected constructor sets `x()`, `y()`, `w()`, `h()`, and `label()` to the passed values and initializes the other instance variables to:

```
type(0);
box(FL_NO_BOX);
color(FL_BACKGROUND_COLOR);
selection_color(FL_BACKGROUND_COLOR);
labeltype(FL_NORMAL_LABEL);
labelstyle(FL_NORMAL_STYLE);
labelsize(FL_NORMAL_SIZE);
labelcolor(FL_FOREGROUND_COLOR);
align(FL_ALIGN_CENTER);
callback(default_callback, 0);
flags(ACTIVE|VISIBLE);
image(0);
deimage(0);
```

1.10.4 Protected Methods of `FL_Widget`

The following methods are provided for subclasses to use:

- `clear_visible()`
- `damage()`
- `draw_box()`
- `draw_focus()`
- `draw_label()`
- `set_flag()`
- `set_visible()`
- `test_shortcut()`
- `type()`

```
void FL_Widget::damage(uchar mask)
```

```
void FL_Widget::damage(uchar mask, int x, int y, int w, int h)
```

```
uchar FL_Widget::damage()
```

The first form indicates that a partial update of the object is needed. The bits in `mask` are OR'd into `damage()`. Your `draw()` routine can examine these bits to limit what it is drawing. The public method `FL_Widget::redraw()` simply does `FL_Widget::damage(FL_DAMAGE_ALL)`, but the implementation of your widget can call the public `damage(n)`.

The second form indicates that a region is damaged. If only these calls are done in a window (no calls to `damage(n)`) then FLTK will clip to the union of all these calls before drawing anything. This can greatly speed up incremental displays. The mask bits are OR'd into `damage()` unless this is a [FL_Window](#) widget.

The third form returns the bitwise-OR of all `damage(n)` calls done since the last `draw()`.

When redrawing your widgets you should look at the damage bits to see what parts of your widget need redrawing. The `handle()` method can then set individual damage bits to limit the amount of drawing that needs to be done:

```
int MyClass::handle(int event) {
    // ...
    if (change_to_part1) damage(1);
    if (change_to_part2) damage(2);
    if (change_to_part3) damage(4);
}

void MyClass::draw() {
    if (damage() & FL_DAMAGE_ALL) {
        // ... draw frame/box and other static stuff ...
    }

    if (damage() & (FL_DAMAGE_ALL | 1)) draw_part1();
    if (damage() & (FL_DAMAGE_ALL | 2)) draw_part2();
    if (damage() & (FL_DAMAGE_ALL | 4)) draw_part3();
}
```

Todo Clarify [FL_Window::damage\(uchar\)](#) handling - seems confused/wrong? ORing value doesn't match setting behavior in [FL_Widget.H](#)!

```
void FL_Widget::draw_box() const
void FL_Widget::draw_box(FL_Boxtype t, FL_Color c) const
```

The first form draws this widget's `box()`, using the dimensions of the widget. The second form uses `t` as the box type and `c` as the color for the box.

```
void FL_Widget::draw_focus()
void FL_Widget::draw_focus(FL_Boxtype t, int x, int y, int w, int h) const
```

Draws a focus box inside the widget's bounding box. The second form allows you to specify a different bounding box.

```
void FL_Widget::draw_label() const
void FL_Widget::draw_label(int x, int y, int w, int h) const
void FL_Widget::draw_label(int x, int y, int w, int h, FL_Align align) const
```

The first form is the usual function for a `draw()` method to call to draw the widget's label. It does not draw the label if it is supposed to be outside the box (on the assumption that the enclosing group will draw those labels).

The second form uses the passed bounding box instead of the widget's bounding box. This is useful so "centered" labels are aligned with some feature, like a moving slider.

The third form draws the label anywhere. It acts as though `FL_ALIGN_INSIDE` has been forced on so the label will appear inside the passed bounding box. This is designed for parent groups to draw labels with.

```
void Fl_Widget::set_flag(int c)
```

Calling `set_flag(SHORTCUT_LABEL)` modifies the behavior of `draw_label()` so that `'&'` characters cause an underscore to be printed under the next letter.

```
void Fl_Widget::set_visible()
void Fl_Widget::clear_visible()
```

Fast inline versions of `Fl_Widget::hide()` and `Fl_Widget::show()`. These do not send the `FL_HIDE` and `FL_SHOW` events to the widget.

```
int Fl_Widget::test_shortcut()
static int Fl_Widget::test_shortcut(const char *s)
```

The first version tests `Fl_Widget::label()` against the current event (which should be a `FL_SHORTCUT` event). If the label contains a `'&'` character and the character after it matches the keypress, this returns true. This returns false if the `SHORTCUT_LABEL` flag is off, if the label is `NULL`, or does not have a `'&'` character in it, or if the keypress does not match the character.

The second version lets you do this test against an arbitrary string.

```
uchar Fl_Widget::type() const
void Fl_Widget::type(uchar t)
```

The property `Fl_Widget::type()` can return an arbitrary 8-bit identifier, and can be set with the protected method `type(uchar t)`. This value had to be provided for Forms compatibility, but you can use it for any purpose you want. Try to keep the value less than 100 to not interfere with reserved values.

FLTK does not use RTTI (Run Time Typing Information) to enhance portability. But this may change in the near future if RTTI becomes standard everywhere.

If you don't have RTTI you can use the clumsy FLTK mechanism, by having `type()` use a unique value. These unique values must be greater than the symbol `FL_RESERVED_TYPE` (which is 100) and less than `FL_WINDOW` (unless you make a subclass of `Fl_Window`). Look through the header files for `FL_RESERVED_TYPE` to find an unused number. If you make a subclass of `Fl_Window` you must use `FL_WINDOW + n` (where `n` must be in the range 1 to 7).

1.10.5 Handling Events

The virtual method `Fl_Widget::handle(int event)` is called to handle each event passed to the widget. It can:

- Change the state of the widget.
- Call `Fl_Widget::redraw()` if the widget needs to be redisplayed.
- Call `Fl_Widget::damage(uchar c)` if the widget needs a partial-update (assuming you provide support for this in your `draw()` method).
- Call `Fl_Widget::do_callback()` if a callback should be generated.
- Call `Fl_Widget::handle()` on child widgets.

Events are identified by the integer argument. Other information about the most recent event is stored in static locations and acquired by calling the `Fl::event_*` methods. This information remains valid until another event is handled.

Here is a sample `handle()` method for a widget that acts as a pushbutton and also accepts the keystroke 'x' to cause the callback:

```
int MyClass::handle(int event) {
    switch (event) {
        case FL_PUSH:
            highlight = 1;
            redraw();
            return 1;
        case FL_DRAG: {
            int t = Fl::event_inside(this);
            if (t != highlight) {
                highlight = t;
                redraw();
            }
        }
        return 1;
        case FL_RELEASE:
            if (highlight) {
                highlight = 0;
                redraw();
                do_callback();
                // never do anything after a callback, as the callback
                // may delete the widget!
            }
            return 1;
        case FL_SHORTCUT:
            if (Fl::event_key() == 'x') {
                do_callback();
                return 1;
            }
            return 0;
        default:
            return Fl_Widget::handle(event);
    }
}
```

You must return non-zero if your `handle()` method uses the event. If you return zero, the parent widget will try sending the event to another widget.

For debugging purposes, event numbers can be printed as their actual event names using the `fl_eventnames[]` array, e.g.:

```
#include <FL/names.h>           // defines fl_eventnames[]
[...
int MyClass::handle(int e) {
    printf("Event was %s (%d)\n", fl_eventnames[e], e);    // e.g. "Event was FL_PUSH (1)"
    [...]
```

1.10.6 Drawing the Widget

The `draw()` virtual method is called when FLTK wants you to redraw your widget. It will be called if and only if `damage()` is non-zero, and `damage()` will be cleared to zero after it returns. The `draw()` method should be declared protected so that it can't be called from non-drawing code.

The `damage()` value contains the bitwise-OR of all the `damage(n)` calls to this widget since it was last drawn. This can be used for minimal update, by only redrawing the parts whose bits are set. FLTK will turn on the `FL_DAMAGE_ALL` bit if it thinks the entire widget must be redrawn, e.g. for an expose event.

Expose events (and the `damage(mask,x,y,w,h)` function described above) will cause `draw()` to be called with FLTK's clipping turned on. You can greatly speed up redrawing in some cases by testing `fl_not_clipped(x,y,w,h)` or `fl_clip_box()` and skipping invisible parts.

Besides the protected methods described above, FLTK provides a large number of basic drawing functions, which are described in the chapter [Drawing Things in FLTK](#).

1.10.7 Resizing the Widget

The `resize(x,y,w,h)` method is called when the widget is being resized or moved. The arguments are the new position, width, and height. `x()`, `y()`, `w()`, and `h()` still remain the old size. You must call `resize()` on your base class with the same arguments to get the widget size to actually change.

This should *not* call `redraw()`, at least if only the `x()` and `y()` change. This is because composite widgets like [Fl_Scroll](#) may have a more efficient way of drawing the new position.

1.10.8 Making a Composite Widget

A "composite" widget contains one or more "child" widgets. To make a composite widget you should subclass [Fl_Group](#). It is possible to make a composite object that is not a subclass of [Fl_Group](#), but you'll have to duplicate the code in [Fl_Group](#) anyways.

Instances of the child widgets may be included in the parent:

```
class MyClass : public Fl_Group {
    Fl_Button the_button;
    Fl_Slider the_slider;
    ...
};
```

The constructor has to initialize these instances. They are automatically added to the group, since the [Fl_Group](#) constructor does `Fl_Group::begin()`. *Don't forget to call `Fl_Group::end()` or use the `Fl_End` pseudo-class:*

```
MyClass::MyClass(int x, int y, int w, int h) :
    Fl_Group(x, y, w, h),
    the_button(x + 5, y + 5, 100, 20),
    the_slider(x, y + 50, w, 20)
{
    ...(you could add dynamically created child widgets here)...
    end(); // don't forget to do this!
}
```

The child widgets need callbacks. These will be called with a pointer to the children, but the widget itself may be found in the `parent()` pointer of the child. Usually these callbacks can be static private methods, with a matching private method:

```
void MyClass::static_slider_cb(Fl_Widget* v, void *) { // static method
    (MyClass*)(v->parent())->slider_cb();
}
void MyClass::slider_cb() { // normal method
    use(the_slider->value());
}
```

If you make the `handle()` method, you can quickly pass all the events to the children using the `Fl_Group::handle()` method. You don't need to override `handle()` if your composite widget does nothing other than pass events to the children:

```
int MyClass::handle(int event) {
    if (Fl_Group::handle(event)) return 1;
    ... handle events that children don't want ...
}
```

If you override `draw()` you need to draw all the children. If `redraw()` or `damage()` is called on a child, `damage(FL_DAMAGE_CHILD)` is done to the group, so this bit of `damage()` can be used to indicate that a child needs to be drawn. It is fastest if you avoid drawing anything else in this case:

```
void MyClass::draw() {
    Fl_Widget *const*a = array();
    if (damage() == FL_DAMAGE_CHILD) { // only redraw some children
        for (int i = children(); i--; a++) {
            update_child(**a);
        }
    } else { // total redraw
        // ... draw background graphics ...
        // now draw all the children atop the background:
        for (int i = children(); i--; a++) {
            draw_child(**a);
            draw_outside_label(**a); // you may not need to do this
        }
    }
}
```

`Fl_Group` provides some protected methods to make drawing easier:

- `draw_child()`
- `draw_children()`
- `draw_outside_label()`
- `update_child()`

void `Fl_Group::draw_child(Fl_Widget &widget) const`

This will force the child's `damage()` bits all to one and call `draw()` on it, then clear the `damage()`. You should call this on all children if a total redraw of your widget is requested, or if you draw something (like a background box) that damages the child. Nothing is done if the child is not `visible()` or if it is clipped.

void `Fl_Group::draw_children()`

A convenience function that draws all children of the group. This is useful if you derived a widget from `Fl_Group` and want to draw a special border or background. You can call `draw_children()` from the derived `draw()` method after drawing the box, border, or background.

void `Fl_Group::draw_outside_label(const Fl_Widget &widget) const`

Draw the labels that are *not* drawn by `draw_label()`. If you want more control over the label positions you might want to call `child->draw_label(x, y, w, h, a)`.

void `Fl_Group::update_child(Fl_Widget& widget) const`

Draws the child only if its `damage()` is non-zero. You should call this on all the children if your own `damage` is equal to `FL_DAMAGE_CHILD`. Nothing is done if the child is not `visible()` or if it is clipped.

1.10.9 Cut and Paste Support

FLTK provides routines to cut and paste UTF-8 encoded text between applications:

- [Fl::copy\(\)](#)
- [Fl::paste\(\)](#)
- [Fl::selection\(\)](#)
- [Fl::selection_owner\(\)](#)

It is also possible to copy and paste image data between applications:

- [Fl_Copy_Surface](#)
- [Fl::clipboard_contains\(\)](#)
- [Fl::paste\(\)](#)

It may be possible to cut/paste other kinds of data by using [Fl::add_handler\(\)](#). Note that handling events beyond those provided by FLTK may be operating system specific. See [Operating System Issues](#) for more details.

1.10.10 Drag And Drop Support

FLTK provides routines to drag and drop UTF-8 encoded text between applications:

Drag'n'drop operations are initiated by copying data to the clipboard and calling the function [Fl::dnd\(\)](#).

Drop attempts are handled via the following events, already described under [Drag and Drop Events](#) in a previous chapter:

- `FL_DND_ENTER`
- `FL_DND_DRAG`
- `FL_DND_LEAVE`
- `FL_DND_RELEASE`
- `FL_PASTE`

1.10.11 Making a subclass of `Fl_Window`

You may want your widget to be a subclass of [Fl_Window](#), [Fl_Double_Window](#), or [Fl_Gl_Window](#). This can be useful if your widget wants to occupy an entire window, and can also be used to take advantage of system-provided clipping, or to work with a library that expects a system window ID to indicate where to draw.

Subclassing [Fl_Window](#) is almost exactly like subclassing [Fl_Group](#), and in fact you can easily switch a subclass back and forth. Watch out for the following differences:

1. [Fl_Window](#) is a subclass of [Fl_Group](#) so *make sure your constructor calls* `end()` unless you actually want children added to your window.
2. When handling events and drawing, the upper-left corner is at 0,0, not `x()`, `y()` as in other [Fl_Widget](#)'s. For instance, to draw a box around the widget, call `draw_box(0, 0, w(), h())`, rather than `draw_box(x(), y(), w(), h())`.

You may also want to subclass [Fl_Window](#) in order to get access to different visuals or to change other attributes of the windows. See the [Operating System Issues](#) chapter for more information.

1.11 Using OpenGL

This chapter discusses using FLTK for your OpenGL applications.

1.11.1 Using OpenGL in FLTK

The easiest way to make an OpenGL display is to subclass [Fl_Gl_Window](#). Your subclass must implement a `draw()` method which uses OpenGL calls to draw the display. Your main program should call `redraw()` when the display needs to change, and (somewhat later) FLTK will call `draw()`.

With a bit of care you can also use OpenGL to draw into normal FLTK windows (see [Using OpenGL in Normal FLTK Windows](#) below). This allows you to use Gouraud shading for drawing your widgets. To do this you use the `gl_start()` and `gl_finish()` functions around your OpenGL code.

You must include FLTK's `<FL/gl.h>` header file. It will include the file `<GL/gl.h>` (on macOS: `<OpenGL/gl.h>`), define some extra drawing functions provided by FLTK, and include the `<windows.h>` header file needed by Windows applications.

Some simple coding rules (see [OpenGL and support of HighDPI displays](#)) allow to write cross-platform code that will support OpenGL run on HighDPI displays (including the 'retina' displays of Apple hardware).

1.11.2 Making a Subclass of Fl_Gl_Window

To make a subclass of [Fl_Gl_Window](#), you must provide:

- A class definition.
- A `draw()` method.
- A `handle()` method if you need to receive input from the user.

If your subclass provides static controls in the window, they must be redrawn whenever the `FL_DAMAGE_ALL` bit is set in the value returned by `damage()`.

1.11.2.1 Defining the Subclass

To define the subclass you just subclass the [Fl_Gl_Window](#) class:

```
class MyWindow : public Fl_Gl_Window {
    void draw();
    int handle(int);

public:
    MyWindow(int X, int Y, int W, int H, const char *L)
        : Fl_Gl_Window(X, Y, W, H, L) {}
};
```

The `draw()` and `handle()` methods are described below. Like any widget, you can include additional private and public data in your class (such as scene graph information, etc.)

1.11.2.2 The draw() Method

The `draw()` method is where you actually do your OpenGL drawing:

```
void MyWindow::draw() {
    if (!valid()) {
        ... set up projection, viewport, etc ...
        ... window size is in w() and h().
        ... valid() is turned on by FLTK after draw() returns
    }
    ... draw ...
}
```

1.11.2.3 The handle() Method

The `handle()` method handles mouse and keyboard events for the window:

```
int MyWindow::handle(int event) {
    switch(event) {
        case FL_PUSH:
            ... mouse down event ...
            ... position in Fl::event_x() and Fl::event_y()
            return 1;
        case FL_DRAG:
            ... mouse moved while down event ...
            return 1;
        case FL_RELEASE:
            ... mouse up event ...
            return 1;
        case FL_FOCUS :
        case FL_UNFOCUS :
            ... Return 1 if you want keyboard events, 0 otherwise
            return 1;
        case FL_KEYBOARD:
            ... keypress, key is in Fl::event_key(), ascii in Fl::event_text()
            ... Return 1 if you understand/use the keyboard event, 0 otherwise...
            return 1;
        case FL_SHORTCUT:
            ... shortcut, key is in Fl::event_key(), ascii in Fl::event_text()
            ... Return 1 if you understand/use the shortcut event, 0 otherwise...
            return 1;
        default:
            // pass other events to the base class...
            return Fl_Gl_Window::handle(event);
    }
}
```

When `handle()` is called, the OpenGL context is not set up! If your display changes, you should call `redraw()` and let `draw()` do the work. Don't call any OpenGL drawing functions from inside `handle()` !

You can call *some* OpenGL stuff like hit detection and texture loading functions by doing:

```
case FL_PUSH:
    make_current();           // make OpenGL context current
    if (!valid()) {

        ... set up projection exactly the same as draw ...

        valid(1);            // stop it from doing this next time
    }
    ... ok to call NON-DRAWING OpenGL code here, such as hit
    detection, loading textures, etc...
```

Your main program can now create one of your windows by doing `new MyWindow(...)`.

You can also use your new window class in FLUID by:

1. Putting your class definition in a `MyWindow.H` file.
2. Creating a `Fl_Box` widget in FLUID.
3. In the widget panel fill in the "class" field with `MyWindow`. This will make FLUID produce constructors for your new class.
4. In the "Extra Code" field put `#include "MyWindow.H"`, so that the FLUID output file will compile.

You must put `glwindow->show()` in your main code after calling `show()` on the window containing the OpenGL window.

1.11.3 OpenGL and support of HighDPI displays

HighDPI displays (including the so-called 'retina' displays of Apple hardware) are supported by FLTK in such a way that 1 unit of an FLTK quantity (say, the value given by `Fl_Gl_Window::w()`) corresponds to more than 1 pixel on the display. Conversely, when a program specifies the width and height of the OpenGL viewport, it is necessary to use an API that returns quantities expressed in pixels. That can be done as follows:

```
Fl_Gl_Window *glw = ...;
glViewport(0, 0, glw->pixel_w(), glw->pixel_h());
```

which makes use of the `Fl_Gl_Window::pixel_w()` and `Fl_Gl_Window::pixel_h()` methods giving the size in pixels of an `Fl_Gl_Window` that is potentially mapped to a HighDPI display. Method `Fl_Gl_Window::pixels_per_unit()` can also be useful in this context.

Note

A further coding rule is necessary to properly support retina displays and OpenGL under macOS (see [OpenGL and 'retina' displays](#))

1.11.4 Using OpenGL in Normal FLTK Windows

Note

Drawing both with OpenGL and Quartz in a normal FLTK window is not possible with the macOS platform. This technique is therefore not useful under macOS because it permits nothing more than what is possible with class `Fl_Gl_Window`.

You can put OpenGL code into the `draw()` method, as described in [Drawing the Widget](#) in the previous chapter, or into the code for a `boxtype` or other places with some care.

Most importantly, before you show *any* windows, including those that don't have OpenGL drawing, you **must** initialize FLTK so that it knows it is going to use OpenGL. You may use any of the symbols described for `Fl_Gl_Window`↔
: :`mode()` to describe how you intend to use OpenGL:

```
Fl::gl_visual(FL_RGB);
```

You can then put OpenGL drawing code anywhere you can draw normally by surrounding it with `gl_start()` and `gl_finish()` to set up, and later release, an OpenGL context with an orthographic projection so that 0,0 is the lower-left corner of the window and each pixel is one unit. The current clipping is reproduced with OpenGL `glScissor()` commands. These functions also synchronize the OpenGL graphics stream with the drawing done by other X, Windows, or FLTK functions.

```
gl_start();
... put your OpenGL code here ...
gl_finish();
```

The same context is reused each time. If your code changes the projection transformation or anything else you should use `glPushMatrix()` and `glPopMatrix()` functions to put the state back before calling `gl_finish()`.

You may want to use `Fl_Window::current()->h()` to get the drawable height so that you can flip the Y coordinates.

Unfortunately, there are a bunch of limitations you must adhere to for maximum portability:

- You must choose a default visual with `Fl::gl_visual()`.
- You cannot pass `FL_DOUBLE` to `Fl::gl_visual()`.
- You cannot use `Fl_Double_Window` or `Fl_Overlay_Window`.

Do *not* call `gl_start()` or `gl_finish()` when drawing into an `Fl_Gl_Window` !

1.11.5 Using FLTK widgets in OpenGL Windows

FLTK widgets can be added to `Fl_Gl_Window`s just as they would be added to `Fl_Window`s. They are rendered as an overlay over the user defined OpenGL graphics using 'fl_.' graphics calls that are implemented in GL.

`Fl_Gl_Window` does not add subsequent widgets as children by default as `Fl_Window` does. Call `myGlWindow->begin()` after creating the GL window to automatically add following widgets. Remember to call `myGlWindow->end()`.

```
class My_Gl_Window : public Fl_Gl_Window {
...
    void draw();
...
};

...
myGlWindow = new My_Gl_Window(0, 0, 500, 500);
myGlWindow->begin();
myButton = new Fl_Button(10, 10, 120, 24, "Hello!");
myGlWindow->end();
...

void My_Gl_Window::draw() {
    // ... user GL drawing code
    Fl_Gl_Window::draw(); // Draw FLTK child widgets.
}
```

Users can draw into the overlay by using GL graphics calls as well as all `fl_.` graphics calls from the "Drawing Fast Shapes" section.

```
void My_Gl_Window::draw() {
    // ... user GL drawing code
    Fl_Gl_Window::draw_begin(); // Set up 1:1 projection
    Fl_Window::draw();          // Draw FLTK children
    fl_color(FL_RED);
    fl_rect(10, 10, 100, 100);
    Fl_Gl_Window::draw_end();   // Restore GL state
}
```

Widgets can be drawn with transparencies by assigning an alpha value to a colormap entry and using that color in the widget.

```
Fl::set_color(FL_FREE_COLOR, 255, 255, 0, 127); // 50% transparent yellow
myGlWindow = new My_Gl_Window(0, 0, 500, 500);
myGlWindow->begin();
myButton = new Fl_Button(10, 10, 120, 24, "Hello!");
myButton->box(FL_BORDER_BOX);
myButton->color(FL_FREE_COLOR);
myGlWindow->end();
```

Transparencies can also be set directly when drawing. This can be used to create custom box types and RGB overlay drawings with an alpha channel.

```
fl_color(0, 255, 0, 127); // 50% transparent green
fl_rectf(10, 10, 100, 100);
fl_color(FL_RED); // back to opaque red
fl_rect(20, 20, 80, 80);
```

1.11.6 OpenGL Drawing Functions

FLTK provides some useful OpenGL drawing functions. They can be freely mixed with any OpenGL calls, and are defined by including `<FL/gl.h>` which you should include instead of the OpenGL header `<GL/gl.h>`.

void `gl_color(Fl_Color)`

Sets the current OpenGL color to a FLTK color. *For color-index modes it will use `fl_xpixel(c)`, which is only right if this window uses the default colormap!*

void `gl_rect(int x, int y, int w, int h)`
 void `gl_rectf(int x, int y, int w, int h)`

Outlines or fills a rectangle with the current color. If `Fl_Gl_Window::ortho()` has been called, then the rectangle will exactly fill the pixel rectangle passed.

void `gl_font(Fl_Font fontid, int size)`

Sets the current OpenGL font to the same font you get by calling `fl_font()`.

int `gl_height()`
 int `gl_descent()`
 float `gl_width(const char *s)`
 float `gl_width(const char *s, int n)`
 float `gl_width(uchar c)`

Returns information about the current OpenGL font.

void `gl_draw(const char *s)`
 void `gl_draw(const char *s, int n)`

Draws a nul-terminated string or an array of `n` characters in the current OpenGL font at the current raster position.

void `gl_draw(const char *s, int x, int y)`
 void `gl_draw(const char *s, int n, int x, int y)`
 void `gl_draw(const char *s, float x, float y)`
 void `gl_draw(const char *s, int n, float x, float y)`

Draws a nul-terminated string or an array of `n` characters in the current OpenGL font at the given position.

void `gl_draw(const char *s, int x, int y, int w, int h, Fl_Align)`

Draws a string formatted into a box, with newlines and tabs expanded, other control characters changed to `^X`, and aligned with the edges or center. Exactly the same output as `fl_draw()`.

1.11.7 Speeding up OpenGL

Performance of `Fl_Gl_Window` may be improved on some types of OpenGL implementations, in particular MESA and other software emulators, by setting the `GL_SWAP_TYPE` environment variable. This variable declares what is in the backbuffer after you do a `swapbuffers`.

- `setenv GL_SWAP_TYPE COPY`

This indicates that the back buffer is copied to the front buffer, and still contains its old data. This is true of many hardware implementations. Setting this will speed up emulation of overlays, and widgets that can do partial update can take advantage of this as `damage()` will not be cleared to -1.

- `setenv GL_SWAP_TYPE NODAMAGE`

This indicates that nothing changes the back buffer except drawing into it. This is true of MESA and Win32 software emulation and perhaps some hardware emulation on systems with lots of memory.

- All other values for `GL_SWAP_TYPE`, and not setting the variable, cause FLTK to assume that the back buffer must be completely redrawn after a swap.

This is easily tested by running the `gl_overlay` demo program and seeing if the display is correct when you drag another window over it or if you drag the window off the screen and back on. You have to exit and run the program again for it to see any changes to the environment variable.

1.11.8 Using OpenGL Optimizer with FLTK

OpenGL Optimizer is a scene graph toolkit for OpenGL available from Silicon Graphics for IRIX and Microsoft Windows. It allows you to view large scenes without writing a lot of OpenGL code.

OptimizerWindow Class Definition

To use **OpenGL Optimizer** with FLTK you'll need to create a subclass of `Fl_Gl_Window` that includes several state variables:

```
class OptimizerWindow : public Fl_Gl_Window {
    csContext *context_; // Initialized to 0 and set by draw()...
    csDrawAction *draw_action_; // Draw action...
    csGroup *scene_; // Scene to draw...
    csCamera *camera_; // Viewport for scene...

    void draw();

public:
    OptimizerWindow(int X, int Y, int W, int H, const char *L)
        : Fl_Gl_Window(X, Y, W, H, L) {
        context_ = (csContext *)0;
        draw_action_ = (csDrawAction *)0;
        scene_ = (csGroup *)0;
        camera_ = (csCamera *)0;
    }

    void scene(csGroup *g) { scene_ = g; redraw(); }

    void camera(csCamera *c) {
        camera_ = c;
        if (context_) {
            draw_action_>setCamera(camera_);
            camera_>draw(draw_action_);
            redraw();
        }
    }
};
```

The camera() Method

The `camera()` method sets the camera (projection and viewpoint) to use when drawing the scene. The scene is redrawn after this call.

The draw() Method

The `draw()` method performs the needed initialization and does the actual drawing:

```
void OptimizerWindow::draw() {
    if (!context_) {
        // This is the first time we've been asked to draw; create the
        // Optimizer context for the scene...

#ifdef _WIN32
        context_ = new csContext((HDC)fl_getHDC());
        context_>ref();
        context_>makeCurrent((HDC)fl_getHDC());
#else
        context_ = new csContext(fl_display, fl_visual);
        context_>ref();
        context_>makeCurrent(fl_display, fl_window);
#endif // _WIN32

        ... perform other context setup as desired ...

        // Then create the draw action to handle drawing things...

        draw_action_ = new csDrawAction;
        if (camera_) {
            draw_action_>setCamera(camera_);
            camera_>draw(draw_action_);
        }
    } else {
#ifdef _WIN32
        context_>makeCurrent((HDC)fl_getHDC());
#else
        context_>makeCurrent(fl_display, fl_window);
#endif // _WIN32
    }

    if (!valid()) {
        // Update the viewport for this context...
        context_>setViewport(0, 0, w(), h());
    }

    // Clear the window...
    context_>clear(csContext::COLOR_CLEAR | csContext::DEPTH_CLEAR,
        0.0f,          // Red
        0.0f,          // Green
        0.0f,          // Blue
        1.0f);         // Alpha

    // Then draw the scene (if any)...
    if (scene_)
        draw_action_>apply(scene_);
}
```

The scene() Method

The `scene()` method sets the scene to be drawn. The scene is a collection of 3D objects in a `csGroup`. The scene is redrawn after this call.

1.11.9 Using OpenGL 3.0 (or higher versions)

The examples subdirectory contains `OpenGL3test.cxx`, a toy program showing how to use OpenGL 3.0 (or higher versions) with FLTK in a cross-platform fashion. It contains also `OpenGL3-glut-test.cxx` which shows how to use FLTK's GLUT compatibility and OpenGL 3.

To access OpenGL 3.0 (or higher versions), use the `FL_OPENGL3` flag when calling `FL_Gl_Window::mode(int a)` or `glutInitDisplayMode()`.

On the Windows and Linux platforms, FLTK creates contexts implementing the highest OpenGL version supported by the hardware. Such contexts may also be compatible with lower OpenGL versions. Access to functions from OpenGL versions above 1.1 requires to load function pointers at runtime on these platforms. FLTK recommends to use the GLEW library to perform this. It is therefore necessary to install the GLEW library (see below).

On the macOS platform, MacOS 10.7 or above is required; GLEW is possible but not necessary. FLTK creates contexts for OpenGL versions 1 and 2 without the `FL_OPENGL3` flag and for OpenGL versions 3.2 and above (**but not below**) with it.

GLEW installation (Linux and Windows platforms)

FLTK needs a header file, `GL/glew.h`, and a library, `libGLEW.*` or equivalent, to support OpenGL 3 and above.

These can be obtained for most Linux distributions by installing package `libglew-dev`.

For the Windows platform :

- the header and a Visual Studio static library (`glew32.lib`) can be downloaded from <http://glew.sourceforge.net/> ;
- a MinGW-style static library (`libglew32.a`) can be built from source (same web site) with the `make` command. Alternatively, pre-built files are available for these architectures :
 - x86: download files `glew.h` and `libglew32.a`;
 - x86_64: install GLEW as an MSYS2 package with command :
`pacman -S mingw-w64-x86_64-glew`

Source-level changes for OpenGL 3:

- Put this in all OpenGL-using source files (instead of, or before if needed, `#include <FL/gl.h>`, and before `#include <FL/glut.h>` if you use GLUT):

```
#if defined(__APPLE__)
# include <OpenGL/gl3.h> // defines OpenGL 3.0+ functions
#else
# if defined(_WIN32)
#   define GLEW_STATIC 1
# endif
# include <GL/glew.h>
#endif
```

- Add the `FL_OPENGL3` flag when calling `FL_Gl_Window::mode(int a)` or `glutInitDisplayMode()`.
- Put this in the `handle(int event)` member function of the first to be created among your `FL_Gl_Window`-derived classes:

```
#ifndef __APPLE__
static int first = 1;
if (first && event == FL_SHOW && shown()) {
    first = 0;
    make_current();
    glewInit(); // defines pters to functions of OpenGL V 1.2 and above
}
#endif
```

- Alternatively, if you use GLUT, put

```
#ifndef __APPLE__
    glewInit(); // defines pters to functions of OpenGL V 1.2 and above
#endif
```

after the first `glutCreateWindow()` call.

If GLEW is installed on the Mac OS development platform, it is possible to use the same code for all platforms, with one exception: put

```
#ifdef __APPLE__
glewExperimental = GL_TRUE;
#endif
```

before the `glewInit()` call.

Testing for success of the `glewInit()` call

Testing whether the `glewInit()` call is successful is to be done as follows:

```
#include <FL/platform.H> // defines FLTK_USE_WAYLAND under the Wayland platform
#include <FL/Fl.H> // for Fl::warning()
#ifdef __APPLE__
# if defined(_WIN32)
#   define GLEW_STATIC 1
# endif
# include <GL/glew.h>

GLenum err = glewInit(); // defines pters to functions of OpenGL V 1.2 and above
# ifdef FLTK_USE_WAYLAND
    // glewInit returns GLEW_ERROR_NO_GLX_DISPLAY with Wayland
    if (fl_wl_display() && err == GLEW_ERROR_NO_GLX_DISPLAY) err = GLEW_OK;
# endif
if (err != GLEW_OK) Fl::warning("glewInit() failed returning %u", err);
#endif // ! __APPLE__
```

Changes in the build process

Link with `libGLEW.so` (with X11 or Wayland), `libglew32.a` (with MinGW) or `glew32.lib` (with MS Visual Studio); no change is needed on the Mac OS platform.

1.12 FLTK Runtime Options

In this chapter, we will cover how to access and alter settings for applications created using FLTK, both as an administrator and as a regular user.

Subchapters:

- [Runtime Options](#)
- [Obtaining Current Settings](#)
- [Administrative Tool](#)
- [List of Options](#)

1.12.1 Runtime Options

FLTK keeps track of various aspects of the user interface in a system-wide database. Users have the ability to set their own preferences and override default or system settings. For instance, FLTK will display a dotted rectangle around the widget with current focus. This might not be desirable for users who do not use keyboard navigation and do not need the rectangle. This can be turned off by setting the `OPTION_VISIBLE_FOCUS` option to 'off' for that user, which will disable the focus rectangle in all FLTK-based applications.

1.12.2 Obtaining Current Settings

Options are kept in preference files using the signature `Fl_Preferences::CORE_SYSTEM`, `"fltk.org"`, `"fltk"` for system-wide settings and `Fl_Preferences::CORE_USER`, `"fltk.org"`, `"fltk"` for individual users. They can be accessed by using the function `bool Fl::option(Fl_Option opt)`. If an application needs to temporarily override user or system settings, it can use the function `void option(Fl_Option opt, bool val)`.

To make changes to options permanently, FLTK provides an administrative tool called `fltk-options`.

1.12.3 Administrative Tool

`fltk-options` is a hybrid app that is part of FLTK and can be installed on the target system. It includes an up-to-date man page.

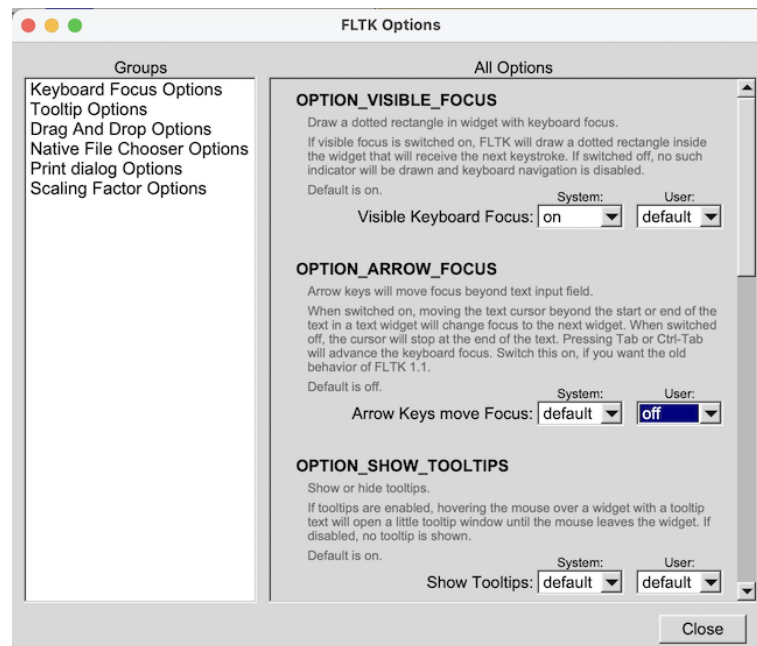


Figure 1.29 fltk-options Application

When `fltk-options` is called without any command-line arguments, it opens in interactive mode and provides a user interface to view and alter all system and current user options.

Starting the tool from a shell, the command-line options `-S` and `-U` can be used to display or change system or user options. On MS-Windows, `fltk-options` is also available as `fltk-options-cmd.exe`.

1.12.4 List of Options

Calling `fltk-options --help` gives a list of all available commands, and options and their values. `fltk-options --help OPTION` prints more detailed information for `OPTION` if available. In interactive mode, tooltips provide this additional information.

A full list of options can be found in the manual at [Fl::Fl_Option](#).

1.13 Advanced FLTK

This chapter explains advanced programming and design topics that will help you to get the most out of FLTK.

1.13.1 Multithreading

FLTK can be used to implement a GUI for a multithreaded application but, as with multithreaded programming generally, there are some concepts and caveats that must be kept in mind.

Key amongst these is that, for many of the target platforms on which FLTK is supported, only the `main()` thread of the process is permitted to handle system events, create or destroy windows and open or close windows. Further, only the `main()` thread of the process can safely write to the display.

To support this in a portable way, all FLTK `draw()` methods are executed in the `main()` thread. A worker thread may update the state of an existing widget, but it may not do any rendering directly, nor create or destroy a window. (**NOTE:** A special case exists for [Fl_Gl_Window](#) where it can, with suitable precautions, be possible to safely render to an existing GL context from a worker thread.)

Creating portable threads

We do not provide a threading interface as part of the library. A simple example showing how threads can be implemented, for all supported platforms, can be found in `test/threads.h` and `test/threads.cxx`.

FLTK has been used with a variety of thread interfaces, so if the simple example shown in `test/threads.cxx` does not cover your needs, you might want to select a third-party library that provides the features you require.

1.13.2 FLTK multithread locking - `Fl::lock()` and `Fl::unlock()`

In a multithreaded program, drawing of widgets (in the `main()` thread) happens asynchronously to widgets being updated by worker threads, so no drawing can occur safely whilst a widget is being modified (and no widget should be modified whilst drawing is in progress).

FLTK supports multithreaded applications using a locking mechanism internally. This allows a worker thread to lock the rendering context, preventing any drawing from taking place, whilst it changes the value of its widget.

Note

The converse is also true; whilst a worker thread holds the lock, the `main()` thread may not be able to process any drawing requests, nor service any events. So a worker thread that holds the FLTK lock **must** contrive to do so for the shortest time possible or it could impair operation of the application.

The lock operates broadly as follows.

Using the FLTK library, the `main()` thread holds the lock whenever it is processing events or redrawing the display. It acquires (locks) and releases (unlocks) the FLTK lock automatically and no "user intervention" is required. Indeed, a function that runs in the context of the `main()` thread ideally should **not** acquire / release the FLTK lock explicitly. (Though note that the lock calls are recursive, so calling `Fl::lock()` from a thread that already holds the lock, including the `main()` thread, is benign. The only constraint is that every call to `Fl::lock()` **must** be balanced by a corresponding call to `Fl::unlock()` to ensure the lock count is preserved.)

The `main()` thread **must** call `Fl::lock()` **once** before any windows are shown, to enable the internal lock (it is "off" by default since it is not useful in single-threaded applications) but thereafter the `main()` thread lock is managed by the library internally.

A worker thread, when it wants to alter the value of a widget, can acquire the lock using `Fl::lock()`, update the widget, then release the lock using `Fl::unlock()`. Acquiring the lock ensures that the worker thread can update the widget, without any risk that the `main()` thread will attempt to redraw the widget whilst it is being updated.

Note that acquiring the lock is a blocking action; the worker thread will stall for as long as it takes to acquire the lock. If the `main()` thread is engaged in some complex drawing operation this may block the worker thread for a long time, effectively serializing what ought to be parallel operations. (This frequently comes as a surprise to coders less familiar with multithreaded programming issues; see the discussion of "lockless programming" later for strategies for managing this.)

To incorporate the locking mechanism in the library, FLTK must be compiled with `--enable-threads` set during the `configure` process. IDE-based versions of FLTK are automatically compiled with the locking mechanism incorporated if possible. Since version 1.3, the `configure` script that builds the FLTK library also sets `--enable-threads` by default.

1.13.3 Simple multithreaded examples using Fl::lock

In `main()`, call `Fl::lock()` once before `Fl::run()` or `Fl::wait()` to enable the lock and start the runtime multithreading support for your program. All callbacks and derived functions like `handle()` and `draw()` will now be properly locked.

This might look something like this:

```
int main(int argc, char **argv) {
    /* Create your windows and widgets here */

    Fl::lock(); /* "start" the FLTK lock mechanism */

    /* show your window */
    main_win->show(argc, argv);

    /* start your worker threads */
    ... start threads ...

    /* Run the FLTK main loop */
    int result = Fl::run();

    /* terminate any pending worker threads */
    ... stop threads ...

    return result;
}
```

You can start as many threads as you like. From within a thread (other than the `main()` thread) FLTK calls must be wrapped with calls to `Fl::lock()` and `Fl::unlock()`:

```
void my_thread(void) {
    while (thread_still_running) {
        /* do thread work */
        ...
        /* compute new values for widgets */
        ...

        Fl::lock();      // acquire the lock
        my_widget->update(values);
        Fl::unlock();    // release the lock; allow other threads to access FLTK again
        Fl::awake();     // use Fl::awake() to signal main thread to refresh the GUI
    }
}
```

Note

To trigger a refresh of the GUI from a worker thread, the worker code should call `Fl::awake()`

Using Fl::awake thread messages

You can send messages from worker threads to the `main()` thread using `Fl::awake(void* message)`. If using this thread message interface, your `main()` might look like this:

```
int main(int argc, char **argv) {
    /* Create your windows and widgets here */

    Fl::lock(); /* "start" the FLTK lock mechanism */

    /* show your window */
    main_win->show(argc, argv);

    /* start your worker threads */
    ... start threads ...

    /* Run the FLTK loop and process thread messages */
    while (Fl::wait() > 0) {
        if ((next_message = Fl::thread_message()) != NULL) {
            /* process your data, update widgets, etc. */
            ...
        }
    }

    /* terminate any pending worker threads */
    ... stop threads ...

    return 0;
}
```

Your worker threads can send messages to the `main()` thread using `Fl::awake(void* message)`:

```
void *msg;      // "msg" is a pointer to your message
Fl::awake(msg); // send "msg" to main thread
```

A message can be anything you like. The `main()` thread can retrieve the message by calling `Fl::thread_message()`.

Using `Fl::awake` callback messages

You can also request that the `main()` thread call a function on behalf of the worker thread by using `Fl::awake(Fl_Awake_Handler cb, void* userdata)`.

The `main()` thread will execute the callback "as soon as possible" when next processing the pending events. This can be used by a worker thread to perform operations (for example showing or hiding windows) that are prohibited in a worker thread.

```
void do_something_cb(void *userdata) {
    // Will run in the context of the main thread
    ... do_stuff ...
}

// running in worker thread
void *data; // "data" is a pointer to your user data
Fl::awake(do_something_cb, data); // call to execute cb in main thread
```

Note

The `main()` thread will execute the `Fl_Awake_Handler` callback `do_something_cb` asynchronously to the worker thread, at some short but indeterminate time after the worker thread registers the request. When it executes the `Fl_Awake_Handler` callback, the `main()` thread will use the contents of `*userdata` **at the time of execution**, not necessarily the contents that `*userdata` had at the time that the worker thread posted the callback request. The worker thread should therefore contrive **not** to alter the contents of `*userdata` once it posts the callback, since the worker thread does not know when the `main()` thread will consume that data. It is often useful that `userdata` point to a struct, one member of which the `main()` thread can modify to indicate that it has consumed the data, thereby allowing the worker thread to re-use or update `userdata`.

Warning

The mechanisms used to deliver `Fl::awake(void* message)` and `Fl::awake(Fl_Awake_Handler cb, void* userdata)` events to the `main()` thread can interact in unexpected ways on some platforms. Therefore, for reliable operation, it is advised that a program use either `Fl::awake(Fl_Awake_Handler cb, void* userdata)` or `Fl::awake(void* message)`, but that they never be intermixed. Calling `Fl::awake()` with no parameters should be safe in either case.

If you have to choose between using the `Fl::awake(void* message)` and `Fl::awake(Fl_Awake_Handler cb, void* userdata)` mechanisms and don't know which to choose, then try the `Fl::awake(Fl_Awake_Handler cb, void* userdata)` method first as it tends to be more powerful in general.

1.13.4 FLTK multithreaded "lockless programming"

The simple multithreaded examples shown above, using the FLTK lock, work well for many cases where multiple threads are required. However, when that model is extended to more complex programs, it often produces results that the developer did not anticipate.

A typical case might go something like this. A developer creates a program to process a huge data set. The program has a `main()` thread and 7 worker threads and is targeted to run on an 8-core computer. When it runs, the program divides the data between the 7 worker threads, and as they process their share of the data, each thread updates its portion of the GUI with the results, locking and unlocking as they do so.

But when this program runs, it is much slower than expected and the developer finds that only one of the eight CPU cores seems to be utilised, despite there being 8 threads in the program. What happened?

The threads in the program all run as expected, but they end up being serialized (that is, not able to run in parallel) because they all depend on the single FLTK lock. Acquiring (and releasing) that lock has an associated cost, and is a **blocking** action if the lock is already held by any other worker thread or by the `main()` thread.

If the worker threads are acquiring the lock "too often", then the lock will **always** be held **somewhere** and every attempt by any other thread (even `main()`) to lock will cause that other thread (including `main()`) to block. And blocking `main()` also blocks event handling, display refresh...

As a result, only one thread will be running at any given time, and the multithreaded program is effectively reduced to being a (complicated and somewhat less efficient) single thread program.

A "solution" is for the worker threads to lock "less often", such that they do not block each other or the `main()` thread. But judging what constitutes locking "too often" for any given configuration, and hence will block, is a very tricky question. What works well on one machine, with a given graphics card and CPU configuration may behave very differently on another target machine.

There are "interesting" variations on this theme, too: for example it is possible that a "faulty" multithreaded program such as described above will work adequately on a single-core machine (where all threads are inherently serialized anyway and so are less likely to block each other) but then stall or even deadlock in unexpected ways on a multicore machine when the threads do interfere with each other. (I have seen this - it really happens.)

The "better" solution is to avoid using the FLTK lock so far as possible. Instead, the code should be designed so that the worker threads do not update the GUI themselves and therefore never need to acquire the FLTK lock. This would be FLTK multithreaded "lockless programming".

There are a number of ways this can be achieved (or at least approximated) in practice but the most direct approach is for the worker threads to make use of the `Fl::awake(Fl_Awake_Handler cb, void* userdata)` method so that GUI updates can all run in the context of the `main()` thread, alleviating the need for the worker thread to ever lock. The onus is then on the worker threads to manage the `userdata` so that it is delivered safely to the `main()` thread, but there are many ways that can be done.

Note

Using `Fl::awake` is not, strictly speaking, entirely "lockless" since the awake handler mechanism incorporates resource locking internally to protect the queue of pending awake messages. These resource locks are held transiently and generally do not trigger the pathological blocking issues described here.

However, aside from using `Fl::awake`, there are many other ways that a "lockless" design can be implemented, including message passing, various forms of IPC, etc.

If you need high performing multithreaded programming, then take some time to study the options and understand the advantages and disadvantages of each; we can't even begin to scratch the surface of this huge topic here!

And of course occasional, sparse, use of the FLTK lock from worker threads will do no harm; it is "excessive" locking (whatever that might be) that triggers the failing behaviour.

It is always a Good Idea to update the GUI at the lowest rate that is acceptable when processing bulk data (or indeed, in all cases!) Updating at a few frames per second is probably adequate for providing feedback during a long calculation. At the upper limit, anything faster than the frame rate of your monitor and the updates will never even be displayed; why waste CPU computing pixels that you will never show?

1.13.5 FLTK multithreaded Constraints

FLTK supports multiple platforms, some of which allow only the `main()` thread to handle system events and open or close windows. The safe thing to do is to adhere to the following rules for threads on all operating systems:

- Don't `show()` or `hide()` anything that contains `Fl_Window` based widgets from a worker thread. This includes any windows, dialogs, file choosers, subwindows or widgets using `Fl_Gl_Window`. Note that this constraint also applies to non-window widgets that have tooltips, since the tooltip will contain a `Fl_Window` object. The safe and portable approach is **never** to call `show()` or `hide()` on any widget from the context of a worker thread. Instead you can use the `Fl_Awake_Handler` variant of `Fl::awake()` to request the `main()` thread to create, destroy, show or hide the widget on behalf of the worker thread.
- Don't call `Fl::run()`, `Fl::wait()`, `Fl::flush()`, `Fl::check()` or any related methods that will handle system messages from a worker thread
- Don't intermix use of `Fl::awake(Fl_Awake_Handler cb, void* userdata)` and `Fl::awake(void* message)` calls in the same program as they may interact unpredictably on some platforms; choose one or other style of `Fl::awake(<thing>)` mechanism and use that. (Intermixing calls to `Fl::awake()` should be safe with either however.)
- Starting with FLTK 1.4, it's possible to start (or cancel) a timer from a worker thread under the condition that the call to `Fl::add_timeout` (or `Fl::remove_timeout`) is wrapped in `Fl::lock()` and `Fl::unlock()`.
- Don't change window decorations or titles from a worker thread
- The `make_current()` method will probably not work well for regular windows, but should always work for a `Fl_Gl_Window` to allow for high speed rendering on graphics cards with multiple pipelines. Managing thread-safe access to the GL pipelines is left as an exercise for the reader! (And may be target specific...)

See also: `Fl::lock()`, `Fl::unlock()`, `Fl::awake()`, `Fl::awake(Fl_Awake_Handler cb, void* userdata)`, `Fl::awake(void* message)`, `Fl::thread_message()`.

1.14 Unicode and UTF-8 Support

This chapter explains how FLTK handles international text via Unicode and UTF-8.

Unicode support was added to FLTK starting with version 1.3.0 and is still incomplete but mostly functional. This chapter is Work in Progress, reflecting the current state of Unicode support.

1.14.1 About Unicode, ISO 10646 and UTF-8

The summary of Unicode, ISO 10646 and UTF-8 given below is deliberately brief and provides just enough information for the rest of this chapter.

For further information, please see:

- <https://unicode.org>
- <https://iso.org>
- <https://en.wikipedia.org/wiki/Unicode>
- <https://www.cl.cam.ac.uk/~mgk25/unicode.html>
- <https://tools.ietf.org/html/rfc3629>

The Unicode Standard

The Unicode Standard was originally developed by a consortium of mainly US computer manufacturers and developers of multi-lingual software. It has now become a defacto standard for character encoding and is supported by most of the major computing companies in the world.

Before Unicode, many different systems, on different platforms, had been developed for encoding characters for different languages, but no single encoding could satisfy all languages. Unicode provides access to over 130,000 characters used in all the major languages written today, and is independent of platform and language.

Unicode also provides higher-level concepts needed for text processing and typographic publishing systems, such as algorithms for sorting and comparing text, composite character and text rendering, right-to-left and bi-directional text handling.

Note

There are currently no plans to add this extra functionality to FLTK.

ISO 10646

The International Organisation for Standardization (ISO) had also been trying to develop a single unified character set. Although both ISO and the Unicode Consortium continue to publish their own standards, they have agreed to coordinate their work so that specific versions of the Unicode and ISO 10646 standards are compatible with each other.

The international standard ISO 10646 defines the **Universal Character Set** (UCS) which contains the characters required for almost all known languages. The standard also defines three different implementation levels specifying how these characters can be combined.

Note

There are currently no plans for handling the different implementation levels or the combining characters in FLTK.

In UCS, characters have a unique numerical code and an official name, and are usually shown using 'U+' and the code in hexadecimal, e.g. U+0041 is the "Latin capital letter A". The UCS characters U+0000 to U+007F correspond to US-ASCII, and U+0000 to U+00FF correspond to ISO 8859-1 (Latin1).

ISO 10646 was originally designed to handle a 31-bit character set from U+00000000 to U+7FFFFFFF, but the current idea is that 21 bits will be sufficient for all future needs, giving characters up to U+10FFFF. The complete character set is sub-divided into *planes*. *Plane 0*, also known as the **Basic Multilingual Plane** (BMP), ranges from U+0000 to U+FFFF and consists of the most commonly used characters from previous encoding standards. Other planes contain characters for specialist applications.

Todo FLTK 1.3 and later supports the full Unicode range (21 bits), but there are a few exceptions, for instance binary shortcut values in menus ([Fl_Shortcut](#)) can only be used with characters from the BMP (16 bits). This may be extended in a future FLTK version.

The UCS also defines various methods of encoding characters as a sequence of bytes. UCS-2 encodes Unicode characters into two bytes, which is wasteful if you are only dealing with ASCII or Latin1 text, and insufficient if you need characters above U+00FFFF. UCS-4 uses four bytes, which lets it handle higher characters, but this is even more wasteful for ASCII or Latin1.

UTF-8

The Unicode standard defines various UCS Transformation Formats (UTF). UTF-16 and UTF-32 are based on units of two and four bytes. UCS characters requiring more than 16 bits are encoded using "surrogate pairs" in UTF-16.

UTF-8 encodes all Unicode characters into variable length sequences of bytes. Unicode characters in the 7-bit ASCII range map to the same value and are represented as a single byte, making the transformation to Unicode quick and easy.

All UCS characters above U+007F are encoded as a sequence of several bytes. The top bits of the first byte are set to show the length of the byte sequence, and subsequent bytes are always in the range 0x80 to 0xBF. This combination provides some level of synchronisation and error detection.

Unicode range	Byte sequences
U+00000000 - U+0000007F	0xxxxxxx
U+00000080 - U+000007FF	110xxxxx 10xxxxxx
U+00000800 - U+0000FFFF	1110xxxx 10xxxxxx 10xxxxxx
U+00010000 - U+001FFFFF	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
U+00200000 - U+03FFFFFF	111110xx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx
U+04000000 - U+7FFFFFFF	1111110x 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx

Note

This table contains theoretical values outside the valid Unicode range (U+000000 - U+10FFFF). Such values can only be returned by conversion functions for illegal input values (see [Illegal Unicode and UTF-8 Sequences](#)).

Moving from ASCII encoding to Unicode will allow all new FLTK applications to be easily internationalized and used all over the world. By choosing UTF-8 encoding, FLTK remains largely source-code compatible to previous iterations of the library.

1.14.2 Unicode in FLTK

Todo Work through the code and this documentation to harmonize the `[OksiD]` and `[fltk2]` functions.

FLTK will be entirely converted to Unicode using UTF-8 encoding. If a different encoding is required by the underlying operating system, FLTK will convert the string as needed.

It is important to note that the initial implementation of Unicode and UTF-8 in FLTK involves three important areas:

- provision of Unicode character tables and some simple related functions;
- conversion of `char*` variables and function parameters from single byte per character representation to UTF-8 variable length sequences;
- modifications to the display font interface to accept general Unicode character or UCS code numbers instead of just ASCII or Latin1 characters.

The current implementation of Unicode / UTF-8 in FLTK will impose the following limitations:

- An implementation note in the `[OksiD]` code says that all functions are LIMITED to 24 bit Unicode values, but also says that only 16 bits are really used under linux and win32. **[Can we verify this?]**
- The `[fltk2]` `fl_utf8encode()` and `fl_utf8decode()` functions are designed to handle Unicode characters in the range U+000000 to U+10FFFF inclusive, which covers all UTF-16 characters, as specified in RFC 3629. *Note that the user must first convert UTF-16 surrogate pairs to UCS.*
- FLTK will only handle single characters, so composed characters consisting of a base character and floating accent characters will be treated as multiple characters.
- FLTK will only compare or sort strings on a byte by byte basis and not on a general Unicode character basis.
- FLTK will not handle right-to-left or bi-directional text.

Todo Verify 16/24 bit Unicode limit for different character sets? OksiD's code appears limited to 16-bit whereas the FLTK2 code appears to handle a wider set. What about illegal characters? See comments in `fl_utf8fromwc()` and `fl_utf8toUtf16()`.

1.14.3 Illegal Unicode and UTF-8 Sequences

Three pre-processor variables are defined in the source code [1] that determine how `fl_utf8decode()` handles illegal UTF-8 sequences:

- if `ERRORS_TO_CP1252` is set to 1 (the default), `fl_utf8decode()` will assume that a byte sequence starting with a byte in the range 0x80 to 0x9f represents a Microsoft CP1252 character, and will return the value of an equivalent UCS character. Otherwise, it will be processed as an illegal byte value as described below.
- if `STRICT_RFC3629` is set to 1 (not the default!) then UTF-8 sequences that correspond to illegal UCS values are treated as errors. Illegal UCS values include those above U+10FFFF, or corresponding to UTF-16 surrogate pairs. Illegal byte values are handled as described below.
- if `ERRORS_TO_ISO8859_1` is set to 1 (the default), the illegal byte value is returned unchanged, otherwise 0xFFFD, the Unicode REPLACEMENT CHARACTER, is returned instead.

[1] Since FLTK 1.3.4 you may set these three pre-processor variables on your compile command line with -D"variable=value" (value: 0 or 1) to avoid editing the source code.

fl_utf8encode() is less strict, and only generates the UTF-8 sequence for 0xFFFD, the Unicode REPLACEMENT CHARACTER, if it is asked to encode a UCS value above U+10FFFF.

Many of the [fltk2] functions below use fl_utf8decode() and fl_utf8encode() in their own implementation, and are therefore somewhat protected from bad UTF-8 sequences.

The [OksiD] fl_utf8len() function assumes that the byte it is passed is the first byte in a UTF-8 sequence, and returns the length of the sequence. Trailing bytes in a UTF-8 sequence will return -1.

- **WARNING:** fl_utf8len() can not distinguish between single bytes representing Microsoft CP1252 characters 0x80-0x9f and those forming part of a valid UTF-8 sequence. You are strongly advised not to use fl_utf8len() in your own code unless you know that the byte sequence contains only valid UTF-8 sequences.
- **WARNING:** Some of the [OksiD] functions below still use fl_utf8len() in their implementations. These may need further validation.

Please see the individual function description for further details about error handling and return values.

1.14.4 FLTK Unicode and UTF-8 Functions

This section provides a brief overview of the functions. For more details, consult the main text for each function via its link.

int [fl_utf8locale\(\)](#) **FLTK2**

[fl_utf8locale\(\)](#) returns true if the "locale" seems to indicate that UTF-8 encoding is used.

It is highly recommended that you change your system so this does return true!

int [fl_utf8test\(const char *src, unsigned len\)](#) **FLTK2**

[fl_utf8test\(\)](#) examines the first len bytes of src. It returns 0 if there are any illegal UTF-8 sequences; 1 if src contains plain ASCII or if len is zero; or 2, 3 or 4 to indicate the range of Unicode characters found.

int [fl_utf_nb_char\(const unsigned char *buf, int len\)](#) **OksiD**

Returns the number of UTF-8 characters in the first len bytes of buf.

int [fl_unichar_to_utf8_size\(Fl_Unichar\)](#)

int [fl_utf8bytes\(unsigned ucs\)](#)

Returns the number of bytes needed to encode `ucs` in UTF-8.

`int fl_utf8len(char c)` **OksID**

If `c` is a valid first byte of a UTF-8 encoded character sequence, `fl_utf8len()` will return the number of bytes in that sequence. It returns -1 if `c` is not a valid first byte.

`unsigned int fl_nonspacing(unsigned int ucs)` **OksID**

Returns true if `ucs` is a non-spacing character.

`const char* fl_utf8back(const char *p, const char *start, const char *end)` **FLTK2**

`const char* fl_utf8fwd(const char *p, const char *start, const char *end)` **FLTK2**

If `p` already points to the start of a UTF-8 character sequence, these functions will return `p`. Otherwise `fl_utf8back()` searches backwards from `p` and `fl_utf8fwd()` searches forwards from `p`, within the `start` and `end` limits, looking for the start of a UTF-8 character.

`unsigned int fl_utf8decode(const char *p, const char *end, int *len)` **FLTK2**

`int fl_utf8encode(unsigned ucs, char *buf)` **FLTK2**

`fl_utf8decode()` attempts to decode the UTF-8 character that starts at `p` and may not extend past `end`. It returns the Unicode value, and the length of the UTF-8 character sequence is returned via the `len` argument. `fl_utf8encode()` writes the UTF-8 encoding of `ucs` into `buf` and returns the number of bytes in the sequence. See the main documentation for the treatment of illegal Unicode and UTF-8 sequences.

`unsigned int fl_utf8froma(char *dst, unsigned dstlen, const char *src, unsigned srclen)` **FLTK2**

`unsigned int fl_utf8toa(const char *src, unsigned srclen, char *dst, unsigned dstlen)` **FLTK2**

`fl_utf8froma()` converts a character string containing single bytes per character (i.e. ASCII or ISO-8859-1) into UTF-8. If the `src` string contains only ASCII characters, the return value will be the same as `srclen`.

`fl_utf8toa()` converts a string containing UTF-8 characters into single byte characters. UTF-8 characters that do not correspond to ASCII or ISO-8859-1 characters below 0xFF are replaced with '?'.

Both functions return the number of bytes that would be written, not counting the null terminator. `dstlen` provides a means of limiting the number of bytes written, so setting `dstlen` to zero is a means of measuring how much storage would be needed before doing the real conversion.

`char* fl_utf2mbcs(const char *src)` **Oksid**

converts a UTF-8 string to a local multi-byte character string. **[More info required here!]**

`unsigned int fl_utf8fromwc(char *dst, unsigned dstlen, const wchar_t *src, unsigned srclen)` **FLTK2**

`unsigned int fl_utf8towc(const char *src, unsigned srclen, wchar_t *dst, unsigned dstlen)` **FLTK2**

`unsigned int fl_utf8toUtf16(const char *src, unsigned srclen, unsigned short *dst, unsigned dstlen)` **FLTK2**

These routines convert between UTF-8 and `wchar_t` or "wide character" strings. The difficulty lies in the fact that `sizeof(wchar_t)` is 2 on Windows and 4 on Linux and most other systems. Therefore some "wide characters" on Windows may be represented as "surrogate pairs" of more than one `wchar_t`.

`fl_utf8fromwc()` converts from a "wide character" string to UTF-8. Note that `srclen` is the number of `wchar_t` elements in the source string and on Windows this might be larger than the number of characters. `dstlen` specifies the maximum number of **bytes** to copy, including the null terminator.

`fl_utf8towc()` converts a UTF-8 string into a "wide character" string. Note that on Windows, some "wide characters" might result in "surrogate pairs" and therefore the return value might be more than the number of characters. `dstlen` specifies the maximum number of `wchar_t` elements to copy, including a zero terminating element. **[Is this all worded correctly?]**

`fl_utf8toUtf16()` converts a UTF-8 string into a "wide character" string using UTF-16 encoding to handle the "surrogate pairs" on Windows. `dstlen` specifies the maximum number of `wchar_t` elements to copy, including a zero terminating element. **[Is this all worded correctly?]**

These routines all return the number of elements that would be required for a full conversion of the `src` string, including the zero terminator. Therefore setting `dstlen` to zero is a way of measuring how much storage would be needed before doing the real conversion.

`unsigned int fl_utf8from_mb(char *dst, unsigned dstlen, const char *src, unsigned srclen)` **FLTK2**

`unsigned int fl_utf8to_mb(const char *src, unsigned srclen, char *dst, unsigned dstlen)` **FLTK2**

These functions convert between UTF-8 and the locale-specific multi-byte encodings used on some systems for filenames, etc. If `fl_utf8locale()` returns true, these functions don't do anything useful. **[Is this all worded correctly?]**

```
int fl_tolower(unsigned int ucs) OksiD
int fl_toupper(unsigned int ucs) OksiD
int fl_utf_tolower(const unsigned char *str, int len, char *buf) OksiD
int fl_utf_toupper(const unsigned char *str, int len, char *buf) OksiD
```

`fl_tolower()` and `fl_toupper()` convert a single Unicode character from upper to lower case, and vice versa. `fl_utf_tolower()` and `fl_utf_toupper()` convert a string of bytes, some of which may be multi-byte UTF-8 encodings of Unicode characters, from upper to lower case, and vice versa.

Warning: to be safe, `buf` length must be at least `3*len` [for 16-bit Unicode]

```
int fl_utf_strcasecmp(const char *s1, const char *s2) OksiD
int fl_utf_strncasecmp(const char *s1, const char *s2, int n) OksiD
```

`fl_utf_strcasecmp()` is a UTF-8 aware string comparison function that converts the strings to lower case Unicode as part of the comparison. `fl_utf_strncasecmp()` only compares the first `n` characters [bytes?]

1.14.5 FLTK Unicode Versions of System Calls

- `int fl_access(const char* f, int mode) OksiD`
- `int fl_chmod(const char* f, int mode) OksiD`
- `int fl_execvp(const char* file, char* const* argv) OksiD`
- `FILE* fl_fopen(const char* f, const char* mode) OksiD`
- `char* fl_getcwd(char* buf, int maxlen) OksiD`
- `char* fl_getenv(const char* name) OksiD`
- `char fl_make_path(const char* path) - returns char ? OksiD`
- `void fl_make_path_for_file(const char* path) OksiD`
- `int fl_mkdir(const char* f, int mode) OksiD`
- `int fl_open(const char* f, int o, ...) OksiD`
- `int fl_rename(const char* f, const char* t) OksiD`
- `int fl_rmdir(const char* f) OksiD`
- `int fl_stat(const char* path, struct stat* buffer) OksiD`
- `int fl_system(const char* f) OksiD`
- `int fl_unlink(const char* f) OksiD`

TODO:

- more doc on unicode, add links
- write something about filename encoding on OS X...
- explain the `fl_utf8_...` commands
- explain issues with [FL_Preferences](#)

1.15 Constants and Enumerations

Note

This file is not actively maintained any more, but is left here as a reference, until the doxygen documentation is completed.

See also

[FL/Enumerations.H](#).

This appendix lists the enumerations provided in the `<FL/Enumerations.H>` header file, organized by section. Constants whose value are zero are marked with "(0)", this is often useful to know when programming.

1.15.1 Version Numbers

The FLTK version number is stored in a number of compile-time constants:

- `FL_MAJOR_VERSION` - The major release number, currently 1
- `FL_MINOR_VERSION` - The minor release number, currently 4
- `FL_PATCH_VERSION` - The patch release number, currently 1
- `FL_VERSION` - [Deprecated] A combined floating-point version number for the major, minor, and patch release numbers, currently 1.0400
- `FL_API_VERSION` - A combined integer version number for the major, minor, and patch release numbers, currently 10400 (use this instead of `FL_VERSION`, if possible)
- `FL_ABI_VERSION` - A combined integer version number for the application binary interface (ABI) major, minor, and patch release numbers, currently 10400 (default)

Note

The ABI version (`FL_ABI_VERSION`) is usually constant throughout one major/minor release version, for instance 10300 if `FL_API_VERSION` is 10304. Hence the ABI is constant if only the patch version is changed. You can change this with `configure` or `CMake` though if you want the latest enhancements (called "ABI features", see `CHANGES`).

1.15.2 Events

Events are identified by an [FL_Event](#) enumeration value. The following events are currently defined:

- FL_NO_EVENT - No event (or an event fltk does not understand) occurred (0).
- FL_PUSH - A mouse button was pushed.
- FL_RELEASE - A mouse button was released.
- FL_ENTER - The mouse pointer entered a widget.
- FL_LEAVE - The mouse pointer left a widget.
- FL_DRAG - The mouse pointer was moved with a button pressed.
- FL_FOCUS - A widget should receive keyboard focus.
- FL_UNFOCUS - A widget loses keyboard focus.
- FL_KEYBOARD - A key was pressed.
- FL_CLOSE - A window was closed.
- FL_MOVE - The mouse pointer was moved with no buttons pressed.
- FL_SHORTCUT - The user pressed a shortcut key.
- FL_DEACTIVATE - The widget has been deactivated.
- FL_ACTIVATE - The widget has been activated.
- FL_HIDE - The widget has been hidden.
- FL_SHOW - The widget has been shown.
- FL_PASTE - The widget should paste the contents of the clipboard.
- FL_SELECTIONCLEAR - The widget should clear any selections made for the clipboard.
- FL_MOUSEWHEEL - The horizontal or vertical mousewheel was turned.
- FL_DND_ENTER - The mouse pointer entered a widget dragging data.
- FL_DND_DRAG - The mouse pointer was moved dragging data.
- FL_DND_LEAVE - The mouse pointer left a widget still dragging data.
- FL_DND_RELEASE - Dragged data is about to be dropped.
- FL_SCREEN_CONFIGURATION_CHANGED - The screen configuration (number, positions) was changed.
- FL_FULLSCREEN - The fullscreen state of the window has changed.

1.15.3 Callback "When" Conditions

The following constants determine when a callback is performed:

- FL_WHEN_NEVER - Never call the callback (0).
- FL_WHEN_CHANGED - Do the callback only when the widget value changes.
- FL_WHEN_NOT_CHANGED - Do the callback whenever the user interacts with the widget.
- FL_WHEN_RELEASE - Do the callback when the button or key is released and the value changes.
- FL_WHEN_ENTER_KEY - Do the callback when the user presses the ENTER key and the value changes.
- FL_WHEN_RELEASE_ALWAYS - Do the callback when the button or key is released, even if the value doesn't change.
- FL_WHEN_ENTER_KEY_ALWAYS - Do the callback when the user presses the ENTER key, even if the value doesn't change.

1.15.4 Fl::event_button() Values

The following constants define the button numbers for FL_PUSH and FL_RELEASE events:

- FL_LEFT_MOUSE - the left mouse button
- FL_MIDDLE_MOUSE - the middle mouse button
- FL_RIGHT_MOUSE - the right mouse button
- FL_BACK_MOUSE - the back mouse button (side button 1)
- FL_FORWARD_MOUSE - the forward mouse button (side button 2)

1.15.5 Fl::event_key() Values

The following constants define the non-ASCII keys on the keyboard for FL_KEYBOARD and FL_SHORTCUT events:

- FL_Button - A mouse button; use `Fl_Button + n` for mouse button `n`.
- FL_BackSpace - The backspace key.
- FL_Tab - The tab key.
- FL_Enter - The enter key.
- FL_Pause - The pause key.
- FL_Scroll_Lock - The scroll lock key.
- FL_Escape - The escape key.
- FL_Home - The home key.
- FL_Left - The left arrow key.
- FL_Up - The up arrow key.
- FL_Right - The right arrow key.
- FL_Down - The down arrow key.
- FL_Page_Up - The page-up key.
- FL_Page_Down - The page-down key.
- FL_End - The end key.
- FL_Print - The print (or print-screen) key.
- FL_Insert - The insert key.
- FL_Menu - The menu key.
- FL_Num_Lock - The num lock key.
- FL_KP - One of the keypad numbers or keys; use `FL_KP + 'n'` for number `n` and, say, `FL_KP + '*'`.
- FL_KP_Enter - The enter key on the keypad.
- FL_F - One of the function keys; use `FL_F + n` for function key `n`.
- FL_Shift_L - The lefthand shift key.

- `FL_Shift_R` - The righthand shift key.
- `FL_Control_L` - The lefthand control key.
- `FL_Control_R` - The righthand control key.
- `FL_Caps_Lock` - The caps lock key.
- `FL_Meta_L` - The left meta/Windows key.
- `FL_Meta_R` - The right meta/Windows key.
- `FL_Alt_L` - The left alt key.
- `FL_Alt_R` - The right alt key.
- `FL_Delete` - The delete key.
- `FL_Alt_Gr` - The AltGr key on some international keyboards.

1.15.6 `Fl::event_state()` Values

The following constants define bits in the `Fl::event_state()` value:

- `FL_SHIFT` - One of the shift keys is down.
- `FL_CAPS_LOCK` - The caps lock is on.
- `FL_CTRL` - One of the ctrl keys is down.
- `FL_ALT` - One of the alt keys is down.
- `FL_NUM_LOCK` - The num lock is on.
- `FL_META` - One of the meta/Windows keys is down.
- `FL_COMMAND` - An alias for `FL_CTRL` on Windows, X11 and Wayland, or `FL_META` on MacOS X.
- `FL_CONTROL` - An alias for `FL_META` on Windows, X11 and Wayland, or `FL_CTRL` on MacOS X.
- `FL_SCROLL_LOCK` - The scroll lock is on.
- `FL_BUTTON1` - Mouse button 1 is pushed.
- `FL_BUTTON2` - Mouse button 2 is pushed.
- `FL_BUTTON3` - Mouse button 3 is pushed.
- `FL_BUTTON4` - Mouse button 4 (back) is pushed.
- `FL_BUTTON5` - Mouse button 5 (forward) is pushed.
- `FL_BUTTONS` - Any mouse button (1..5) is pushed.
- `FL_BUTTON(n)` - Mouse button `n` (where `n > 0`) is pushed.

1.15.7 Alignment Values

The following constants define bits that can be used with `FL_Widget::align()` to control the positioning of the label:

- `FL_ALIGN_CENTER` - The label is centered (0).
- `FL_ALIGN_TOP` - The label is top-aligned.
- `FL_ALIGN_BOTTOM` - The label is bottom-aligned.
- `FL_ALIGN_LEFT` - The label is left-aligned.
- `FL_ALIGN_RIGHT` - The label is right-aligned.
- `FL_ALIGN_CLIP` - The label is clipped to the widget.
- `FL_ALIGN_WRAP` - The label text is wrapped as needed.
- `FL_ALIGN_TOP_LEFT` - The label appears at the top of the widget, aligned to the left.
- `FL_ALIGN_TOP_RIGHT` - The label appears at the top of the widget, aligned to the right.
- `FL_ALIGN_BOTTOM_LEFT` - The label appears at the bottom of the widget, aligned to the left.
- `FL_ALIGN_BOTTOM_RIGHT` - The label appears at the bottom of the widget, aligned to the right.
- `FL_ALIGN_LEFT_TOP` - The label appears to the left of the widget, aligned at the top. Outside labels only.
- `FL_ALIGN_RIGHT_TOP` - The label appears to the right of the widget, aligned at the top. Outside labels only.
- `FL_ALIGN_LEFT_BOTTOM` - The label appears to the left of the widget, aligned at the bottom. Outside labels only.
- `FL_ALIGN_RIGHT_BOTTOM` - The label appears to the right of the widget, aligned at the bottom. Outside labels only.
- `FL_ALIGN_INSIDE` - 'or' this with other values to put label inside the widget.
- `FL_ALIGN_TEXT_OVER_IMAGE` - Label text will appear above the image.
- `FL_ALIGN_IMAGE_OVER_TEXT` - Label text will be below the image.
- `FL_ALIGN_IMAGE_NEXT_TO_TEXT` - The image will appear to the left of the text.
- `FL_ALIGN_TEXT_NEXT_TO_IMAGE` - The image will appear to the right of the text.
- `FL_ALIGN_IMAGE_BACKDROP` - The image will be used as a background for the widget.

1.15.8 Fonts

The following constants define the standard FLTK fonts:

- `FL_HELVETICA` - Helvetica (or Arial) normal (0).
- `FL_HELVETICA_BOLD` - Helvetica (or Arial) bold.
- `FL_HELVETICA_ITALIC` - Helvetica (or Arial) oblique.
- `FL_HELVETICA_BOLD_ITALIC` - Helvetica (or Arial) bold-oblique.
- `FL_COURIER` - Courier normal.
- `FL_COURIER_BOLD` - Courier bold.

- FL_COURIER_ITALIC - Courier italic.
- FL_COURIER_BOLD_ITALIC - Courier bold-italic.
- FL_TIMES - Times roman.
- FL_TIMES_BOLD - Times bold.
- FL_TIMES_ITALIC - Times italic.
- FL_TIMES_BOLD_ITALIC - Times bold-italic.
- FL_SYMBOL - Standard symbol font.
- FL_SCREEN - Default monospaced screen font.
- FL_SCREEN_BOLD - Default monospaced bold screen font.
- FL_ZAPF_DINGBATS - Zapf-dingbats font.

1.15.9 Colors

The FL_Color enumeration type holds a FLTK color value. Colors are either 8-bit indexes into a [virtual colormap](#) or 24-bit RGB color values. Color indices occupy the lower 8 bits of the value, while RGB colors occupy the upper 24 bits, for a byte organization of RGBI.

1.15.9.1 Color Constants

Constants are defined for the user-defined foreground and background colors, as well as specific colors and the start of the grayscale ramp and color cube in the [virtual colormap](#). Inline functions are provided to retrieve specific grayscale, color cube, or RGB color values.

The following color constants can be used to access the user-defined colors:

- FL_BACKGROUND_COLOR - the default background color
- FL_BACKGROUND2_COLOR - the default background color for text, list, and valuator widgets
- FL_FOREGROUND_COLOR - the default foreground color (0) used for labels and text
- FL_INACTIVE_COLOR - the inactive foreground color
- FL_SELECTION_COLOR - the default selection/highlight color

The following color constants can be used to access the colors from the FLTK standard color cube:

- FL_BLACK
- FL_BLUE
- FL_CYAN
- FL_DARK_BLUE
- FL_DARK_CYAN
- FL_DARK_GREEN
- FL_DARK_MAGENTA

- FL_DARK_RED
- FL_DARK_YELLOW
- FL_GREEN
- FL_MAGENTA
- FL_RED
- FL_WHITE
- FL_YELLOW

The following are named values within the standard grayscale:

- FL_GRAY0
- FL_DARK3
- FL_DARK2
- FL_DARK1
- FL_LIGHT1
- FL_LIGHT2
- FL_LIGHT3

The inline methods for getting a grayscale, color cube, or RGB color value are described in the [Colors](#) section of the [Drawing Things in FLTK](#) chapter.

1.15.10 Cursors

The following constants define the mouse cursors that are available in FLTK. The double-headed arrows are bitmaps provided by FLTK on X, the others are provided by system-defined cursors.

- FL_CURSOR_DEFAULT - the default cursor, usually an arrow (0)
- FL_CURSOR_ARROW - an arrow pointer
- FL_CURSOR_CROSS - crosshair
- FL_CURSOR_WAIT - watch or hourglass
- FL_CURSOR_INSERT - I-beam
- FL_CURSOR_HAND - hand (uparrow on Windows)
- FL_CURSOR_HELP - question mark
- FL_CURSOR_MOVE - 4-pointed arrow
- FL_CURSOR_NS - up/down arrow
- FL_CURSOR_WE - left/right arrow
- FL_CURSOR_NWSE - diagonal arrow
- FL_CURSOR_NESW - diagonal arrow
- FL_CURSOR_NONE - invisible

1.15.11 FD "When" Conditions

- `FL_READ` - Call the callback when there is data to be read.
- `FL_WRITE` - Call the callback when data can be written without blocking.
- `FL_EXCEPT` - Call the callback if an exception occurs on the file.

1.15.12 Damage Masks

The following damage mask bits are used by the standard FLTK widgets:

- `FL_DAMAGE_CHILD` - A child needs to be redrawn.
- `FL_DAMAGE_EXPOSE` - The window was exposed.
- `FL_DAMAGE_SCROLL` - The [Fl_Scroll](#) widget was scrolled.
- `FL_DAMAGE_OVERLAY` - The overlay planes need to be redrawn.
- `FL_DAMAGE_USER1` - First user-defined damage bit.
- `FL_DAMAGE_USER2` - Second user-defined damage bit.
- `FL_DAMAGE_ALL` - Everything needs to be redrawn.

1.16 GLUT Compatibility

This appendix describes the GLUT compatibility header file supplied with FLTK. FLTK's GLUT compatibility is based on the original GLUT 3.7 and the follow-on FreeGLUT 2.4.0 libraries.

1.16.1 Using the GLUT Compatibility Header File

You should be able to compile existing GLUT source code by including `<FL/glut.H>` instead of `<GL/glut.h>`. This can be done by editing the source, by changing the `-I` switches to the compiler, or by providing a symbolic link from `GL/glut.h` to `FL/glut.H`.

All files calling GLUT procedures must be compiled with C++. You may have to alter them slightly to get them to compile without warnings, and you may have to rename them to get make to use the C++ compiler.

You must link with the FLTK library. Most of `FL/glut.H` is inline functions. You should take a look at it (and maybe at `test/glpuzzle.cxx` in the FLTK source) if you are having trouble porting your GLUT program.

This has been tested with most of the demo programs that come with the GLUT and FreeGLUT distributions.

1.16.2 Known Problems

The following functions and/or arguments to functions are missing, and you will have to replace them or comment them out for your code to compile:

- `glutGet (GLUT_ELAPSED_TIME)`
- `glutGet (GLUT_SCREEN_HEIGHT_MM)`
- `glutGet (GLUT_SCREEN_WIDTH_MM)`
- `glutGet (GLUT_WINDOW_NUM_CHILDREN)`
- `glutInitDisplayMode (GLUT_LUMINANCE)`
- `glutKeyboardUpFunc(void(*callback)(unsigned char key, int x, int y))`
- `glutLayerGet (GLUT_HAS_OVERLAY)`
- `glutLayerGet (GLUT_LAYER_IN_USE)`
- `glutPushWindow ()`
- `glutSetColor (), glutGetColor (), glutCopyColormap ()`
- `glutVideoResize ()` missing.
- `glutWarpPointer ()`
- `glutWindowStatusFunc ()`
- Spaceball, buttonbox, dials, and tablet functions

Most of the symbols/enumerations have different values than GLUT uses. This will break code that relies on the actual values. The only symbols guaranteed to have the same values are true/false pairs like `GLUT_DOWN` and `GLUT_UP`, mouse buttons `GLUT_LEFT_BUTTON`, `GLUT_MIDDLE_BUTTON`, `GLUT_RIGHT_BUTTON`, and `GLUT_KEY_F1` thru `GLUT_KEY_F12`.

The strings passed as menu labels are not copied.

`glutPostRedisplay ()` does not work if called from inside a display function. You must use `glutIdleFunc ()` if you want your display to update continuously.

`glutSwapBuffers ()` does not work from inside a display function. This is on purpose, because FLTK swaps the buffers for you.

`glutUseLayer ()` does not work well, and should only be used to initialize transformations inside a resize callback. You should redraw overlays by using `glutOverlayDisplayFunc ()`.

Overlays are cleared before the overlay display function is called. `glutLayerGet (GLUT_OVERLAY_DAMAGED)` always returns true for compatibility with some GLUT overlay programs. You must rewrite your code so that `gl_color ()` is used to choose colors in an overlay, or you will get random overlay colors.

`glutSetCursor (GLUT_CURSOR_FULL_CROSSHAIR)` just results in a small crosshair.

The fonts used by `glutBitmapCharacter ()` and `glutBitmapWidth ()` may be different.

`glutInit (argc, argv)` will consume different switches than GLUT does. It accepts the switches recognized by `Fl::args()`, and will accept any abbreviation of these switches (such as `"-di"` for `"-display"`).

1.16.3 Mixing GLUT and FLTK Code

You can make your GLUT window a child of a `Fl_Window` with the following scheme. The biggest trick is that GLUT insists on a call to `show()` the window at the point it is created, which means the `Fl_Window` parent window must already be shown.

- Don't call `glutInit()`.
- Create your `Fl_Window`, and any FLTK widgets. Leave a blank area in the window for your GLUT window.
- `show()` the `Fl_Window`. Perhaps call `show(argc, argv)`.
- Call `window->begin()` so that the GLUT window will be automatically added to it.
- Use `glutInitWindowSize()` and `glutInitWindowPosition()` to set the location in the parent window to put the GLUT window.
- Put your GLUT code next. It probably does not need many changes. Call `window->end()` immediately after the `glutCreateWindow()` !
- You can call either `glutMainLoop()`, `Fl::run()`, or loop calling `Fl::wait()` to run the program.

1.16.4 class `Fl_Glut_Window`

1.16.4.1 Class Hierarchy

```
Fl_Gl_Window
|
+----Fl_Glut_Window
```

1.16.4.2 Include Files

```
#include <FL/glut.H>
```

1.16.4.3 Description

Each GLUT window is an instance of this class. You may find it useful to manipulate instances directly rather than use GLUT window id's. These may be created without opening the display, and thus can fit better into FLTK's method of creating windows.

The current GLUT window is available in the global variable `glut_window`.

`new Fl_Glut_Window(...)` is the same as `glutCreateWindow()` except it does not `show()` the window or make the window current.

`window->make_current()` is the same as `glutSetWindow(number)`. If the window has not had `show()` called on it yet, some functions that assume an OpenGL context will not work. If you do `show()` the window, call `make_current()` again to set the context.

`~Fl_Glut_Window()` is the same as `glutDestroyWindow()`.

1.16.4.4 Members

The `Fl_Glut_Window` class contains several public members that can be altered directly:

member	description
display	A pointer to the function to call to draw the normal planes.
entry	A pointer to the function to call when the mouse moves into or out of the window.
keyboard	A pointer to the function to call when a regular key is pressed.
menu[3]	The menu to post when one of the mouse buttons is pressed.
mouse	A pointer to the function to call when a button is pressed or released.
motion	A pointer to the function to call when the mouse is moved with a button down.
overlaydisplay	A pointer to the function to call to draw the overlay planes.
passivemotion	A pointer to the function to call when the mouse is moved with no buttons down.
reshape	A pointer to the function to call when the window is resized.
special	A pointer to the function to call when a special key is pressed.
visibility	A pointer to the function to call when the window is iconified or restored (made visible.)

1.16.4.5 Methods

`Fl_Glut_Window::Fl_Glut_Window(int x, int y, int w, int h, const char *title = 0)`

`Fl_Glut_Window::Fl_Glut_Window(int w, int h, const char *title = 0)`

The first constructor takes 4 int arguments to create the window with a preset position and size. The second constructor with 2 arguments will create the window with a preset size, but the window manager will choose the position according to its own whims.

`virtual Fl_Glut_Window::~~Fl_Glut_Window()`

Destroys the GLUT window.

`void Fl_Glut_Window::make_current()`

Switches all drawing functions to the GLUT window.

1.17 Forms Compatibility

This appendix describes the Forms compatibility included with FLTK.

Note

The Forms compatibility library is deprecated, no longer actively maintained since FLTK 1.3.0, and likely to be removed completely in FLTK 1.5.

Since FLTK 1.4 building the Forms compatibility library `fltk_forms` (configure/Makefiles) or `fltk::forms` (CMake) can be disabled with one of these commands:

```
- ./configure --disable-forms ...
- cmake -D FLTK_BUILD_FORMS:BOOL=OFF ...
- cmake-gui ...
```

Fluid can still import Forms and XForms designer (.fd) files but w/o any guarantees for working results. Manual fixes may be necessary.

In the next minor or major release (1.5 or higher) the Forms compatibility library will not be built by default or will be removed entirely.

1.17.1 Importing Forms Layout Files

FLUID can read the `.fd` files put out by all versions of Forms and XForms `fdesign`. However, it will mangle them a bit, but it prints a warning message about anything it does not understand. FLUID cannot write `fdesign` files, so you should save to a new name so you don't write over the old one.

You will need to edit your main code considerably to get it to link with the output from FLUID. If you are not interested in this you may have more immediate luck with the forms compatibility header, `<FL/forms.H>`.

1.17.2 Using the Compatibility Header File

You should be able to compile existing Forms or XForms source code by changing the include directory switch to your compiler so that the `forms.h` file supplied with FLTK is included. The `forms.h` file simply pulls in `<FL/forms.H>` so you don't need to change your source code. Take a look at `<FL/forms.H>` to see how it works, but the basic trick is lots of inline functions. Most of the XForms demo programs work without changes.

You will also have to compile your Forms or XForms program using a C++ compiler. The FLTK library does not provide C bindings or header files.

Although FLTK was designed to be compatible with the GL Forms library (version 0.3 or so), XForms has bloated severely and its interface is X-specific. Therefore, XForms compatibility is no longer a goal of FLTK. Compatibility was limited to things that were free, or that would add code that would not be linked in if the feature is unused, or that was not X-specific.

To use any new features of FLTK, you should rewrite your code to not use the inline functions and instead use "pure" FLTK. This will make it a lot cleaner and make it easier to figure out how to call the FLTK functions. Unfortunately this conversion is harder than expected and even Digital Domain's inhouse code still uses `forms.H` a lot.

1.17.3 Problems You Will Encounter

Many parts of XForms use X-specific structures like `XEvent` in their interface. I did not emulate these! Unfortunately these features (such as the "canvas" widget) are needed by most large programs. You will need to rewrite these to use FLTK subclasses.

`Fl_Free` widgets emulate the *old* Forms "free" widget. It may be useful for porting programs that change the `handle()` function on widgets, but you will still need to rewrite things.

`Fl_Timer` widgets are provided to emulate the XForms timer. These work, but are quite inefficient and inaccurate compared to using `Fl::add_timeout()`.

All instance variables are hidden. If you directly refer to the `x`, `y`, `w`, `h`, `label`, or other fields of your Forms widgets you will have to add empty parenthesis after each reference. The easiest way to do this is to globally replace `"->x"` with `"->x()"`, etc. Replace `"boxtype"` with `"box()"`.

`const char *` arguments to most FLTK methods are simply stored, while Forms would `strdup()` the passed string. This is most noticeable with the label of widgets. Your program must always pass static data such as a string constant or malloc'd buffer to `label()`. If you are using labels to display program output you may want to try the `Fl_Output` widget.

The default fonts and sizes are matched to the older GL version of Forms, so all labels will draw somewhat larger than an XForms program does.

`fdesign` outputs a setting of a `"fdui"` instance variable to the main window. I did not emulate this because I wanted all instance variables to be hidden. You can store the same information in the `user_data()` field of a window. To do this, search through the `fdesign` output for all occurrences of `"->fdui"` and edit to use `"->user_data()"` instead. This will require casts and is not trivial.

The prototype for the functions passed to `fl_add_timeout()` and `fl_set_idle_callback()` callback are different.

All the following XForms calls are missing:

- `FL_REVISION, fl_library_version()`
- `FL_RETURN_DBLCLICK` (use [Fl::event_clicks\(\)](#))
- `fl_add_signal_callback()`
- `fl_set_form_atactivate()` `fl_set_form_atdeactivate()`
- `fl_set_form_property()`
- `fl_set_app_mainform()`, `fl_get_app_mainform()`
- `fl_set_form_minsize()`, `fl_set_form_maxsize()`
- `fl_set_form_event_cmask()`, `fl_get_form_event_cmask()`
- `fl_set_form_dblbuffer()`, `fl_set_object_dblbuffer()` (use an [Fl_Double_Window](#) instead)
- `fl_adjust_form_size()`
- `fl_register_raw_callback()`
- `fl_set_object_bw()`, `fl_set_border_width()`
- `fl_set_object_resize()`, `fl_set_object_gravity()`
- `fl_set_object_shortcutkey()`
- `fl_set_object_automatic()`
- `fl_get_object_bbox()` (maybe FLTK should do this)
- `fl_set_object_prehandler()`, `fl_set_object_posthandler()`
- `fl_enumerate_fonts()`
- **Most drawing functions**
- `fl_set_coordunit()` (FLTK uses pixels all the time)
- `fl_ringbell()`
- `fl_gettime()`
- `fl_win*()` (all these functions)
- `fl_initialize(argc, argv, x, y, z)` ignores last 3 arguments
- `fl_read_bitmapfile()`, `fl_read_pixmapfile()`
- `fl_addto_browser_chars()`
- `FL_MENU_BUTTON` just draws normally
- `fl_set_bitmapbutton_file()`, `fl_set_pixmapbutton_file()`
- `FL_CANVAS` objects
- `FL_DIGITAL_CLOCK` (comes out analog)
- `fl_create_bitmap_cursor()`, `fl_set_cursor_color()`
- `fl_set_dial_angles()`
- `fl_show_oneliner()`
- `fl_set_choice_shortcut(a, b, c)`
- command log

- Only some of file selector is emulated
- `FL_DATE_INPUT`
- `fl_pup*()` (all these functions)
- `textbox` object (should be easy but I had no sample programs)
- `xyplot` object

1.17.4 Additional Notes

These notes were written for porting programs written with the older IRISGL version of Forms. Most of these problems are the same ones encountered when going from old Forms to XForms:

Does Not Run In Background

The IRISGL library always forked when you created the first window, unless `"foreground()"` was called. FLTK acts like `"foreground()"` is called all the time. If you really want the fork behavior do `"if (fork()) exit(0)"` right at the start of your program.

You Cannot Use IRISGL Windows or `fl_queue`

If a Forms (not XForms) program if you wanted your own window for displaying things you would create a IRISGL window and draw in it, periodically calling Forms to check if the user hit buttons on the panels. If the user did things to the IRISGL window, you would find this out by having the value `FL_EVENT` returned from the call to Forms.

None of this works with FLTK. Nor will it compile, the necessary calls are not in the interface.

You have to make a subclass of `FL_Gl_Window` and write a `draw()` method and `handle()` method. This may require anywhere from a trivial to a major rewrite.

If you draw into the overlay planes you will have to also write a `draw_overlay()` method and call `redraw_overlay()` on the OpenGL window.

One easy way to hack your program so it works is to make the `draw()` and `handle()` methods on your window set some static variables, storing what event happened. Then in the main loop of your program, call `Fl::wait()` and then check these variables, acting on them as though they are events read from `fl_queue`.

You Must Use OpenGL to Draw Everything

The file `<FL/gl.h>` defines replacements for a lot of IRISGL calls, translating them to OpenGL. There are much better translators available that you might want to investigate.

You Cannot Make Forms Subclasses

Programs that call `fl_make_object` or directly setting the handle routine will not compile. You have to rewrite them to use a subclass of `FL_Widget`. It is important to note that the `handle()` method is not exactly the same as the `handle()` function of Forms. Where a Forms `handle()` returned non-zero, your `handle()` must call `do_callback()`. And your `handle()` must return non-zero if it "understood" the event.

An attempt has been made to emulate the "free" widget. This appears to work quite well. It may be quicker to modify your subclass into a "free" widget, since the "handle" functions match.

If your subclass draws into the overlay you are in trouble and will have to rewrite things a lot.

You Cannot Use <device.h>

If you have written your own "free" widgets you will probably get a lot of errors about "getvaluator". You should substitute:

Forms	FLTK
MOUSE_X	Fl::event_x_root()
MOUSE_Y	Fl::event_y_root()
LEFTSHIFTKEY,RIGHTSHIFTKEY	Fl::event_shift()
CAPSLOCKKEY	Fl::event_capslock()
LEFTCTRLKEY,RIGHTCTRLKEY	Fl::event_ctrl()
LEFTALTKEY,RIGHTALTKEY	Fl::event_alt()
MOUSE1,RIGHTMOUSE	Fl::event_state()
MOUSE2,MIDDLEMOUSE	Fl::event_state()
MOUSE3,LEFTMOUSE	Fl::event_state()

Anything else in `getvaluator` and you are on your own...

Font Numbers Are Different

The "style" numbers have been changed because I wanted to insert bold-italic versions of the normal fonts. If you use Times, Courier, or Bookman to display any text you will get a different font out of FLTK. If you are really desperate to fix this use the following code:

```
fl_font_name(3, "*courier-medium-r-no*");
fl_font_name(4, "*courier-bold-r-no*");
fl_font_name(5, "*courier-medium-o-no*");
fl_font_name(6, "*times-medium-r-no*");
fl_font_name(7, "*times-bold-r-no*");
fl_font_name(8, "*times-medium-i-no*");
fl_font_name(9, "*bookman-light-r-no*");
fl_font_name(10, "*bookman-demi-r-no*");
fl_font_name(11, "*bookman-light-i-no*");
```

1.18 Operating System Issues

This appendix describes the operating system specific interfaces in FLTK:

- [Accessing the OS Interfaces](#)
- [The Wayland/X11 hybrid library](#)
- [The UNIX \(X11\) Interface](#)
- [The Windows Interface](#)
- [The Apple OS X Interface](#)
- [The Wayland Interface](#)

1.18.1 Accessing the OS Interfaces

All programs that need to access the operating system specific interfaces must include the following header file:

```
#include <FL/platform.H>
```

This header file will define the appropriate interface for your environment. The pages that follow describe the functionality that is provided for each operating system.

Note

These definitions used to be in [FL/x.H](#) up to FLTK 1.3.x. Usage of [FL/x.H](#) is deprecated since FLTK 1.4.0. You should replace all references of [FL/x.H](#) with [FL/platform.H](#) if your target is FLTK 1.4 or later. [FL/x.H](#) will be retained for backwards compatibility for some releases but will be removed in a later (not yet specified) FLTK release.

WARNING:

The interfaces provided by this header file may change radically in new FLTK releases. Use them only when an existing generic FLTK interface is not sufficient.

1.18.2 The Wayland/X11 hybrid library

By default, the FLTK library is, under Linux and Unix, a Wayland/X11 hybrid which can run FLTK-based apps as Wayland clients or as X11 clients. The choice between running an app as a Wayland or an X11 client is done as follows, when the app runs function [fl_open_display\(\)](#) (that function can be called explicitly by the app or implicitly by FLTK, for example the first time an [Fl_Window](#) is [show\(\)](#)'n) :

- if the app contains a global boolean variable named [fl_disable_wayland](#) and this variable is true, X11 is used;
- if environment variable `FLTK_BACKEND` is not defined, Wayland is used if a Wayland compositor is available, otherwise X11 is used;
- if `$FLTK_BACKEND` equals "wayland", the library makes the app a Wayland client, and stops with error if no Wayland compositor is available;

- if `$FLTK_BACKEND` equals "x11", the library makes the app an X11 client even if a Wayland compositor is available.

The first condition listed above is meant to facilitate conversion of code written for FLTK 1.3.x and containing X11-specific code; add this single statement anywhere in the app's source code:

```
FL_EXPORT bool fl_disable_wayland = true;
```

and the app will always run as an X11 client.

After function `fl_open_display()` has been called, exactly one of the functions `fl_wl_display()` and `fl_x11_display()` returns a non-NULL value. When the former function does, the app runs as a Wayland client, and Wayland-specific functions and symbols described below ([The Wayland Interface](#)) can be used, whereas X11-specific functions and symbols cannot. Otherwise, the app runs as an X11 client, and only X11-specific functions and symbols below ([The UNIX \(X11\) Interface](#)) can be used.

Because a single app can be expected to run either Wayland or X11, it's necessary to use distinct names for global variables and functions in the X11- and the Wayland-specific source code.

Non-default configurations of the FLTK library under Linux/Unix are described in file `README.Wayland.txt`.

1.18.3 The UNIX (X11) Interface

Cross-platform applications should bracket X11-specific source code between `#if defined(FLTK_USE_X11) / #endif` and should ensure function `fl_x11_display()` returns non-NULL before calling X11-specific functions and using X11-specific symbols.

The UNIX interface provides access to the X Window System state information and data structures.

1.18.3.1 Handling Other X Events

```
void Fl::add_handler(int (*f)(int))
```

Installs a function to parse unrecognized events. If FLTK cannot figure out what to do with an event, it calls each of these functions (most recent first) until one of them returns non-zero. If none of them returns non-zero then the event is ignored.

FLTK calls this for any X events it does not recognize, or X events with a window ID that FLTK does not recognize. You can look at the X event in the `fl_xevent` variable.

The argument is the FLTK event type that was not handled, or zero for unrecognized X events. These handlers are also called for global shortcuts and some other events that the widget they were passed to did not handle, for example `FL_SHORTCUT`.

```
extern XEvent *fl_xevent
```

This variable contains the most recent X event.

extern `ulong fl_event_time`

This variable contains the time stamp from the most recent X event that reported it; not all events do. Many X calls like cut and paste need this value.

Window `fl_xid(const FL_Window *)`

Returns the XID for a window, or zero if not `shown()`.

Deprecated Kept for compatibility with FLTK versions before 1.4. Use preferentially `fl_x11_xid(const FL_Window *)` with versions 1.4 and above.

`FL_Window *fl_find(ulong xid)`

Returns the `FL_Window` that corresponds to the given XID, or `NULL` if not found. This function uses a cache so it is slightly faster than iterating through the windows yourself.

Deprecated Kept for compatibility with FLTK versions before 1.4. Use preferentially `fl_x11_find(Window)` with versions 1.4 and above.

int `fl_handle(const XEvent &)`

This call allows you to supply the X events to FLTK, which may allow FLTK to cooperate with another toolkit or library. The return value is non-zero if FLTK understood the event. If the window does not belong to FLTK and the `add_handler()` functions all return 0, this function will return false.

Besides feeding events your code should call `Fl::flush()` periodically so that FLTK redraws its windows.

This function will call the callback functions. It will not return until they complete. In particular, if a callback pops up a modal window by calling `fl_ask()`, for instance, it will not return until the modal function returns.

1.18.3.2 Drawing using Xlib

The following global variables are set before `Fl_Widget::draw()` is called, or by `Fl_Window::make_current()`:

```
extern Display *fl_display; // for compatibility with previous FLTK versions
extern Display *fl_x11_display(); // preferred access starting with FLTK 1.4
extern Window fl_window;
extern GC fl_gc; // for compatibility with previous FLTK versions
extern GC fl_x11_gc(); // preferred access starting with FLTK 1.4
extern int fl_screen;
extern XVisualInfo *fl_visual;
extern Colormap fl_colormap;
```

You must use them to produce Xlib calls. Don't attempt to change them. A typical X drawing call is written like this: `XDrawSomething(fl_display, fl_window, fl_gc, ...);`

Other information such as the position or size of the X window can be found by looking at `Fl_Window::current()`, which returns a pointer to the `Fl_Window` being drawn.

```
unsigned long fl_xpixel(Fl_Color i)
unsigned long fl_xpixel(uchar r, uchar g, uchar b)
```

Returns the X pixel number used to draw the given FLTK color index or RGB color. This is the X pixel that `fl_color()` would use.

```
int fl_parse_color(const char* p, uchar& r, uchar& g, uchar& b)
```

Convert a name into the red, green, and blue values of a color by parsing the X11 color names. On other systems, `fl_parse_color()` can only convert names in hexadecimal encoding, for example `#ff8083`.

```
extern XFontStruct *fl_xfont
```

Points to the font selected by the most recent `fl_font()`. This is not necessarily the current font of `fl_gc`, which is not set until `fl_draw()` is called. If FLTK was compiled with Xft support, `fl_xfont` will usually be 0 and `fl_xftfont` will contain a pointer to the `XftFont` structure instead.

```
extern void *fl_xftfont
```

If FLTK was compiled with Xft support enabled, `fl_xftfont` points to the xft font selected by the most recent `fl_font()`. Otherwise it will be 0. `fl_xftfont` should be cast to `XftFont*`.

1.18.3.3 Changing the Display, Screen, or X Visual

FLTK uses only a single display, screen, X visual, and X colormap. This greatly simplifies its internal structure and makes it much smaller and faster. You can change which it uses by setting global variables *before the first `Fl_Window::show()` is called*. You may also want to call `Fl::visual()`, which is a portable interface to get a full color and/or double buffered visual.

```
int Fl::display(const char *)
```

Set which X display to use. This actually does `putenv("DISPLAY=...")` so that child programs will display on the same screen if called with `exec()`. This must be done before the display is opened. This call is provided under MacOS and Windows but it has no effect.

extern Display *fl_display

The open X display. This is needed as an argument to most Xlib calls. Don't attempt to change it! This is `NULL` before the display is opened.

void [fl_open_display\(\)](#)

Opens the display. Does nothing if it is already open. This will make sure `fl_display` is non-zero. You should call this if you wish to do X calls and there is a chance that your code will be called before the first `show()` of a window.

This may call `Fl::abort()` if there is an error opening the display.

void [fl_x11_use_display\(Display *d\)](#)

Directs FLTK to use a pre-established X11 connection.

void [fl_close_display\(\)](#)

This closes the X connection. You do *not* need to call this to exit, and in fact it is faster to not do so! It may be useful to call this if you want your program to continue without the X connection. You cannot open the display again, and probably cannot call any FLTK functions.

extern int fl_screen

Which screen number to use. This is set by [fl_open_display\(\)](#) to the default screen. You can change it by setting this to a different value immediately afterwards. It can also be set by changing the last number in the [Fl::display\(\)](#) string to "host:0.#".

extern XVisualInfo *fl_visual
extern Colormap fl_colormap

The visual and colormap that FLTK will use for all windows. These are set by [fl_open_display\(\)](#) to the default visual and colormap. You can change them before calling `show()` on the first window. Typical code for changing the default visual is:

```
Fl::args(argc, argv); // do this first so $DISPLAY is set
fl_open_display();
fl_visual = find_a_good_visual(fl_display, fl_screen);
if (!fl_visual) Fl::abort("No good visual");
fl_colormap = make_a_colormap(fl_display, fl_visual->visual, fl_visual->depth);
// it is now ok to show() windows:
window->show(argc, argv);
```

1.18.3.4 Using a Subclass of `Fl_Window` for Special X Stuff

FLTK can manage an X window on a different screen, visual and/or colormap, you just can't use FLTK's drawing routines to draw into it. But you can write your own `draw()` method that uses Xlib (and/or OpenGL) calls only.

FLTK can also manage XID's provided by other libraries or programs, and call those libraries when the window needs to be redrawn.

To do this, you need to make a subclass of `Fl_Window` and override some of these virtual functions:

virtual void `Fl_Window::show()`

If the window is already `shown()` this must cause it to be raised, this can usually be done by calling `Fl_Window::show()`. If not `shown()` your implementation must call either `Fl_X::set_xid()` or `Fl_X::make_xid()`.

An example:

```
void MyWindow::show() {
    if (shown()) {Fl_Window::show(); return;} // you must do this!
    fl_open_display(); // necessary if this is first window
    // we only calculate the necessary visual colormap once:
    static XVisualInfo *visual;
    static Colormap colormap;
    if (!visual) {
        visual = figure_out_visual();
        colormap = XCreateColormap(fl_display, RootWindow(fl_display, fl_screen),
                                   vis->visual, AllocNone);
    }
    Fl_X::make_xid(this, visual, colormap);
}
```

```
Fl_X *Fl_X::set_xid(Fl_Window*, Window xid)
```

Allocate a hidden class called an `Fl_X`, put the XID into it, and set a pointer to it from the `Fl_Window`. This causes `Fl_Window::shown()` to return true.

```
void Fl_X::make_xid(Fl_Window*, XVisualInfo* = fl_visual, Colormap = fl_colormap)
```

This static method does the most onerous parts of creating an X window, including setting the label, resize limitations, etc. It then does `Fl_X::set_xid()` with this new window and maps the window.

virtual void `Fl_Window::flush()`

This virtual function is called by `Fl::flush()` to update the window. For FLTK's own windows it does this by setting the global variables `fl_window` and `fl_gc` and then calling the `draw()` method. For your own windows you might just want to put all the drawing code in here.

The X region that is a combination of all `damage()` calls done so far is in `Fl_X::flx(this)->region`. If `NULL` then you should redraw the entire window. The undocumented function `fl_clip_region()` ($\leftrightarrow$ `XRegion`) will initialize the FLTK clip stack with a region or `NULL` for no clipping. You must set region to `NULL` afterwards as `fl_clip_region()` will own and delete it when done.

If `damage()` & `FL_DAMAGE_EXPOSE` then only X expose events have happened. This may be useful if you have an undamaged image (such as a backing buffer) around.

Here is a sample where an undamaged image is kept somewhere:

```
void MyWindow::flush() {
    fl_clip_region(Fl_X::flx(this)->region);
    Fl_X::flx(this)->region = 0;
    if (damage() != 2) {... draw things into backing store ...}
    ... copy backing store to window ...
}
```

Note

For compatibility with FLTK versions before 1.4, member function `Fl_X::flx(Fl_Window*)` can also be written `Fl_X::i(Fl_Window*)`.

virtual void `Fl_Window::hide()`

Destroy the window server copy of the window. Usually you will destroy contexts, pixmaps, or other resources used by the window, and then call `Fl_Window::hide()` to get rid of the main window identified by `xid()`. If you override this, you must also override the destructor as shown:

```
void MyWindow::hide() {
    if (mypixmap) {
        XFreePixmap(fl_display, mypixmap);
        mypixmap = 0;
    }
    Fl_Window::hide(); // you must call this
}
```

virtual void `Fl_Window::~~Fl_Window()`

Because of the way C++ works, if you override `hide()` you *must* override the destructor as well (otherwise only the base class `hide()` is called):

```
MyWindow::~MyWindow() {
    hide();
}
```

Note

Access to the `Fl_X` hidden class requires to `#define FL_INTERNALS` before compilation.

1.18.3.5 Setting the Icon of a Window

FLTK recommends to set window icons using these platform-independent methods: `FL_Window::icon(const FL_RGB_Image *)` and `FL_Window::icons(const FL_RGB_Image *[], int)`. See also methods setting default window icons `FL_Window::default_icon(const FL_RGB_Image *)` and `FL_Window::default_icons(const FL_RGB_Image *[], int)`.

FLTK on X11 also supports, for backward compatibility, use of the deprecated method `FL_Window::icon(const void *)` as follows :

Sets the icon for the window to the passed pointer. You will need to cast the icon `Pixmap` to a `char*` when calling this method. To set a monochrome icon using a bitmap compiled with your application use:

```
#include "icon.xbm"

fl_open_display(); // needed if display has not been previously opened

Pixmap p = XCreateBitmapFromData(fl_display, DefaultRootWindow(fl_display),
                                icon_bits, icon_width, icon_height);

window->icon((const void*)p);
```

To use a multi-colored icon, the XPM format and library should be used as follows:

```
#include <X11/xpm.h>
#include "icon.xpm"

fl_open_display(); // needed if display has not been previously opened

Pixmap p, mask;

XpmCreatePixmapFromData(fl_display, DefaultRootWindow(fl_display),
                        icon_xpm, &p, &mask, NULL);

window->icon((const void *)p);
```

When using the Xpm library, be sure to include it in the list of libraries that are used to link the application (usually `-lXpm`).

NOTE:

You must call `FL_Window::show(int argc, char** argv)` for the icon to be used. The `FL_Window::show()` method does not bind the icon to the window.

Any window icon must be set with the above methods before the window is shown.

1.18.3.6 X Resources

When the `FL_Window::show(int argc, char** argv)` method is called, FLTK looks for the following X resources:

- `background` - The default background color for widgets (color).
- `dndTextOps` - The default setting for drag and drop text operations (boolean).
- `foreground` - The default foreground (label) color for widgets (color).
- `scheme` - The default scheme to use (string).
- `selectBackground` - The default selection color for menus, etc. (color).
- `Text.background` - The default background color for text fields (color).
- `tooltips` - The default setting for tooltips (boolean).
- `visibleFocus` - The default setting for visible keyboard focus on non-text widgets (boolean).

Resources associated with the first window's `FL_Window::xclass()` string are queried first, or if no class has been specified then the class `"fltk"` is used (e.g. `fltk.background`). If no match is found, a global search is done (e.g. `*background`).

1.18.3.7 Display Scaling Factor

FLTK uses the value of the `Xft.dpi` resource divided by 96. to initialize the display scaling factor. That is also what is done by the gnome and KDE desktops.

1.18.4 The Windows Interface

Cross-platform applications should bracket Windows-specific source code between `#ifdef _WIN32 / #endif`.

The Windows interface provides access to the Windows GDI state information and data structures.

1.18.4.1 Using filenames with non-ASCII characters

In FLTK, all strings, including filenames, are UTF-8 encoded. The utility functions `fl_fopen()` and `fl_open()` allow to open files potentially having non-ASCII names in a cross-platform fashion, whereas the standard `fopen()/open()` functions fail to do so.

1.18.4.2 Responding to WM_QUIT

FLTK will intercept `WM_QUIT` messages that are directed towards the thread that runs the main loop. These are converted to `SIGTERM` signals via `raise()`. This allows you to deal with outside termination requests with the same code on both Windows and UNIX systems. Other processes can send this message via `PostThreadMessage()` in order to request, rather than force your application to terminate.

1.18.4.3 Handling Other Windows API Messages

By default a single `WNDCLASSEX` called "FLTK" is created. All `Fl_Window`'s are of this class unless you use `Fl_Window::xclass()`. The window class is created the first time `Fl_Window::show()` is called.

You can probably combine FLTK with other libraries that make their own window classes. The easiest way is to call `Fl::wait()`, as it will call `DispatchMessage()` for all messages to the other windows. If necessary you can let the other library take over as long as it calls `DispatchMessage()`, but you will have to arrange for the function `Fl::flush()` to be called regularly so that widgets are updated, timeouts are handled, and the idle functions are called.

```
extern MSG fl_msg
```

This variable contains the most recent message read by `GetMessage()`, which is called by `Fl::wait()`. This may not be the most recent message sent to an FLTK window, because silly Windows calls the handle procedures directly for some events (sigh).

```
void Fl::add_handler(int (*f)(int))
```

Installs a function to parse unrecognized messages sent to FLTK windows. If FLTK cannot figure out what to do with a message, it calls each of these functions (most recent first) until one of them returns non-zero. The argument passed to the functions is the FLTK event that was not handled or zero for unknown messages. If all the handlers return zero then FLTK calls `DefWindowProc()`.

```
HWND fl_xid(const Fl_Window *)
```

Returns the window handle for a `Fl_Window`, or zero if not `shown()`.

```
Fl_Window *fl_find(HWND xid)
```

Returns the `Fl_Window` that corresponds to the given window handle, or `NULL` if not found. This function uses a cache so it is slightly faster than iterating through the windows yourself.

1.18.4.4 Drawing Things Using the Windows GDI

When the virtual function `Fl_Widget::draw()` is called, FLTK stores all the extra arguments you need to make a proper GDI call in some global variables:

```
extern HINSTANCE fl_display; // for compatibility with previous FLTK versions
extern HINSTANCE fl_win32_display(); // preferred access starting with FLTK 1.4
extern HWND fl_window;
extern HDC fl_gc; // for compatibility with previous FLTK versions
extern HDC fl_win32_gc(); // preferred access starting with FLTK 1.4
COLORREF fl_RGB();
HPEN fl_pen();
HBRUSH fl_brush();
```

These global variables are set before `Fl_Widget::draw()` is called, or by `Fl_Window::make_current()`. You can refer to them when needed to produce GDI calls, but don't attempt to change them. The functions return GDI objects for the current color set by `fl_color()` and are created as needed and cached. A typical GDI drawing call is written like this:

```
DrawSomething(fl_gc, ..., fl_brush());
```

It may also be useful to refer to `Fl_Window::current()` to get the window's size or position.

1.18.4.5 HighDPI support

FLTK apps for the Windows platform are by default "Per-monitor DPI-aware V2". This means that any window automatically adjusts its physical size in relation to the scaling factor of the display where it maps. This also means that all drawings (e.g., text, lines, images) take advantage of the full resolution of the display in use. FLTK apps may also use the manifest mechanism to declare their level of DPI awareness. The FLTK library adapts to the DPI awareness level set in the app's manifest, which can be lower than the default level if the manifest sets it so.

1.18.4.6 Display Scaling Factor

FLTK uses the value given by function `GetDpiForMonitor()` divided by 96. to initialize the scaling factor of each display in the system. This matches the value of "Change the size of text, apps and other items" found in section "System" subsection "Display" of Windows settings.

1.18.4.7 Setting the Icon of a Window

FLTK recommends to set window icons using these platform-independent methods: `Fl_Window::icon(const Fl_RGB_Image *)` and `Fl_Window::icons(const Fl_RGB_Image *[], int)`. See also methods setting default window icons `Fl_Window::default_icon(const Fl_RGB_Image *)` and `Fl_Window::default_icons(const Fl_RGB_Image *[], int)`.

FLTK on Windows also supports, for backward compatibility, use of the deprecated method `Fl_Window::icon(const void *)` as follows :

Set the icon for the window to the passed pointer. You will need to cast the `HICON` handle to a `char*` when calling this method. To set the icon using an icon resource compiled with your application use:

```
window->icon((const void *)LoadIcon(fl_display, MAKEINTRESOURCE(IDI_ICON)));
```

You can also use the `LoadImage()` and related functions to load specific resolutions or create the icon from bitmap data.

NOTE:

You must call `Fl_Window::show(int argc, char** argv)` for the icon to be used. The `Fl_Window::show()` method does not bind the icon to the window.

Any window icon must be set with the above methods before the window is shown.

1.18.4.8 How to Not Get a MSDOS Console Window

Windows has a really stupid mode switch stored in the executables that controls whether or not to make a console window.

To always get a console window you simply create a console application (the `"/SUBSYSTEM:CONSOLE"` option for the linker). For a GUI-only application create a Windows application (the `"/SUBSYSTEM:WINDOWS"` option for the linker).

FLTK includes a `WinMain()` function that calls the ANSI standard `main()` entry point for you. *This function creates a console window when you use the debug version of the library.*

Windows applications without a console cannot write to `stdout` or `stderr`, even if they are run from a console window. Any output is silently thrown away. Additionally, Windows applications are run in the background by the console, although you can use `"start /wait program"` to run them in the foreground.

1.18.4.9 Known Windows Bugs and Problems

The following is a list of known bugs and problems in the Windows version of FLTK:

- If a program is deactivated, `Fl::wait()` does not return until it is activated again, even though many events are delivered to the program. This can cause idle background processes to stop unexpectedly. This also happens while the user is dragging or resizing windows or otherwise holding the mouse down. We were forced to remove most of the efficiency FLTK uses for redrawing in order to get windows to update while being moved. This is a design error in Windows and probably impossible to get around.
- `Fl_Gl_Window::can_do_overlay()` returns true until the first time it attempts to draw an overlay, and then correctly returns whether or not there is overlay hardware.
- `SetCapture` (used by `Fl::grab()`) doesn't work, and the main window title bar turns gray while menus are popped up.
- Compilation with `gcc 3.4.4` and `-Os` exposes an optimisation bug in `gcc`. The symptom is that when drawing filled circles only the perimeter is drawn. This can for instance be seen in the symbols demo. Other optimisation options such as `-O2` and `-O3` seem to work OK. More details can be found in STR#1656

1.18.5 The Apple OS X Interface

Cross-platform applications should bracket macOS-specific source code between `#if defined(__APPLE__)` and `!defined(FLTK_USE_X11) / #endif`.

FLTK supports Apple OS X using the Apple Cocoa library. Older versions of MacOS are no longer supported.

Control, Option, and Command Modifier Keys

FLTK maps the Mac 'control' key to `FL_CTRL`, the 'option' key to `FL_ALT` and the 'Apple' key to `FL_META`. Furthermore, `FL_COMMAND` designates the 'Apple' key on Mac OS X and the 'control' key on other platforms. Keyboard events return the key name in `Fl::event_key()` and the keystroke translation in `Fl::event_text()`. For example, typing Option-Y on a Mac US keyboard will set `FL_ALT` in `Fl::event_state()`, set `Fl::event_key()` to 'y' and return the Yen symbol in `Fl::event_text()`.

Right Click simulation with Ctrl Click

The Apple HIG guidelines indicate applications should support 'Ctrl Click' to simulate 'Right Click' for e.g. context menus, so users with one-button mice and one-click trackpads can still access right-click features. However, paraphrasing [Manolo's comment on the fltk.coredev newsgroup](#):

- *FLTK does /not/ support Ctrl-Click == Right Click itself because Mac OS X event processing doesn't support this at the system level: the system reports left-clicks with the ctrl modifier when the user ctrl-clicks, and OS X system preferences don't allow changing this behavior. Therefore, applications must handle simulation of Right Click with Ctrl Click in the application code.*

Ian MacArthur provided the following handle() method code snippet showing an example of how to do this:

```
case FL_PUSH:
{
    int btn = Fl::event_button();
#ifdef __APPLE__
    int ev_state = Fl::event_state();
#endif
    //
    // Context menu can be called up in one of two ways: -
    // 1 - right click, as normally used on Windows and Linux
    // 2 - Ctrl + left click, as sometimes used on Mac
    //
#ifdef __APPLE__
    // On apple, check right click, and ctrl+left click
    if ((btn == FL_RIGHT_MOUSE) || (ev_state == (FL_CTRL | FL_BUTTON1)))
#else
    // On other platforms, only check right click as ctrl+left is used for selections
    if (btn == FL_RIGHT_MOUSE)
#endif
    {
        // Did we right click on the object?..
```

There is a thread about this subject on fltk.coredev (Aug 1-14, 2014) entitled "[RFC] Right click emulation for one button mouse on Mac".

Apple "Quit" Event

When the user presses Cmd-Q or requests a termination of the application, FLTK sends an FL_CLOSE event to all open windows. If any window remains open, the termination request aborts. If all windows close, the application's event loop terminates, that is, `Fl::run()` returns. The application can then follow FLTK's normal termination path executing cleanup code that may be programmed after termination of the event loop, and returning from `main()`. Function `Fl::program_should_quit()` allows to detect whether the event loop terminated because of a program termination request.

Apple "Open" Event

Whenever the user drops a file onto an application icon, OS X generates an Apple Event of the type "Open". You can have FLTK notify you of an Open event by calling the `fl_open_callback()` function.

`void fl_open_display()`

Opens the display. Does nothing if it is already open. You should call this if you wish to do Cocoa or Quartz calls and there is a chance that your code will be called before the first `show()` of a window.

Window `fl_xid(const FL_Window *)`

Returns the window reference for an `FL_Window`, or `NULL` if the window has not been shown. This reference is a pointer to an instance of the subclass `FLWindow` of Cocoa's `NSWindow` class.

`FL_Window *fl_find(Window xid)`

Returns the `FL_Window` that corresponds to the given window reference, or `NULL` if not found.

void `fl_mac_set_about( FL_Callback *cb, void *user_data, int shortcut)`

Attaches the callback `cb` to the "About myprog" item of the system application menu. `cb` will be called with `NULL` first argument and `user_data` second argument. This MacOS-specific function is deprecated in FLTK 1.4 and replaced by `FL_Sys_Menu_Bar::about(FL_Callback *cb, void *data)` which is cross-platform.

`FL_Sys_Menu_Bar` class

The `FL_Sys_Menu_Bar` class allows to build menu bars that, on Mac OS X, are placed in the system menu bar (at top-left of display), and, on other platforms, at a user-chosen location of a user-chosen window.

1.18.5.1 Setting the icon of an application

- First, create a `.icns` file containing several copies of your icon of decreasing sizes. This can be done using the Preview application or the Icon Composer application available in "Graphics Tools for Xcode". To create a high resolution icon file, it is necessary to use the `iconutil` command-line utility.
- Put your `.icns` file in the Resources subdirectory of your application bundle.
- Add these two lines to the `Info.plist` file of your application bundle

```
<key>CFBundleIconFile</key>
<string>foo.icns</string>
```

replacing `foo` by your application name. If you use Xcode, just add your `.icns` file to your application target.

1.18.5.2 Drawing Things Using Quartz

All code inside `FL_Widget::draw()` is expected to call Quartz drawing functions. The Quartz coordinate system is flipped to match FLTK's coordinate system. The origin for all drawing is in the top left corner of the enclosing `FL_Window`. The function `fl_mac_gc()` returns the appropriate Quartz 2D drawing environment (of type `CGContextRef`). For compatibility with previous FLTK versions, deprecated global variable `fl_gc` gives the same value.

Include `FL/platform.H` to declare the `fl_mac_gc()` function (or the `fl_gc` variable).

1.18.5.3 Internationalization

All FLTK programs contain an application menu with, e.g., the About xxx, Hide xxx, and Quit xxx items. This menu can be internationalized/localized by any of two means.

- using the [Fl_Mac_App_Menu](#) class.
- using the standard Mac OS X localization procedure. Create a language-specific .lproj directory (e.g., German.lproj) in the Resources subdirectory of the application bundle. Create therein a Localizable.strings file that translates all menu items to this language. The German Localizable.strings file, for example, contains:

```
"About %@", "Über %@";
"Print Front Window"="Frontfenster drucken";
"Services" = "Dienste";
"Hide %@", "%@" ausblenden";
"Hide Others"="Andere ausblenden";
"Show All"="Alle einblenden";
"Quit %@", "%@" beenden";
```

Set "Print Front Window" = ""; therein so the application menu doesn't show a "Print Front Window" item. To localize the application name itself, create a file InfoPlist.strings in each .lproj directory and put CFBundleName = "localized name"; in each such file.

1.18.5.4 OpenGL and 'retina' displays

It is possible to have OpenGL produce graphics at the high pixel resolution allowed by the so-called 'retina' displays present on recent Apple hardware. For this, call

```
Fl::use_high_res_GL(1);
```

before any [Fl_Gl_Window](#) is shown. Also, adapt your [Fl_Gl_Window::draw\(\)](#) and [Fl_Gl_Window::draw_overlay\(\)](#) methods replacing

```
glViewport(0, 0, w(), h());
```

by

```
glViewport(0, 0, pixel_w(), pixel_h());
```

making use of the [Fl_Gl_Window::pixel_w\(\)](#) and [Fl_Gl_Window::pixel_h\(\)](#) methods that return the width and height of the GL scene in pixels: if the [Fl_Gl_Window](#) is mapped on a retina display, these methods return twice as much as reported by [Fl_Widget::w\(\)](#) and [Fl_Widget::h\(\)](#); if it's mapped on a regular display, they return the same values as [w\(\)](#) and [h\(\)](#). These methods dynamically change their values if the window is moved into/out from a retina display. If [Fl::use_high_res_GL\(1\)](#) is not called, all [Fl_Gl_Window](#)'s are drawn at low resolution. These methods are useful on all platforms because [Fl_Gl_Window::w\(\)](#) and [Fl_Gl_Window::h\(\)](#) don't return, on HighDPI displays, the quantities in pixels necessary to OpenGL functions .

The [Fl_Gl_Window::pixels_per_unit\(\)](#) method is useful when the OpenGL code depends on the pixel dimension of the GL scene. This occurs, e.g., if a window's [handle\(\)](#) method uses [Fl::event_x\(\)](#) and [Fl::event_y\(\)](#) whose returned values should be multiplied by [Fl_Gl_Window::pixels_per_unit\(\)](#) to obtain the adequate pixel units. This method may also be useful, for example, to adjust the width of a line in a high resolution GL scene.

1.18.5.5 FI_Double_Window

OS X double-buffers all windows automatically. On OS X, [FI_Window](#) and [FI_Double_Window](#) are handled internally in the same way.

1.18.5.6 Mac File System Specifics

Resource Forks (OS X pre 10.6)

FLTK does not access the resource fork of an application. However, a minimal resource fork must be created for OS X applications. Starting with OS X 10.6, resource forks are no longer needed.

Caution (OS X 10.2 and older):

When using UNIX commands to copy or move executables, OS X will NOT copy any resource forks! For copying and moving use CpMac and MvMac respectively. For creating a tar archive, all executables need to be stripped from their Resource Fork before packing, e.g. "DeRez fluid > fluid.r". After unpacking the Resource Fork needs to be reattached, e.g. "Rez fluid.r -o fluid".

It is advisable to use the Finder for moving and copying and Mac archiving tools like Sit for distribution as they will handle the Resource Fork correctly.

Mac File Paths

FLTK uses UTF-8-encoded UNIX-style filenames and paths.

See also

[Mac OS X-specific symbols](#)

1.18.6 The Wayland Interface

Cross-platform applications should bracket Wayland-specific source code between `#ifdef FLTK_USE_WAYLAND` / `#endif` and should ensure function `fl_wl_display()` returns non-NULL before calling Wayland-specific functions and using Wayland-specific symbols.

```
extern struct wl_display *fl_wl_display();
```

After `fl_open_display()` has run, function `fl_wl_display()` returns a pointer to the struct `wl_display` representing the connection between the application and Wayland. For example, `wl_display_get_fd(fl_wl_display())` gives the file descriptor one can use to communicate with the Wayland compositor according to the Wayland protocol.

```
struct wld_window *fl_wl_xid(const FI_Window *)
```

Returns a pointer to an FLTK-defined structure holding Wayland-related data created when a window gets `show()`, or NULL if not `show()`.

[Fl_Window](#) *fl_wl_find(struct wld_window * wld_win)

Returns the [Fl_Window](#) that corresponds to the given Window, or NULL if not found.

struct wl_surface *fl_wl_surface(struct wld_window *wld_win)

Returns a pointer to the struct `wl_surface` corresponding to a `show()` top-level window or subwindow.

cairo_t *fl_wl_gc()

Drawing natively to a Wayland window : Within an overridden [Fl_Widget::draw\(\)](#) method, or after a call to [Fl_Window::make_current\(\)](#), it's possible to draw using the Cairo library. Function [fl_wl_gc\(\)](#) returns the adequate `cairo_t*` value. Regular FLTK coordinates, with top-left origin, are to be used. All FLTK-defined drawing functions (e.g., [fl_rect\(\)](#), [fl_draw\(\)](#)) can be used too.

void [fl_close_display\(\)](#)

This closes the Wayland connection. You do not need to call this to exit. It may be useful to call this if you want your program to continue without the Wayland connection. You cannot open the display again, and cannot call any FLTK functions.

See also

[wayland.H](#) for all functions specific of the Wayland platform.

1.18.6.1 HiDPI display support

FLTK Wayland apps automatically scale according to the Wayland-defined, integer-valued scale factor. On a HiDPI display, it's enough to set this factor to 2 for any FLTK app to be drawn using twice as many pixels and thus to be as readable as it is on a regular display. With the gnome and KDE-plasma desktops, that is achieved in the "Displays" section of the "Settings" application, selecting 200 % for the "Scale" parameter. In addition to this, FLTK apps can also be scaled up or down typing `ctrl+/-/0/` and with the `FLTK_SCALING_FACTOR` environment variable.

1.18.6.2 Window icons

Standard FLTK functions `FL_Window::icon(const FL_RGB_Image*)`, `FL_Window::icons(const FL_RGB_Image*[], int)`, `FL_Window::default_icon(const FL_RGB_Image*)` and `FL_Window::default_icons(const FL_RGB_Image*[], int)` have no effect on the Wayland platform. The equivalent of a call to `FL_Window::default_icon(const FL_RGB_Image*)` to set the application-specific window icon can be obtained as follows, using FLTK's editor app as an example:

- create a text file named `editor.desktop` containing :

```
[Desktop Entry]
Version=1.0
Type=Application
Name=Editor
Name[fr]=Editeur
Comment=FLTK editor
Exec=editor %F
Icon=/path/to/icon/file/editor.svg
MimeType=text/plain
```

- The `Name=` line therein determines the string displayed when the app runs.
- Optionally, one or more `Name[locale]=` lines can be used to set locale-specific app names.
- The `Icon=` line accepts also `.png` files.
- Put this file in `/usr/local/share/applications/` so it's available to all system users or in `$HOME/.local/share/applications/` so it's available to a single user.

1.18.6.3 Window titlebars

Wayland supports both client-side window decoration (CSD), where client applications are responsible for drawing window titlebars, and server-side window decoration (SSD), where the Wayland compositor itself draws window titlebars. Among 4 tested Wayland compositors, Mutter (gnome's compositor) and Weston use CSD mode whereas the KWin and Sway compositors use SSD mode. When running in CSD mode, FLTK uses a library called `libdecor` to draw titlebars. The `libdecor` library has been conceived to use various plug-in's to draw titlebars in various fashions intended to match any desktop's preferred titlebar style. FLTK supports drawing titlebars with any `libdecor` plug-in via an environment variable called `LIBDECOR_PLUGIN_DIR` which can be given the name of a directory containing the desired plug-in. When `LIBDECOR_PLUGIN_DIR` is not defined, or points to a directory that doesn't contain a `libdecor` plug-in, FLTK uses its built-in plug-in to draw titlebars. That is the most common situation, until `libdecor` plug-in's become available for popular UNIX desktops.

1.19 Migrating Code from FLTK 1.3 to 1.4

This appendix describes the differences between FLTK 1.3.x and FLTK 1.4.x functions and classes and potential requirements to change source code. We also explain how code can be made compatible so it can be compiled by both FLTK 1.3.x and 1.4.x.

If you need to migrate your code from prior FLTK versions to FLTK 1.4, please consult the relevant appendices in the FLTK 1.3 online documentation or by downloading the FLTK 1.3 documentation. See https://www.fltk.org/doc-1.3/migration_1_3.html and/or <https://www.fltk.org/software.php>, respectively.

1.19.1 Changes in Header Files

We strive to include only necessary header files in the public headers of the FLTK library to reduce dependencies and hence compile times.

We try to avoid including system header files as far as possible. Known exceptions are `<stdio.h>` where file system structures and functions are visible in the public API, for instance `FILE*`, and sometimes essential header files like `<stdlib.h>` and/or `<stddef.h>`. Some required system headers **may** be included in platform specific header files like `<FL/platform.H>` or `<FL/platform_types.h>`.

In earlier versions (1.3.x) some of the public FLTK headers included some not strictly required system headers by accident.

The consequence for building user programs with FLTK 1.4 is: if you require a system or FLTK header in your user program that you don't `#include` explicitly but which has been included by FLTK 1.3.x your FLTK 1.3 program may issue compiler errors or warnings about missing header files or missing declarations when compiled with FLTK 1.4.

This is not a fault of FLTK 1.4 but a fault of the source code that did not include all required headers.

In FLTK 1.4 inclusion of `<FL/Fl.H>` is no longer a strict requirement as it was required and documented in FLTK 1.3.x. In FLTK 1.4 you may still need to `'#include <FL/Fl.H>'` if you are using enumerations or methods of class `Fl` like `Fl::run()` but there are exceptions where this header is included by other FLTK headers, like `Fl_Window.H` and other subclasses.

Suggested solution: include all FLTK and system header files your source code requires explicitly and don't rely on FLTK headers to include a particular header file. If you want your code to be as much as possible compatible with FLTK 1.3.x, then you should `'#include <FL/Fl.H>'` as required by 1.3.x.

You don't need to include headers of base classes - this is done by all FLTK headers as required. Besides that you need to include some support headers if you use FLTK functions like `fl_choice()` and others. This is described in the function's documentation (if a required header is missing in the docs this is a bug).

If you follow these rules your program will be compatible with both FLTK 1.3.x and FLTK 1.4.x as long as you use only functions and classes defined in FLTK 1.3.

1.19.2 Fl_Preferences

Starting with FLTK 1.3, preference databases are expected to be in UTF-8 encoding. Previous databases were stored in the current character set or code page which renders them incompatible for text entries using international characters.

Starting with FLTK 1.4, searching a valid path to store the preference files has changed slightly. Please see `Fl_Preferences::Fl_Preferences(Root, const char*, const char*)` for details.

On Unix/Linux platforms new FLTK preference files are stored using the [XDG Base Directory Specification](#) which means in essence that user preference files are stored in the user's home directory under the subdirectory `.config`, i.e. in `$HOME/.config/fltk.org/` rather than `$HOME/.fltk/fltk.org/`. Existing preference files are still found and used, hence this new location is optional.

You may want to move the preference files from their old locations to their new locations as documented in `Fl_Preferences::Fl_Preferences(Root, const char*, const char*)`.

New `Fl_Preferences` types `Fl_Preferences::USER_L`, `Fl_Preferences::SYSTEM_L` and some more combinations with `"_L"` suffix have been defined to make preference files independent of the current locale. This is particularly important for floating point data which is stored in text form with varying decimal separator depending on the locale (either `'.'` or `','`). You may want to change your program to use these new constants instead of those without the `"_L"` suffix. For more information see the documentation of [Fl_Preferences](#).

1.19.3 Fl::add_timeout and friends

Since FLTK 1.4.0 internal timeout handling has been unified across platforms. This ensures equal timeout handling, improved accuracy of `Fl::repeat_timeout()`, and easier maintenance (less potential for errors).

This will very likely not affect user code, however there is one subtle exception on macOS and Windows: in FLTK 1.3.x these platforms used system timers to schedule timeouts. Since FLTK 1.4.0 all platforms use the same internal timer management that was previously only used on Unix/Linux/X11. The consequence of this change is that the FLTK event loop needs to be executed to trigger timeout events, i.e. you must either call `Fl::wait()` repeatedly or start the event loop with `Fl::run()`.

Code that did not execute the event loop and relied on the system timers has never been cross platform compatible, i.e. it wouldn't work on Unix/Linux. An example would be code that opened a splash window, scheduled a timeout with `Fl::add_timeout()`, and waited for the timer event w/o running the FLTK event loop. Such code must be modified to execute `Fl::run()` and/or use `Fl::wait()`.

1.19.4 New FL_OVERRIDE Macro

FLTK 1.4 defines a new macro `FL_OVERRIDE` as "override" if a recent C++ standard (C++11 or higher) is used to compile your code.

This macro is currently defined in `FL/fl_attr.h` but this may change in a future release. It is enough to `#include <FL/Fl.H>` to enable this macro.

Unfortunately Visual Studio does not define a meaningful value of `__cplusplus` to detect the C++ standard. Hence we use the Visual Studio version (2015 or higher) to decide whether we can define `FL_OVERRIDE` or not.

The `FL_OVERRIDE` macro is used to decorate declarations of overridden virtual methods in subclasses. Example code from `FL/Fl_Window.H`:

```
int handle(int) FL_OVERRIDE;
void resize(int X, int Y, int W, int H) FL_OVERRIDE;
Fl_Window * as_window() FL_OVERRIDE { return this; }
```

The `FL_OVERRIDE` macro translates to `'override'` on newer compilers and to an empty string for older compilers.

We recommend to add this to your overridden virtual methods in subclasses derived from FLTK base classes (widgets) and to compile with C++ standard C++11 or higher to enable the compiler to detect some errors if methods are not overridden correctly.

You don't need to declare the overridden methods `'virtual'` if you use `FL_OVERRIDE` or the keyword `override`.

Hint: For the GCC and clang compilers you can enable the warning `'-Wsuggest-override'` to detect where you may (want to) add the `FL_OVERRIDE` macro.

1.19.5 Fl_Image::copy() 'const'

Since FLTK 1.4.0 the virtual method `Fl_Image::copy()` has been declared 'const' so read-only ('const') images can be copied w/o casts.

This will very likely not affect user code. However, if you derived your own class from any of the `Fl_*_Image` variants **and** you overrode `Your_Image::copy()` then you **must** declare this 'const' as well, i.e. you must add the keyword 'const' to the declaration of `copy()` in your header file and in the implementation.

We suggest to add the new `FL_OVERRIDE` macro or the keyword 'override' (see above) to your own overridden method declarations to enable the compiler to detect such incompatibilities.

Code example in header file:

```
class Your_Image {
    // ...
    Fl_Image *copy() const FL_OVERRIDE;
    Fl_Image *copy(int w, int h) const FL_OVERRIDE;
};
```

Note the `'const'` attribute **and** the `FL_OVERRIDE` macro.

1.19.6 Using X11 specific code with a "hybrid" FLTK library

Read this section if your FLTK 1.3 program uses X11 specific code with platform specific guards like

```
#if defined(WIN32)
    // Windows specific code here
#elif defined(__APPLE__)
    // macOS specific code here
#else
    // X11 specific code here
#endif
```

or similar.

You may skip this section if you don't use platform (X11) specific code.

FLTK 1.4 introduced Wayland support on Unix-like systems that support Wayland at runtime. The default build selects the Wayland runtime if it exists at program startup and falls back to using X11 if Wayland is not supported. On all currently known Wayland-enabled systems X11 programs are still supported by using "XWayland" which must be installed and enabled on the system. This is usually the case (as of this writing in November 2024).

Since using the Wayland runtime is the default you may get runtime errors if your program uses X11 specific code with the Wayland runtime.

There are two solutions:

1. Change the conditional code shown above everywhere in your program. This is the best and recommended long-term solution but may require some work.
2. Disable the Wayland backend (runtime) for your program. This can be a short-term solution for quick "porting" success.

Details about solution 1 are beyond the scope of this documentation but here's an example how such code can be rewritten in FLTK 1.4 for all platforms. This code is now in `test/fltk-versions.cxx` but may be changed in the future.

Change this as you need.

```
static const char *get_platform() {
    #if defined(_WIN32)
        return "Windows";
    #elif defined(FLTK_USE_X11) || defined(FLTK_USE_WAYLAND)
        #if defined(FLTK_USE_X11)
            if (fl_x11_display())
                return "Unix/Linux (X11)";
        #endif
        #if defined(FLTK_USE_WAYLAND)
            if (fl_wl_display())
                return "Unix/Linux (Wayland)";
        #endif
        return "X11 or Wayland (backend unknown or display not opened)";
    #elif defined(__APPLE__)
        return "macOS (native)";
    #endif
    return "platform unknown, unsupported, or display not opened";
}
```

Solution 2 is described in chapter 2.1 of `README.Wayland.txt`. Please read this chapter if you need to support X11 specific code w/o the Wayland runtime.

Excerpt from `README.Wayland.txt`:

It is possible to force a program linked to a Wayland-enabled FLTK library to use X11 in all situations by putting this declaration somewhere in the source code:

```
FL_EXPORT bool fl_disable_wayland = true;
```

Please note that you may need to do more to link and run your program successfully, depending on the build system.

Additional info can be found in chapter [The Wayland/X11 hybrid library](#).

1.19.7 Modern CMake

FLTK 1.4.0 supports "modern" CMake rather than old or "classic" CMake which was used in FLTK 1.3.x. Modern CMake was introduced in CMake 3.0 (~ 2014) and further developed in later CMake versions. FLTK 1.4.0 requires at least CMake 3.15 (~ 2019) as of February 2024.

There are a lot of advantages that motivated this transition (mentioning only some):

- easier to use for projects using FLTK
- better structure
- uses CMake targets rather than variables
- embeddable in user projects via `FetchContent()` etc.
- embeddable in user projects via `add_subdirectory()`
- better coexistence with main projects if built as a subproject

Note that CMake targets can provide all required build flags and build dependencies which is the main advantage for user projects. For instance, instead of linking both `fltk` and `fltk_images` you need only `fltk_images` and `fltk` is linked in automatically.

Unfortunately there is one drawback you may encounter: Several CMake build option names have been changed, compared to FLTK 1.3.x. This is due to the fact that CMake cache variables are shared between the main (aka superbuild) project and all subprojects. Therefore all FLTK options are now prefixed with `FLTK_`.

This feature is now CMake standard and very common in newer projects. The CMake developers recommend strongly to use modern CMake.

We took the opportunity to redesign all CMake related options and target names for FLTK 1.4.0 to avoid changing these names later. Note that CMake support in 1.3.x was only experimental and the one in FLTK 1.4 (Git) up to the official release was beta state by definition. We apologize for all inconveniencies, hope that this is one of the rare exceptions in FLTK development, and that the new names are now stable as usual.

Changes in Detail:

Since FLTK 1.4.0 CMake target names are "namespaced", i.e. they are created with the prefix `'fltk::'` and the old prefix `'fltk_'` has been stripped off as far as the `CMakeLists.txt` file of user projects is concerned. The known filenames on disk did not change though.

The shared library target names use the common suffix `"-shared"` rather than `"_SHARED"`.

The library `'fltk_cairo'` is no longer used. Its functionality has been included in `libfltk`. FLTK 1.4.0 creates a dummy (empty) `libfltk_cairo` for backwards compatibility only. Please remove `fltk_cairo` from your projects and use only `'fltk::fltk'` and/or the other libraries instead.

For more information and documentation of all options please refer to the file `README.CMake.txt` in the FLTK root directory.

Old and New Library Targets:

Library	Old Target	New Target	Shared Library Target
<code>fltk</code>	<code>fltk</code>	<code>fltk::fltk</code>	<code>fltk::fltk-shared</code>
<code>fltk_forms</code>	<code>fltk_forms</code>	<code>fltk::forms</code>	<code>fltk::forms-shared</code>
<code>fltk_gl</code>	<code>fltk_gl</code>	<code>fltk::gl</code>	<code>fltk::gl-shared</code>
<code>fltk_images</code>	<code>fltk_images</code>	<code>fltk::images</code>	<code>fltk::images-shared</code>
<code>fltk_jpeg</code>	<code>fltk_jpeg</code>	<code>fltk::jpeg</code>	<code>fltk::jpeg-shared</code>
<code>fltk_png</code>	<code>fltk_png</code>	<code>fltk::png</code>	<code>fltk::png-shared</code>
<code>fltk_z</code>	<code>fltk_z</code>	<code>fltk::z</code>	<code>fltk::z-shared</code>
<code>fluid</code>	<code>fluid</code>	<code>fltk::fluid</code>	n/a

For project developers used to the old (1.3.x) names the following table can assist to find the new option names. This table is ordered alphabetically by the old option name. Note that some option names did not change and some of the "old" names have been introduced in early 1.4.0 development.

Old Option Name (FLTK 1.3.x)	New Option Name (FLTK 1.4.x)
FLTK_BUILD_EXAMPLES	FLTK_BUILD_EXAMPLES
FLTK_BUILD_FLTK_OPTIONS	FLTK_BUILD_FLTK_OPTIONS
FLTK_BUILD_FLUID	FLTK_BUILD_FLUID
FLTK_BUILD_FORMS	FLTK_BUILD_FORMS
FLTK_BUILD_TEST	FLTK_BUILD_TEST
FLTK_MSVC_RUNTIME_DLL	FLTK_MSVC_RUNTIME_DLL
OPTION_ABI_VERSION	FLTK_ABI_VERSION
OPTION_ALLOW_GTK_PLUGIN	FLTK_USE_LIBDECOR_GTK
OPTION_APPLE_X11	FLTK_BACKEND_X11
OPTION_ARCHFLAGS	FLTK_ARCHFLAGS
OPTION_BUILD_HTML_DOCUMENTATION	FLTK_BUILD_HTML_DOCS
OPTION_BUILD_PDF_DOCUMENTATION	FLTK_BUILD_PDF_DOCS
OPTION_BUILD_SHARED_LIBS	FLTK_BUILD_SHARED_LIBS
OPTION_CAIRO	FLTK_OPTION_CAIRO_WINDOW
OPTION_CAIROEXT	FLTK_OPTION_CAIRO_EXT
OPTION_CREATE_LINKS	FLTK_INSTALL_LINKS
OPTION_FILESYSTEM_SUPPORT	FLTK_OPTION_FILESYSTEM_SUPPORT
OPTION_INCLUDE_DRIVER_DOCUMENTATION	FLTK_INCLUDE_DRIVER_DOCS
OPTION_INSTALL_HTML_DOCUMENTATION	FLTK_INSTALL_HTML_DOCS
OPTION_INSTALL_PDF_DOCUMENTATION	FLTK_INSTALL_PDF_DOCS
OPTION_LARGE_FILE	FLTK_OPTION_LARGE_FILE
OPTION_OPTIM	FLTK_OPTION_OPTIM
OPTION_PRINT_SUPPORT	FLTK_OPTION_PRINT_SUPPORT
OPTION_USE_CAIRO	FLTK_GRAPHICS_CAIRO
OPTION_USE_GDIPLUS	FLTK_GRAPHICS_GDIPLUS
OPTION_USE_GL	FLTK_BUILD_GL
OPTION_USE_KDIALOG	FLTK_USE_KDIALOG
OPTION_USE_PANGO	FLTK_USE_PANGO
OPTION_USE_POLL	FLTK_USE_POLL
OPTION_USE_STD	FLTK_OPTION_STD
OPTION_USE_SVG	FLTK_OPTION_SVG
OPTION_USE_SYSTEM_LIBDECOR	FLTK_USE_SYSTEM_LIBDECOR
OPTION_USE_SYSTEM_LIBJPEG	FLTK_USE_SYSTEM_LIBJPEG
OPTION_USE_SYSTEM_LIBPNG	FLTK_USE_SYSTEM_LIBPNG
OPTION_USE_SYSTEM_ZLIB	FLTK_USE_SYSTEM_ZLIB
OPTION_USE_THREADS	FLTK_USE_PTHREADS
OPTION_USE_WAYLAND	FLTK_BACKEND_WAYLAND
OPTION_USE_XCURSOR	FLTK_USE_XCURSOR
OPTION_USE_XFIXES	FLTK_USE_XFIXES
OPTION_USE_XFT	FLTK_USE_XFT
OPTION_USE_XINERAMA	FLTK_USE_XINERAMA
OPTION_USE_XRENDER	FLTK_USE_XRENDER
OPTION_WAYLAND_ONLY	FLTK_BACKEND_X11=OFF

1.19.8 New FL_HELVETICA Font on Windows

In FLTK releases before 1.4.2 the Windows font name used for the FLTK font 'FL_HELVETICA' and its variants was named 'Arial'. It turned out that this was a very old font used in early Windows versions and had been replaced with the newer font name 'Microsoft Sans Serif'. This font has many more UTF-8 capable glyphs (characters) and

is almost 100% compatible with the previously used font 'Arial'. It is supported by Microsoft since Windows XP, our oldest currently (partially) supported Windows platform.

This new font is used since FLTK 1.4.2.

We don't expect any issues with this much better font but if you really want strict backwards compatibility with FLTK 1.3.x or early 1.4.0/1.4.1 versions you can apply a tiny patch at the beginning of your program.

This code will load the old font 'Arial' on Windows:

```
#ifndef _WIN32
// reset Windows fonts to pre-1.4.2 state
Fl::set_font(FL_HELVETICA, "Arial");
Fl::set_font(FL_HELVETICA + 1, "BArial");
Fl::set_font(FL_HELVETICA + 2, "IArial");
Fl::set_font(FL_HELVETICA + 3, "PArial");
#endif
```

1.20 Software License

December 11, 2001

The FLTK library and included programs are provided under the terms of the GNU Library General Public License (LGPL) with the following exceptions:

1. Modifications to the FLTK configure script, config header file, and makefiles by themselves to support a specific platform do not constitute a modified or derivative work.

The authors do request that such modifications be contributed to the FLTK project - send all contributions through the "Software Trouble Report" on the following page: <https://www.fltk.org/bugs.php>

2. Widgets that are subclassed from FLTK widgets do not constitute a derivative work.

3. Static linking of applications and widgets to the FLTK library does not constitute a derivative work and does not require the author to provide source code for the application or widget, use the shared FLTK libraries, or link their applications or widgets against a user-supplied version of FLTK.

If you link the application or widget to a modified version of FLTK, then the changes to FLTK must be provided under the terms of the LGPL in sections 1, 2, and 4.

4. You do not have to provide a copy of the FLTK license with programs that are linked to the FLTK library, nor do you have to identify the FLTK license in your program or documentation as required by section 6 of the LGPL.

However, programs must still identify their use of FLTK. The following example statement can be included in user documentation to satisfy this requirement:

[program/widget] is based in part on the work of the FLTK project (<https://www.fltk.org>).

GNU LIBRARY GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.

59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.
[This is the first released version of the library GPL. It is numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given

copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy. For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE

COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

1.21 Example Source Code

The FLTK distribution contains over 60 sample applications written in, or ported to, FLTK. If the FLTK archive you received does not contain either an 'examples' or 'test' directory, you can download the complete FLTK distribution from <https://www.fltk.org/software.php>.

Most of the example programs were created while testing a group of widgets. They are not meant to be great achievements in clean C++ programming, but merely a test platform to verify the functionality of the FLTK library.

Note that extra example programs are also available in an additional 'examples' directory, but these are **NOT** built automatically when you build FLTK, unlike those in the 'test' directory shown below.

1.21.1 Example Applications: Overview

adjuster	animated	arc	ask	bitmap	blocks
boxtype	browser	button	buttons	cairo_test	checkers
clipboard	clock	colbrowser	color_chooser	cube	CubeView
cursor	curve	demo	device	doublebuffer	editor
fast_slow	file_chooser	FLUID	fonts	forms	fractals
fullscreen	gl_overlay	glpuzzle	hello	help_dialog	icon
iconize	image	inactive	input	input_choice	keyboard
label	line_style	list_visuals	mandelbrot	menubar	message
minimum	native-filechooser	navigation	offscreen	output	overlay
pack	pixmap	pixmap_browser	preferences	radio	resize
resizebox	rotated_text	scroll	shape	subwindow	sudoku
symbols	table	tabs	threads	tile	tiled_image
tree	twowin	unittests	utf8	valuators	windowfocus

1.21.1.1 adjuster

`adjuster` shows a nifty little widget for quickly setting values in a great range.

1.21.1.2 animated

`animated` shows a window with an animated square that shows drawing with transparency (alpha channel).

1.21.1.3 arc

The `arc` demo explains how to derive your own widget to generate some custom drawings. The sample drawings use the matrix based arc drawing for some fun effects.

1.21.1.4 `ask`

`ask` shows some of FLTK's standard dialog boxes. Click the correct answers or you may end up in a loop, or you may end up in a loop, or you... .

1.21.1.5 `bitmap`

This simple test shows the use of a single color bitmap as a label for a box widget. Bitmaps are stored in the X11 '.bmp' file format and can be part of the source code.

1.21.1.6 `blocks`

A wonderful and addictive game that shows the usage of FLTK timers, graphics, and how to implement sound on all platforms. `blocks` is also a good example for the Mac OS X specific bundle format.

1.21.1.7 `boxtype`

`boxtype` gives an overview of readily available boxes and frames in FLTK. More types can be added by the application programmer. When using themes, FLTK shuffles boxtypes around to give your program a new look.

1.21.1.8 `browser`

`browser` shows the capabilities of the [Fl_Browser](#) widget. Important features tested are loading of files, line formatting, and correct positioning of the browser data window.

1.21.1.9 `button`

The `button` test is a simple demo of push-buttons and callbacks.

1.21.1.10 `buttons`

`buttons` shows a sample of FLTK button types.

1.21.1.11 `cairo_test`

`cairo_test` shows a sample of drawing with Cairo in an [Fl_Cairo_Window](#). This program can only be built completely if FLTK was configured with Cairo support. It displays a message box if Cairo support was not included.

1.21.1.12 `checkers`

Written by Steve Poulsen in early 1979, `checkers` shows how to convert a VT100 text-terminal based program into a neat application with a graphical UI. Check out the code that drags the pieces, and how the pieces are drawn by layering. Then tell me how to beat the computer at Checkers.

1.21.1.13 `clipboard`

The `clipboard` demo can be used to view the contents of the system clipboard, either text or image contents. Currently an image is preferred if the clipboard contains both formats. Images can be stored as PNG files so screenshots can be stored on disk with this little FLTK demo program.

1.21.1.14 clock

The `clock` demo shows two analog clocks. The innards of the `FL_Clock` widget are pretty interesting, explaining the use of timeouts and matrix based drawing.

1.21.1.15 colbrowser

`colbrowser` runs only on X11 systems. It reads `/usr/lib/X11/rgb.txt` to show the color representation of every text entry in the file. This is beautiful, but only moderately useful unless your UI is written in *Motif*.

1.21.1.16 color_chooser

The `color_chooser` gives a short demo of FLTK's palette based color chooser and of the RGB based color wheel.

1.21.1.17 cube

The `cube` demo shows the speed of OpenGL. It also tests the ability to render two OpenGL buffers into a single window, and shows OpenGL text.

1.21.1.18 CubeView

`CubeView` shows how to create a UI containing OpenGL with FLUID.

1.21.1.19 cursor

The `cursor` demo shows all mouse cursor shapes that come standard with FLTK. The *fgcolor* and *bicolor* sliders work only on few systems (some version of Irix for example).

1.21.1.20 curve

`curve` draws a nice Bézier curve into a custom widget. The *points* option for splines is not supported on all platforms.

1.21.1.21 demo

This tool allows quick access to most programs in the `test` directory. The menu tree can be changed by editing `test/demo.menu`.

1.21.1.22 device

Exercises the `FL_Image_Surface`, `FL_Copy_Surface`, and `FL_Printer` classes to draw to an `FL_Image` object, copy graphical data to the clipboard, and for print support.

Note

The `clipboard.cxx` program of the 'test' directory is a clipboard watching application that continuously displays the textual or graphical content of the system clipboard (a.k.a pasteboard on macOS) exercising `FL::paste()`.

1.21.1.23 doublebuffer

The `doublebuffer` demo shows the difference between a single buffered window, which may flicker during a slow redraw, and a double buffered window, which never flickers, but uses twice the amount of RAM. Some modern OS's double buffer all windows automatically to allow transparency and shadows on the desktop. FLTK is smart enough to not tripple buffer a window in that case.

1.21.1.24 editor

FLTK has two very different text input widgets. `Fl_Input` and derived classes are rather light weight, however `Fl_Text_Editor` is a complete port of *nedit* (with permission). The `editor` test is almost a full application, showing custom syntax highlighting and dialog creation. See chapter [Designing a Simple Text Editor](#) for a tutorial about creating this program.

1.21.1.25 fast_slow

`fast_slow` shows how an application can use the `Fl_Widget::when()` setting to receive different kinds of callbacks.

1.21.1.26 file_chooser

The standard FLTK `file_chooser` is the result of many iterations, trying to find a middle ground between a complex browser and a fast light implementation.

1.21.1.27 fonts

`fonts` shows all available text fonts on the host system. If your machine still has some pixmap based fonts, the supported sizes will be shown in bold face. Only the first 256 fonts will be listed.

1.21.1.28 forms

`forms` is an XForms program with very few changes. Search for "fltk" to find all changes necessary to port to fltk. This demo shows the different boxtypes. Note that some boxtypes are not appropriate for some objects.

1.21.1.29 fractals

`fractals` shows how to mix OpenGL, Glut and FLTK code. FLTK supports a rather large subset of Glut, so that many Glut applications compile just fine.

1.21.1.30 fullscreen

This demo shows how to do many of the window manipulations that are popular for games. You can toggle the border on/off, switch between single- and double-buffered rendering, and take over the entire screen. More information in the source code.

1.21.1.31 gl_overlay

`gl_overlay` shows OpenGL overlay plane rendering. If no hardware overlay plane is available, FLTK will simulate it for you.

1.21.1.32 glpuzzle

The `glpuzzle` test shows how most Glut source code compiles easily under FLTK.

1.21.1.33 `hello`

`hello`: Hello, World. Need I say more? Well, maybe. This tiny demo shows how little is needed to get a functioning application running with FLTK. Quite impressive, I'd say.

1.21.1.34 `help_dialog`

`help_dialog` displays the built-in FLTK help browser. The [FL_Help_Dialog](#) understands a subset of html and renders various image formats. This widget makes it easy to provide help pages to the user without depending on the operating system's html browser.

1.21.1.35 `icon`

`icon` demonstrates how an application icon can be set from an image. This icon should be displayed in the window bar (label), in the task bar, and in the task switcher (Windows: Alt-Tab). This feature is platform specific, hence it is possible that you can't see the icon. On Unix/Linux (X11) this can even depend on the Window Manager (WM).

1.21.1.36 `iconize`

`iconize` demonstrates the effect of the window functions `hide()`, `iconize()`, and `show()`.

1.21.1.37 `image`

The `image` demo shows how an image can be created on the fly. This generated image contains an alpha (transparency) channel which lets previous renderings 'shine through', either via true transparency or by using screen door transparency (pixelation).

1.21.1.38 `inactive`

`inactive` tests the correct rendering of inactive widgets. To see the inactive version of images, you can check out the `pixmap` or `image` test.

1.21.1.39 `input`

This tool shows and tests different types of text input fields based on [FL_Input_](#). The `input` program also tests various settings of [FL_Input::when\(\)](#).

1.21.1.40 `input_choice`

`input_choice` tests the latest addition to FLTK1, a text input field with an attached pulldown menu. Windows users will recognize similarities to the 'ComboBox'. `input_choice` starts up in 'plastic' scheme, but the traditional scheme is also supported.

1.21.1.41 `keyboard`

FLTK unifies keyboard events for all platforms. The `keyboard` test can be used to check the return values of [FL::event_key\(\)](#) and [FL::event_text\(\)](#). It is also great to see the modifier buttons and the scroll wheel at work. Quit this application by closing the window. The ESC key will not work.

1.21.1.42 label

Every FLTK widget can have a label attached to it. The `label` demo shows alignment, clipping, and wrapping of text labels. Labels can contain symbols at the start and end of the text, like `@FLTK` or `@circle uh-huh @square`.

1.21.1.43 line_style

Advanced line drawing can be tested with `line_style`. Not all platforms support all line styles.

1.21.1.44 list_visuals

This little app finds all available pixel formats for the current X11 screen. But since you are now an FLTK user, you don't have to worry about any of this.

1.21.1.45 mandelbrot

`mandelbrot` shows two advanced topics in one test. It creates grayscale images on the fly, updating them via the `idle` callback system. This is one of the few occasions where the `idle` callback is very useful by giving all available processor time to the application without blocking the UI or other apps.

1.21.1.46 menubar

The `menubar` tests many aspects of FLTK's popup menu system. Among the features are radio buttons, menus taller than the screen, arbitrary sub menu depth, and global shortcuts.

1.21.1.47 message

`message` pops up a few of FLTK's standard message boxes.

1.21.1.48 minimum

The `minimum` test program verifies that the update regions are set correctly. In a real life application, the trail would be avoided by choosing a smaller label or by setting label clipping differently.

1.21.1.49 native-filechooser

The `native-filechooser` program invokes the platform specific file chooser, if available (see [FI_Native_File_Chooser](#) widget).

1.21.1.50 navigation

`navigation` demonstrates how the text cursor moves from text field to text field when using the arrow keys, tab, and shift-tab.

1.21.1.51 offscreen

`offscreen` shows how to draw into an offscreen image and display the offscreen image in the program window.

1.21.1.52 output

`output` shows the difference between the single line and multi line mode of the [FL_Output](#) widget. Fonts can be selected from the FLTK standard list of fonts.

1.21.1.53 overlay

The `overlay` test app shows how easy an FLTK window can be layered to display cursor and manipulator style elements. This example derives a new class from [FL_Overlay_Window](#) and provides a new function to draw custom overlays.

1.21.1.54 pack

The `pack` test program demonstrates the resizing and repositioning of children of the [FL_Pack](#) group. Putting an [FL_Pack](#) into an [FL_Scroll](#) is a useful way to create a browser for large sets of data.

1.21.1.55 pixmap

This simple test shows the use of a LUT based pixmap as a label for a box widget. Pixmapes are stored in the X11 '.xpm' file format and can be part of the source code. Pixmapes support one transparent color.

1.21.1.56 pixmap_browser

`pixmap_browser` tests the shared-image interface. When using the same image multiple times, [FL_Shared_Image](#) will keep it only once in memory.

1.21.1.57 preferences

I do have my `preferences` in the morning, but sometimes I just can't remember a thing. This is where the [FL_Preferences](#) come in handy. They remember any kind of data between program launches.

1.21.1.58 radio

The `radio` tool was created entirely with *FLUID*. It shows some of the available button types and tests radio button behavior.

1.21.1.59 resizebox

`resizebox` shows some possible ways of FLTK's automatic resize behavior.

1.21.1.60 rotated_text

`rotated_text` shows how text can be rotated, i.e. drawn in any given angle. This demo is device specific, for instance it works under X11 only if configured with Xft.

1.21.1.61 resize

The `resize` demo tests size and position functions with the given window manager.

1.21.1.62 scroll

`scroll` shows how to scroll an area of widgets, one of them being a slow custom drawing. [Fl_Scroll](#) uses clipping and smart window area copying to improve redraw speed. The buttons at the bottom of the window control decoration rendering and updates.

1.21.1.63 `shape`

`shape` is a very minimal demo that shows how to create your own OpenGL rendering widget. Now that you know that, go ahead and write that flight simulator you always dreamt of.

1.21.1.64 `subwindow`

The `subwindow` demo tests messaging and drawing between the main window and 'true' sub windows. A sub window is different to a group by resetting the FLTK coordinate system to 0, 0 in the top left corner. On Win32 and X11, subwindows have their own operating system specific handle.

1.21.1.65 `sudoku`

Another highly addictive game - don't play it, I warned you. The implementation shows how to create application icons, how to deal with OS specifics, and how to generate sound.

1.21.1.66 `symbols`

`symbols` are a speciality of FLTK. These little vector drawings can be integrated into labels. They scale and rotate, and with a little patience, you can define your own. The rotation number refers to 45 degree rotations if you were looking at a numeric keypad (2 is down, 6 is right, etc.).

1.21.1.67 `table`

The `table` demo shows the features of the [Fl_Table](#) widget.

1.21.1.68 `tabs`

The `tabs` tool was created with *FLUID*. It tests correct hiding and redisplaying of tabs, navigation across tabs, resize behavior, and no unneeded redrawing of invisible widgets.

The `tabs` application shows the [Fl_Tabs](#) widget on the left and the [Fl_Wizard](#) widget on the right side for direct comparison of these two panel management widgets.

1.21.1.69 `threads`

FLTK can be used in a multithreading environment. There are some limitations, mostly due to the underlying operating system. `threads` shows how to use [Fl::lock\(\)](#), [Fl::unlock\(\)](#), and [Fl::awake\(\)](#) in secondary threads to keep FLTK happy. Although locking works on all platforms, this demo is not available on every machine.

1.21.1.70 `tile`

The `tile` tool shows a nice way of using [Fl_Tile](#). To test correct resizing of subwindows, the widget for region 1 is created from an [Fl_Window](#) class.

1.21.1.71 tiled_image

The `tiled_image` demo uses a small image as the background for a window by repeating it over the full size of the widget. The window is resizable and shows how the image gets repeated.

1.21.1.72 tree

The `tree` demo shows the features of the `Fl_Tree` widget.

1.21.1.73 twowin

The `twowin` program tests focus transfer from one window to another window.

1.21.1.74 unittests

`unittests` exercises all of FLTK's drawing features (e.g., text, lines, circles, images), as well as scrollbars and schemes.

1.21.1.75 utf8

`utf8` shows all fonts available to the platform that runs it, and how each font draws each of the Unicode code points ranging between U+0020 and U+FFFF.

1.21.1.76 valuator

`valuator` shows all of FLTK's nifty widgets to change numeric values.

1.21.1.77 windowfocus

`windowfocus` shows a very special case when a new window is shown while the focus stays in the original window.

1.21.1.78 FLUID

FLUID is not only a big test program, but also a very useful visual UI designer. Many parts of FLUID were created using FLUID. Check out the FLUID User Manual and the tutorials that come with it at <https://www.fltk.org/documentation.php>.

1.21.2 Example Applications: Images

This chapter contains a few selected images of the test and example applications listed above. It is not meant to be complete or a full reference. The reason some images are included here is to show how the display **should** look when running the example programs.

1.21.2.1 cairo_test

The `cairo_test` demo program shows three shiny buttons drawn with Cairo in an `Fl_Cairo_Window`.



Figure 1.30 Buttons drawn with Cairo

1.21.2.2 icon

The `icon` program lets you set the program icon from an image (here an `Fl_RGB_Image`).

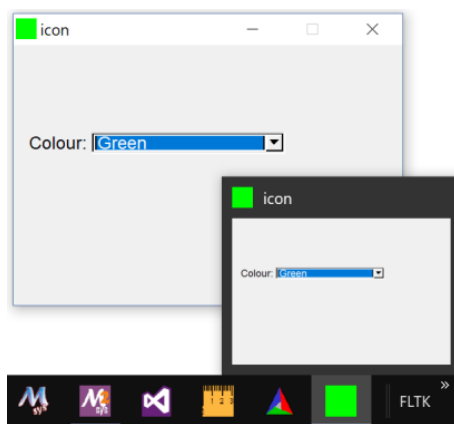


Figure 1.31 Green icon (Windows 10)

1.21.2.3 unittests

Select "drawing images" in the browser at the left side to see the image drawing example:

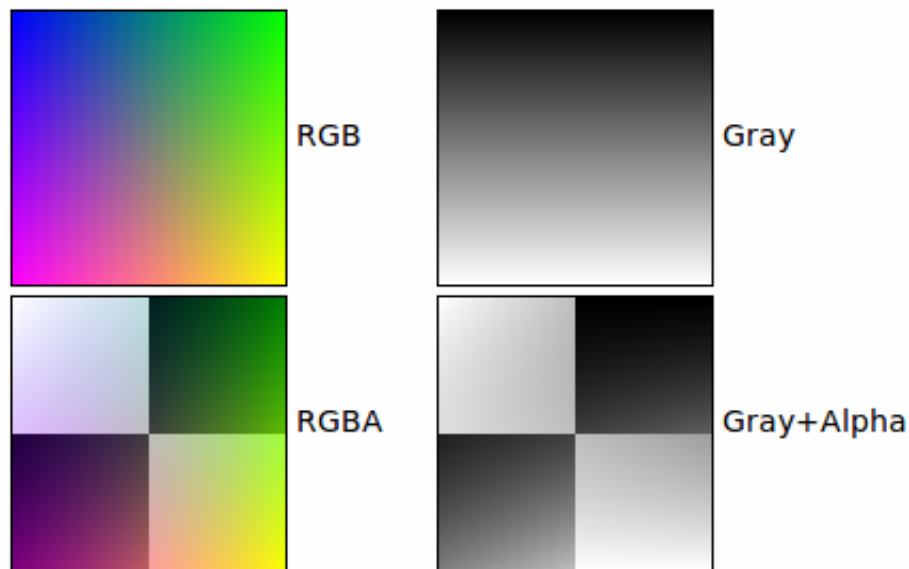


Figure 1.32 Image Drawing

1.22 FAQ (Frequently Asked Questions)

A list of frequently asked questions about FLTK.

This appendix describes various frequently asked questions regarding FLTK.

- [Where do I start learning FLTK?](#)
- [How do I make a box with text?](#)
- [Can I use FLTK to make closed-source commercial applications?](#)
- [Hitting the 'Escape' key closes windows - how do I prevent this?](#)

1.22.1 Where do I start learning FLTK?

It is assumed you know C++, which is the language all FLTK programs are written in, including FLTK itself.

If you like reading manuals to work your way into things, a good start is the FLTK documentation's [Introduction to FLTK](#). Under the [FLTK Basics](#) section there's an example 'hello world' program that includes a line-by-line description.

If you like looking at simple code first to pique your interest, and then read up from there, start with the example programs in the test/ and examples/ directory that is included with the source code. A good place to start is the 'hello world' program in test/hello.cxx. Also do a google search for "FLTK example programs". "Erco's Cheat Page" is one that shows many simple examples of how to do specific things.

If you like to run example programs and look for ones that are like yours and then read them, download and build FLTK from the source, then run the test/demo program. Also, go into the 'examples/' directory and run 'make', then run some of those programs.

If you prefer watching TV to reading books and code, google search for "FLTK video tutorials" which has some introductory examples of how to write FLTK programs in C++ and build them.

1.22.2 How do I make a box with text?

The 'hello world' program shows how to make a box with text. All widgets have labels, so picking a simple widget like [Fl_Box](#) and setting its label() and using align() to align the label and labelfont() to set the font, and labelsize() to set the size, you can get text just how you want.

Labels are not selectable though; if you want selectable text, you can use [Fl_Output](#) or [Fl_Multiline_Output](#) for simple text that doesn't include scrollbars. For more complex text that might want scrollbars and multiple colors/fonts, use either [Fl_Text_Display](#) which handles plain text, or [Fl_Help_View](#) which handles simple HTML formatted text.

1.22.3 Can I use FLTK to make closed-source commercial applications?

Yes. The FLTK [Software License](#) is standard LGPL, but also includes a special clause ("exception") to allow for static linking. Specifically:

[from the top of the FLTK LGPL License section on exceptions]

3. Static linking of applications and widgets to the FLTK library does not constitute a derivative work and does not require the author to provide source code for the application or widget, use the shared FLTK libraries, or link their applications or widgets against a user-supplied version of FLTK.

If you link the application or widget to a modified version of FLTK, then the changes to FLTK must be provided under the terms of the LGPL in sections 1, 2, and 4.

4. You do not have to provide a copy of the FLTK license with programs that are linked to the FLTK library, nor do you have to identify the FLTK license in your program or documentation as required by section 6 of the LGPL.

However, programs must still identify their use of FLTK. The following example statement can be included in user documentation to satisfy this requirement:

[program/widget] is based in part on the work of the
FLTK project (<https://www.fltk.org>).

1.22.4 Hitting the 'Escape' key closes windows - how do I prevent this?

[From FLTK article #378]

1. FLTK has a "global event handler" that makes Escape try to close the window, the same as clicking the close box. To disable this everywhere you can install your own that pretends it wants the escape key and thus stops the default one from seeing it (this may not be what you want, see below about the callbacks):

```
static int my_handler(int event) {
    if (event == FL_SHORTCUT) return 1; // eat all shortcut keys
    return 0;
}
...in main():
    Fl::add_handler(my_handler);
...
```

1. Attempts to close a window (both clicking the close box or typing Escape) call that window's callback. The default version of the callback does `hide()`. To make the window not close or otherwise do something different you replace the callback. To make the main window exit the program:

```
void my_callback(Fl_Widget*, void*) {
    exit(0);
}
...
main_window->callback(my_callback);
...
```

If you don't want Escape to close the main window and exit you can check for and ignore it. This is better than replacing the global handler because Escape will still close pop-up windows:

```
void my_callback(Fl_Widget*, void*) {
    if (Fl::event() == FL_SHORTCUT && Fl::event_key() == FL_Escape)
        return; // ignore Escape
    exit(0);
}
```

The reason for calling a window callback can also be found using using the `Fl::callback_reason()` method:

```
void my_callback(Fl_Widget*, void*) {
    if (Fl::callback_reason() == FL_REASON_CANCELLED)
        return; // ignore that the user pressed the Escape key
    if (Fl::callback_reason() == FL_REASON_CLOSED)
```

```
    save_and_exit(); // user clicked the Close button in the window decoration
    exit(0); // fallback for other callback reasons
}
```

It is very common to ask for confirmation before exiting, this can be done with:

```
void my_callback(Fl_Widget*, void*) {
    if (fl_choice("Are you sure you want to quit?",
                  "continue", "quit", NULL))
        exit(0);
}
```

Chapter 2

FI_Terminal Technical Documentation

This chapter covers the vt100/xterm style "escape codes" used by [FI_Terminal](#) for cursor positioning, text colors, and other display screen control features such as full or partial screen clearing, up/down scrolling, character insert/delete, etc.

2.1 The Escape Codes FI_Terminal Supports

These are the escape codes [FI_Terminal](#) actually supports, and is not the 'complete' list that e.g. xterm supports. Most of the important stuff has been implemented, but esoteric features (such as scroll regions) has not. Features will be added as the widget matures.

```
-----
--- The CSI (Control Sequence Introducer, or "ESC[") ---
-----

ESC[#@ - (ICH) Insert blank Chars (default=1)
ESC[#A - (CUU) Cursor Up, no scroll/wrap
ESC[#B - (CUD) Cursor Down, no scroll/wrap
ESC[#C - (CUF) Cursor Forward, no wrap
ESC[#D - (CUB) Cursor Back, no wrap
ESC[#E - (CNL) Cursor Next Line (crlf) xterm, !gnome
ESC[#F - (CPL) Cursor Preceding Line: move to sol and up # lines
ESC[#G - (CHA) Cursor Horizontal Absolute positioning

    ESC[G - move to column 1 (start of line, sol)
    ESC[#G - move to column #
ESC[#H - (CUP) Cursor Position (#'s are 1 based)

    ESC[H - go to row #1
    ESC[#H - go to (row #) (default=1)
    ESC[#;#H - go to (row# ; col#)
ESC[#I - (CHT) Cursor Horizontal Tab: tab forward

    ESC[#I - tab # times (default 1)
ESC[#J - (ED) Erase in Display

    ESC[0J - clear to end of display (default)
    ESC[1J - clear to start of display
    ESC[2J - clear all lines
    ESC[3J - clear screen history
ESC[#K - (EL) Erase in line

    ESC[0K - clear to end of line (default)
    ESC[1K - clear to start of line
    ESC[2K - clear current line
ESC[#L - (IL) Insert # Lines (default=1)
ESC[#M - (DL) Delete # Lines (default=1)
ESC[#P - (DCH) Delete # Chars (default=1)
ESC[#S - (SU) Scroll Up # lines (default=1)
ESC[#T - (SD) Scroll Down # lines (default=1)
ESC[#X - (ECH) Erase Characters (default=1)
ESC[#Z - (CBT) Cursor Backwards Tab

    ESC[#Z - backwards tab # times (default=1)
ESC[#a - (HPR) move cursor relative [columns] (default=[row,col+1]) (NOT IMPLEMENTED)
ESC[#b - (REP) repeat prev graphics char # times (NOT IMPLEMENTED)
ESC[#d - (VPA) Line Position Absolute [row] (NOT IMPLEMENTED)
ESC[#e - (LPA) Line Position Relative [row] (NOT IMPLEMENTED)
ESC[#f - (CUP) cursor position (#'s 1 based), same as ESC[H
ESC[#g - (TBC) Tabulation Clear

    ESC[0g - Clear tabstop at cursor
    ESC[3g - Clear all tabstops
```

ESC[#m - (SGR) Set Graphic Rendition

*** Attribute Enable ***

```
ESC[0m - reset: normal attribs/default fg/bg color (VT100)
ESC[1m - bold (VT100)
ESC[2m - dim
ESC[3m - italic
ESC[4m - underline (VT100)
ESC[5m - blink (NOT IMPLEMENTED) (VT100)
ESC[6m - (unused)
ESC[7m - inverse (VT100)
ESC[8m - (unused)
ESC[9m - strikeout
ESC[21m - doubly underline (Currently this just does single underline)
```

*** Attribute Disable ***

```
ESC[22m - disable bold/dim
ESC[23m - disable italic
ESC[24m - disable underline
ESC[25m - disable blink (NOT IMPLEMENTED)
ESC[26m - (unused)
ESC[27m - disable inverse
ESC[28m - disable hidden
ESC[29m - disable strikeout
```

*** Foreground Text "8 Color" ***

```
ESC[30m - fg Black
ESC[31m - fg Red
ESC[32m - fg Green
ESC[33m - fg Yellow
ESC[34m - fg Blue
ESC[35m - fg Magenta
ESC[36m - fg Cyan
ESC[37m - fg White
ESC[39m - fg default
```

*** Background Text "8 Color" ***

```
ESC[40m - bg Black
ESC[41m - bg Red
ESC[42m - bg Green
ESC[43m - bg Yellow
ESC[44m - bg Blue
ESC[45m - bg Magenta
ESC[46m - bg Cyan
ESC[47m - bg White
ESC[49m - bg default
```

*** Special RGB Color ***

```
ESC [ 38 ; Red ; Grn ; Blue m - where Red,Grn,Blu are decimal (0-255)
ESC[s - save cursor pos (ansi.sys+xterm+gnome, but NOT vt100)
ESC[u - rest cursor pos (ansi.sys+xterm+gnome, but NOT vt100)
ESC[>#q - (DECSCA) Set Cursor style (block/line/blink..) (NOT IMPLEMENTED)
ESC[#;#r - (DECSTBM) Set scroll Region top;bot (NOT IMPLEMENTED)
ESC[#..$t - (DECRARA) (NOT IMPLEMENTED)
-----
--- C1 Control Codes ---
-----
<ESC>c - (RIS) Reset term to Initial State
<ESC>D - (IND) Index: move cursor down a line, scroll if at bottom
<ESC>E - (NEL) Next Line: basically do a crlf, scroll if at bottom
<ESC>H - (HTS) Horizontal Tab Set: set a tabstop
<ESC>M - (RI) Reverse Index (up w/scroll)
```

NOTE: Acronyms in parens are Digital Equipment Corporation's names these VT features.

2.2 Useful Terminal Escape Code Documentation

Useful links for reference:

- <https://vt100.net/docs/vt100-ug/chapter3.html>
- <https://www.xfree86.org/current/ctlseqs.html>
- <https://www.x.org/docs/xterm/ctlseqs.pdf>
- <https://gist.github.com/justinmk/a5102f9a0c1810437885a04a07ef0a91> <- al-
phabetic!

- <https://invisible-island.net/xterm/ctlseqs/ctlseqs.html>

2.3 FI_Terminal Design Document

When I started this project, I identified the key concepts needed to implement `FI_Terminal`:

- Draw and manage multiline Unicode text in FLTK
- Allow per-character colors and attributes
- Efficient screen buffer to handle "scrollback history"
- Efficient scrolling with vertical scrollbar for even large screen history
- Mouse selection for copy/paste
- Escape code management to implement VT100 style / ANSI escape codes.

A class was created for each character, since characters can be either ASCII or Utf8 encoded byte sequences. This class is called `Utf8Char`, and handles the character, its fg and bg color, and any attributes like dim, bold, italic, etc. For managing the screen, after various experiments, I decided a ring buffer was the best way to manage things, the ring split in two:

- 'screen history' which is where lines scrolled off the top are saved
- 'display screen' displayed to the user at all times, and where the cursor lives

Scrolling the display, either by scrollbar or by new text causing the display to scroll up one line, would simply change an 'offset' index# of where in the ring buffer the top of the screen is, automatically moving the top line into the history, all without moving memory around.

In fact the only time screen memory is moved around is during these infrequent operations:

- during scrolling "down"
- character insert/delete operations within a line
- changing the display size
- changing the history size

So a class "RingBuffer" is defined to manage the ring, and accessing its various parts, either as the entire entity ring, just the history, or just the display.

These three concepts, "ring", "history" and "display" are given abbreviated names in the RingBuffer class's API:

NOTE: Abbreviations "hist" and "disp"

"history" may be abbreviated as "hist", and "display" as "disp" in both this text and the source code. 4 character names are used so they line up cleanly in the source, e.g.

```
ring_rows()    ring_cols()
hist_rows()    hist_cols()
disp_rows()    disp_cols()

4 characters
```

These concepts were able to fit into C++ classes:

Utf8Char

Each character on the screen is a "Utf8Char" which can manage the UTF-8 encoding of any character as one or more bytes. Also in that class is a byte for an attribute (underline, bold, etc), and two integers for fg/bg color.

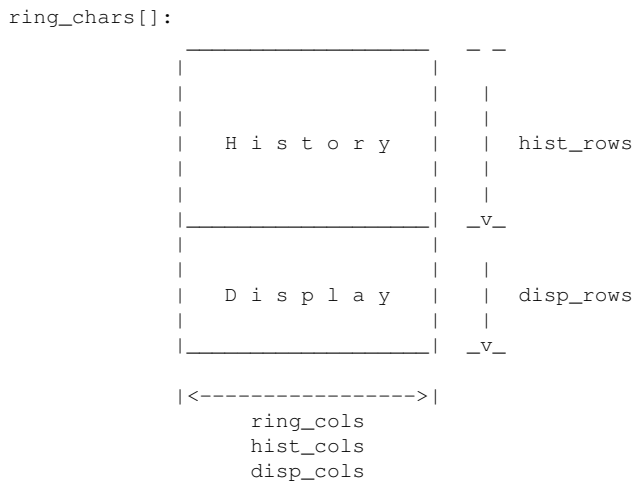
RingBuffer

The RingBuffer class keeps track of the buffer itself, a single array of Utf8Chars called "ring_chars" whose width is ring_cols() and whose height is ring_rows().

The "top" part of the ring is the history, whose width is hist_cols() and whose height is hist_rows(). hist_use_rows() is used to define what part of the history is currently in use.

The "bottom" part of the ring is the display, whose width is disp_cols() and whose height is disp_rows().

An index number called "offset" points to where in the ring buffer the top of the ring currently is. This index changes each time the screen is scrolled, and affects both where the top of the display is, and where the top of the history is. The memory layout of the Utf8Char character array is:



So it's basically a single continuous array of Utf8Char instances where any character can generally be accessed by index# using the formula:

```
ring_chars[ (row*ring_cols)+col ]
```

..where 'row' is the desired row, 'col' is the desired column, and 'ring_cols' is how many columns "wide" the buffer is. The "offset" index affects that formula as an extra row offset, and the resulting index is then clamped within the range of the ring buffer using modulus.

Methods are used to allow direct access to the characters in the buffer that automatically handle the offset and modulus formulas, namely:

```

u8c_ring_row(row,col)  // access the entire ring by row/col
u8c_hist_row(row,col)  // access just the history buffer
u8c_disp_row(row,col)  // access just the display buffer

```

A key concept is the use of the simple 'offset' index integer to allow the starting point of the history and display to be moved around to implement 'text scrolling', such as when crlf at the screen bottom causes a 'scroll up'.

This is simply an "index offset" integer applied to the hist and disp indexes when drawing the display. So after scrolling two lines up, the offset is just increased by 2, redefining where the top of the history and display are, e.g.

```

Offset is 0:      2      Offset now 2:
                  D i s p l a y
                  >
H i s t o r y      H i s t o r y
                  2
                  >
D i s p l a y      D i s p l a y

```

The diagram shows the effect of increasing the offset by 2. On the left, 'Offset is 0:', the 'History' and 'Display' sections are aligned. On the right, 'Offset now 2:', the 'History' section has shifted up by two lines, and the 'Display' section has also shifted up by two lines, leaving a gap at the bottom. The '2' indicates the offset value.

This 'offset' trivially implements "text scrolling", avoiding having to physically move memory around. Just the 'offset' changes, the text remains where it is in memory.

This also makes it appear the top line in the display is 'scrolled up' into the bottom of the scrollbar 'history'. If the offset exceeds the size of the ring buffer, it simply wraps around back to the beginning of the buffer with a modulo.

Indexes into the display and history are also modulo their respective rows, e.g.

```
act_ring_index = (hist_rows + disp_row + offset - scrollbar_pos) % ring_rows;
```

This way indexes for ranges can run beyond the bottom of the ring, and automatically wrap around the ring, e.g.

```
> 2
  3   D i s p l a y
  4
    <-- offset points here

disp
index      H i s t o r y
wraps

      0   D i s p l a y
      1   <- ring_rows points to end of ring
      2 :
      3 :
disp_row(5) -> 4 :.....:
```

The dotted lines show where the display would be if not for the fact it extends beyond the bottom of the ring buffer (due to the current offset), and therefore wraps up to the top of the ring.

So to find a particular row in the display, in this case a 5 line display whose lines lie between 0 and 4, some simple math calculates the row position into the ring:

```
act_ring_index = (histrows      // the display exists AFTER the history, so offset the hist_rows
                  + offset      // include the scroll 'offset'
                  + disp_row    // add the desired row relative to the top of the display (0..disp_rows)
                  ) % ring_rows; // make sure the resulting index is within the ring buffer (0..ring_rows)
```

An additional bit of math makes sure if a negative result occurs, that negative value works relative to the end of the ring, e.g.

```
if (act_ring_index < 0) act_ring_index = ring_rows + act_ring_index;
```

This guarantees the `act_ring_index` is within the ring buffer's address space, with all offsets applied.

The math that implements this can be found in the `u8c_xxxx_row()` methods, where "xxxx" is one of the concept regions "ring", "hist" or "disp":

```
Utf8Char *u8c;
u8c = u8c_ring_row(rrow);    // address within ring, rrow can be 0..(ring_rows-1)
u8c = u8c_hist_row(hrow);    // address within hist, hrow can be 0..(hist_rows-1)
u8c = u8c_disp_row(drow);    // address within disp, drow can be 0..(disp_rows-1)
```

The small bit of math is only involved whenever a new row address is needed, so in a display that's 80x25, to walk all the characters in the screen, the math above would only be called 25 times, once for each row, and each column in the row is just a simple integer offset:

```
for ( int row=0; row<disp_rows(); row++ ) {    // walk rows: disp_rows = 25
    Utf8Char *u8c = u8c_disp_row(row);          // get first char in display 'row'
    for ( int col=0; col<disp_cols(); col++ ) { // walk cols: disp_cols = 80
        u8c[col].do_something();                // work with the char at row/col
    }
}
```

So to recap, the concepts here are:

- The ring buffer itself, a linear array that is conceptually split into a 2 dimensional array of rows and columns whose height and width are:

```
ring_rows -- how many rows in the entire ring buffer
ring_cols -- how many columns in the ring buffer
nchars    -- total chars in ring, e.g. (ring_rows * ring_cols)
```

- The "history" within the ring. For simplicity this is thought of as starting relative to the top of the ring buffer, occupying ring buffer rows:

```
0 .. hist_rows()-1
```

- The "display", or "disp", within the ring, just after the "history". It occupies the ring buffer rows:

```
hist_rows() .. hist_rows()+disp_rows()-1
```

..or similarly:

```
(hist_rows)..(ring_rows-1)
```

The following convenience methods provide access to the start and end indexes within the ring buffer for each entity:

Entire ring ring_srow() – start row index of the ring buffer (always 0) ring_erow() – end row index of the ring buffer
 "history" part of ring hist_srow() – start row index of the screen history hist_erow() – end row index of the screen history

"display" part of ring disp_srow() – start row index of the display disp_erow() – end row index of the display

The values returned by these are as described above. For the hist_xxx() and disp_xxx() methods the 'offset' included into the formula. (For this reason hist_srow() won't always be zero the way ring_srow() is, due to the 'offset')

The values returned by these methods can all be passed to the u8c_ring_row() function to access the actual character buffer's contents.

- An "offset" used to move the "history" and "display" around within the ring buffer to implement the "text scrolling" concept. The offset is applied when new characters are added to the buffer, and during drawing to find where the display actually is within the ring.
- The "scrollbar", which only is used when redrawing the screen the user sees, and is simply an additional offset to all the above, where a scrollbar value of zero (the scrollbar tab at the bottom) shows the display rows, and as the scrollbar values increase as the user moves the scrollbar tab upwards, +1 per line, this is subtracted from the normal starting index to let the user work their way backwards into the scrollbar history. Again, negative numbers wrap around within the ring buffer automatically.

The ring buffer allows new content to simply be appended to the ring buffer, and the index# for the start of the display and start of scrollbar history are simply incremented. So the next time the display is "drawn", it starts at a different position in the ring.

This makes scrolling content at high speed trivial, without memory moves. It also makes the concept of "scrolling" with the scrollbar simple as well, simply being an extra index offset applied during drawing.

Mouse Selection

Dragging the mouse across the screen should highlight the text, allowing the user to extend the selection either beyond or before the point started. Extending the drag to the top of the screen should automatically 'scroll up' to select more lines in the scrollbar history, or below the bottom to do the opposite.

The mouse selection is implemented as a class to keep track of the start/end row/col positions of the selection, and other details such as a flag indicating if a selection has been made, what color the fg/bg text should appear when text is selected, and methods that allow setting and extending the selection, clearing the selection, and "scrolling" the selection, to ensure the row/col indexes adjust correctly to track when the screen or scrollbar is scrolled.

Redraw Timer

Knowing when to redraw is tricky with a terminal, because sometimes high volumes of input will come in asynchronously, so in that case we need to determine when to redraw the screen to show the new content; too quickly will cause the screen to spend more time redrawing itself, preventing new input from being added. Too slowly, the user won't see new information appear in a timely manner.

To solve this, a rate timer is used to prevent too many redraws:

- When new data comes in, a 1/10 sec timer is started and a modify flag is set.
- redraw() is NOT called yet, allowing more data to continue to arrive quickly

- When the 1/10th second timer fires, the callback checks the modify flag:
 - if set, calls `redraw()`, resets the modify to 0, and calls `FI::repeat_timeout()` to repeat the callback in another 1/10th sec.
 - if clear, no new data came in, so DISABLE the timer, done.

In this way, redraws don't happen more than 10x per second, and `redraw()` is called only when there's new content to see.

The redraw rate can be set by the user application using the `FI_Terminal::redraw_rate()`, 0.10 being the default. Some terminal operations necessarily call `redraw()` directly, such as interactive mouse selection, or during user scrolling the terminal's scrollbar, where it's important there's no delay in what the user sees while interacting directly with the widget.

Chapter 3

Development of the FLTK library

- [The Wayland backend for its developer](#)
- [Developer info for bundled libs](#)
- [Developer Information](#)

3.1 The Wayland backend for its developer

This chapter describes how the Wayland backend of FLTK works from a developer's viewpoint.

3.1.1 Introduction to Wayland

Wayland usage involves communication via a Unix domain socket between a client application and another process called the Wayland compositor which creates, moves, resizes and draws windows on the display. Diverse Wayland compositors exist. They can follow rather diverse logics. For example, FreeBSD offers Sway which is a tiling compositor where the display is always entirely filled with whatever resizable windows are mapped at any given time. Compositors follow either the client-side decoration (CSD) rule where client apps draw window titlebars, or the server-side decoration (SSD) rule where the compositor draws titlebars. FLTK supports both CSD and SSD compositors. It uses a library called `libdecor` charged of determining whether a CSD or a SSD compositor is active, and of drawing titlebars in the first case.

Wayland is divided in various protocols that a given compositor may or may not support, although they all support the `core` protocol. Each protocol adds functionality not available in the core protocol. [Wayland Explorer](#) lists all protocols. The core protocol allows a client app to discover what protocols the connected compositor supports. Protocols can be stable, which means they have a defined API that will not change but can be expanded, or unstable. For example, mapping a window on a display is not done by the core protocol but by the `xdg_shell` protocol which is stable. The names of symbols used by unstable protocols always begin with letter 'z'. For example, FLTK uses unstable protocol `Text input` to support CJK input methods; its symbol names begin with `zwp_text_input_v3`.

Wayland makes intensive use of the *listener* mechanism. A listener is a small array of pointers to FLTK-defined callback functions associated to a Wayland-defined object; Wayland calls these functions when defined events occur (more at [Listeners](#) below).

Wayland differs noticeably from X11 in that rendering is left to clients: Wayland provides no drawing API. Instead, Wayland provides objects of type `struct wl_buffer` which encapsulate a memory array of pixel values shared between the client and the compositor. The client app is expected to draw to that memory buffer with whatever means it chooses, and to instruct the compositor to map those pixels to the display when the drawing is complete. The Wayland platform of FLTK draws with the Cairo library to `Fl_Window`'s and `Fl_Image_Surface`'s, and with OpenGL to `Fl_Gl_Window`'s.

Wayland differs also from X11 in that the position of a window in the display is completely hidden to the client app. This prevents function `Fl_Window::position()` from having any effect on a top-level window. Wayland also prevents a client app from knowing whether a window is minimized: `Fl_Window::show()` has no effect on a minimized window. Subwindows can be positioned as usual relatively to their parent window. Wayland allows to create popup windows positioned relatively to a previously mapped other window. This allows FLTK to position adequately menu and tooltip windows (see [Menu windows and other popups](#)). FLTK uses also popups for the small,

yellow windows that display the new scale factor value when it's changed: these are created as short-lived popups centered above `Fl::first_window()`.

Wayland uses a trick of its own to handle lists of linked records. It defines type `struct wl_list` and a few macros (`wl_list_init()`, `wl_list_for_each()`, `wl_list_insert()`, `wl_list_for_each_safe()`, `wl_list_remove()`) to manage linked lists. Records put in these lists must contain a member variable of type `struct wl_list` used to link records together and often named 'link'. Access to such a list is possible memorizing a value of type `struct wl_list` computed by macro `wl_list_init()`. Macro `wl_list_for_each(arg1, arg2, arg3)` allows to run through all list elements with:

- `arg1` is a pointer variable of the type of elements of the linked list;
- `arg2` is the address of a variable of type `struct wl_list` identifying the targeted list;
- `arg3` is the name of the member variable of these elements used to link them together.

For example, `wl_list_for_each()` can be used as follows to scan the linked list of all displays of the system (see `Fl_Wayland_Screen_Driver::output`):

```
Fl_Wayland_Screen_Driver::output *output;
Fl_Wayland_Screen_Driver *scr_driver = (Fl_Wayland_Screen_Driver*)Fl::screen_driver();
wl_list_for_each(output, &(scr_driver->outputs), link) {
    // ... work with output, an item of the linked list of all displays in the system ...
}
```

Overall, and ignoring for now OpenGL usage, FLTK interacts with Wayland as follows :

- When opening the display: FLTK calls `Fl::add_fd()` in `FL_READ` mode to associate a callback function called `wayland_socket_callback` to the socket connecting the client and the compositor.
- Client to compositor: FLTK calls C functions of the `libwayland-client.so`, `libwayland-cursor.so` and `libxkbcommon.so` shared libraries and of the `libdecor` library. These send suitable messages to the compositor writing to the socket. The names of these functions begin with `wl_`, `xkb_` or `libdecor_`.
- Compositor to client: the callback function `wayland_socket_callback` runs when there are data to read in the socket; it calls `wl_display_dispatch()` which interprets the read data and calls corresponding listeners.

The core protocol defines also a number of mostly opaque structures whose names begin with `wl_`. The names of symbols and types defined by the other protocols FLTK uses begin with `xdg_`, `zwp_text_input_v3`, `zxdg_toplevel_decoration`, `gtk_shell1` and `gtk_surface1`. FLTK defines a few structures holding Wayland-related data. The names of FLTK-defined structures don't begin with `wl_`. For example, `struct wld_window` (see `wld_window`) is used to store all Wayland-specific data associated to a mapped `Fl_Window`.

3.1.2 Building libfltk as a Wayland client

Classes `Fl_Wayland_Window_Driver`, `Fl_Wayland_Screen_Driver`, `Fl_Wayland_Graphics_Driver`, `Fl_Wayland_Copy_Surface_Driver`, `Fl_Wayland_Image_Surface_Driver` and `Fl_Wayland_Gl_Window_Driver` and file `fl_wayland_platform_init.cxx` contain all the Wayland-specific code of the FLTK library. This code is located at `src/drivers/Wayland/` in the FLTK source tree. A single C++ source file generally contains all the code of a given class. The code related to copy, paste and drag-and-drop operations, however, is gathered in file `fl_wayland_clipboard_dnd.cxx` and contains a few member functions of class `Fl_Wayland_Screen_Driver`. Furthermore, class `Fl_UNIX_System_Driver` is used by both the Wayland and the X11 FLTK platforms. File `FL/fl_config.h` defines preprocessor variables `FLTK_USE_WAYLAND` and `FLTK_USE_CAIRO`.

The public C API to Wayland, xkb, EGL and libdecor libraries are obtained with

```
#include <wayland-client.h>
#include <wayland-cursor.h>
#include <xkbcommon/xkbcommon.h>
#include <xkbcommon/xkbcommon-compose.h>
#include <linux/input.h> // for BTN_LEFT, BTN_RIGHT, BTN_MIDDLE
#include "../../libdecor/src/libdecor.h"
#include "../../libdecor/src/libdecor-plugin.h"
#ifdef HAVE_GL
#   include <wayland-egl.h>
#   include <EGL/egl.h>
#endif // HAVE_GL
```

as necessary.

File `README.Wayland.txt` details what software packages are needed on Debian-based, Fedora and FreeBSD systems for FLTK to use Wayland. Wayland protocols are packaged as XML files accompanied by a utility program, `wayland-scanner`, able to generate a header file and a necessary glue C source file from a

given XML file. For example, for FLTK to use the `xdg-shell` protocol, these commands are run at build time to generate a `.c` file (`xdg-shell-protocol.c`) that will be compiled into `libfltk` and a header file (`xdg-shell-client-protocol.h`) that the FLTK code will include:

```
PROTOCOLS=`pkg-config --variable=pkgdatadir wayland-protocols`
wayland-scanner private-code $PROTOCOLS/stable/xdg-shell/xdg-shell.xml xdg-shell-protocol.c
wayland-scanner client-header $PROTOCOLS/stable/xdg-shell/xdg-shell.xml xdg-shell-client-protocol.h
```

Similar operations are performed for FLTK to use protocols `xdg-decoration`, `Text input` and `GTK Shell`.

3.1.3 The hybrid Wayland/X11 platform

The Wayland platform of FLTK is normally a two-legged hybrid able to use either Wayland or X11 and to choose between these possibilities at run-time, without any change to the client application. The Wayland/X11 hybrid is essentially a version of the FLTK library containing both all Wayland-specific and all X11-specific code. That's reflected in file `FL/fl_config.h` which defines both `FLTK_USE_WAYLAND` and `FLTK_USE_X11`. This creates the constraint that Wayland and X11 cannot use the same type name for different purposes or the same symbol name. That is why function `fl_xid(const Fl_Window*)` is deprecated in FLTK 1.4 and replaced by `fl_wl_xid()` for Wayland and `fl_x11_xid()` for X11. Also, global variable `Window fl_window` is not used by the Wayland platform which instead uses static struct `wld_window *Fl_Wayland_Window_Driver:: wld_window`. The FLTK library contains also a short source file, `fl_wayland_platform_init.cxx`, that determines, at startup time, whether the app will run as a Wayland or as an X11 client. Function `attempt_wayland()` therein performs this choice as follows :

- if the app defines a global bool variable called `fl_disable_wayland` and this variable is true, the X11 leg is chosen;
- if environment variable `FLTK_BACKEND` is defined to string "wayland", the Wayland leg is chosen;
- if environment variable `FLTK_BACKEND` is defined to string "x11", the X11 leg is chosen;
- otherwise, a connection to a Wayland compositor is attempted; if it's successful, the Wayland leg is chosen; if it's not, the X11 leg is chosen.

The first condition listed above is meant to facilitate transition to FLTK 1.4 of source code written for FLTK 1.3 and containing X11-specific code : it's enough to put

```
FL_EXPORT bool fl_disable_wayland = true;
```

anywhere in the source code, for the app to run with 1.4, using the x11 leg of the hybrid platform, without any other change in the source code nor to the application's environment.

Function `attempt_wayland()` must be called before the very first platform-dependent operation FLTK performs so that operation is done the Wayland or the X11 way, as appropriate. That's why 4 locations of the FLTK source code call `attempt_wayland()`: `Fl_Graphics_Driver::newMainGraphicsDriver()`, `Fl_Screen_Driver::newScreenDriver()`, `Fl_Window_Driver::newWindowDriver(Fl_Window*)`, and `Fl_Image_Surface_Driver::newImageSurfaceDriver()`.

In special situations, such as with embedded systems equipped with the Wayland software but lacking the X11 library, it's possible to build the FLTK library such as it contains only the Wayland backend. This is achieved building FLTK with `cmake -DFLTK_BACKEND_X11=OFF` or with `configure -disable-x11`. In that case, `FL/fl_config.h` does not define `FLTK_USE_X11`.

The rest of this chapter describes what happens when the Wayland leg has been chosen.

3.1.4 Listeners

A Wayland 'listener' is a small array of pointers to FLTK-defined callback functions associated to a Wayland-defined object, usually right after creation of this object, by a call to a specific Wayland function named following the form `wl_XXX_add_listener()`. After defined events have occurred, the Wayland compositor sends appropriate commands to the client through the socket; the event loop detects the availability of data in the socket and calls function `wayland_socket_callback()`; this function calls the appropriate member of the listener and transmits relevant information to the client app as parameters of this call. For example, this code:

```
static void surface_enter(.....) { ..... } // called when a surface enters a display
static void surface_leave(.....) { ..... } // called when a surface leaves a display
```

```
static struct wl_surface_listener surface_listener = {
    surface_enter,
    surface_leave,
```

```
};

some_pointer_type pter_to_data;
struct wl_surface *my_wl_surface;
my_wl_surface = wl_compositor_create_surface(scr_driver->wl_compositor);
wl_surface_add_listener(my_wl_surface, &surface_listener, pter_to_data);
```

creates a Wayland object of type `struct wl_surface` (roughly, a window), and associates it with a 2-member listener called `surface_listener`. After this, Wayland is expected to call the 2 listener members, `surface_listener_enter` or `surface_listener_leave`, each time `my_wl_surface` will enter or leave, respectively, a display. The arguments of these calls, not detailed here, allow the member functions to identify which surface enters or leaves which display. The `wl_surface_add_listener()` call above also associates `pter_to_data` to `my_wl_surface` as *user data*. The `wl_surface` object's "user data" can be obtained later calling function `wl_surface_get_user_data()`.

Wayland function `wl_proxy_get_listener()` returns a pointer to a Wayland object's listener provided that object is transmitted cast to type `struct wl_proxy *`. This gives a handy way to distinguish FLTK-created Wayland objects from objects of other origin: the listener of an FLTK-created object is a known FLTK listener. For example, function `Fl_Wayland_Window_Driver::surface_to_window()` uses this possibility calling `wl_proxy_get_listener( (struct wl_proxy *)wl_surface )` for any object of type `struct wl_surface`: if that object was created as in the example above, this call returns a pointer to FLTK's `surface_listener` static variable.

3.1.5 Opening a Wayland connection

Establishing a Wayland connection requires environment variable `XDG_RUNTIME_DIR` to be defined and to point to a directory containing a socket connected to a Wayland compositor. This variable is usually set by the login procedure of Wayland-friendly desktops. Which socket-file to use within that directory is determined as follows:

- the client may call `Fl::display(const char *display_name)` before `fl_open_display()` runs or use the `-display` command line argument and transmit there the socket name;
- environment variable `WAYLAND_DISPLAY` can be defined to the socket name;
- otherwise, "wayland-0" is used.

Which socket is selected determines the compositor used by the client application: that at the other end of the socket.

Establishing the connection begins with a call to `wl_display_connect(const char *display_name)`. That call is done inside function `attempt_wayland()` mentioned before with a NULL argument, or when a non default Wayland display name is specified as explained above. That call returns a `struct wl_display` pointer or NULL in case of failure. Such NULL return is the hint that allows the FLTK display opening procedure of the Wayland/X11 hybrid to recognize when Wayland access is not possible and to fallback to X11. If the call is successful, its non-NULL return is assigned to class variable `Fl_Wayland_Screen_Driver::wl_display`. The rest of the work is done in function `Fl_Wayland_Screen_Driver::open_display_platform()`. A call to `wl_registry_add_listener()` associates a 2-member listener, whose 1st member, `registry_handle_global()`, will be called by Wayland a number of times to indicate each time a protocol supported by the compositor or a system feature such as displays and keyboards.

FLTK runs this code to receive calls to `registry_handle_global()`:

```
static void sync_done(void *data, struct wl_callback *cb, uint32_t time) {
    *(struct wl_callback **)data = NULL;
    wl_callback_destroy(cb);
}
```

```
static const struct wl_callback_listener sync_listener = {
    sync_done
};
```

```
struct wl_callback *registry_cb = wl_display_sync(wl_display);
wl_callback_add_listener(registry_cb, &sync_listener, &registry_cb);
while (registry_cb) wl_display_dispatch(wl_display);
```

A pointer to an object of type `struct wl_callback` created by function `wl_display_sync()` is assigned to variable `registry_cb`. Then a 1-member listener is attached to this object. Wayland will run this listener's member function, `sync_done()`, after all calls to `registry_handle_global()` have occurred. Function `sync_done()` sets to null variable `registry_cb` and destroys the `wl_callback`. Finally, function `wl_display_dispatch()` is called as long as variable `registry_cb` is not null. Thus, when `sync_done()` runs, FLTK has received all due calls to `registry_handle_global()`.

The prototype of function `registry_handle_global` is:

```
static void registry_handle_global(void *user_data, struct wl_registry *wl_registry,
                                  uint32_t id, const char *interface, uint32_t version)
```

Each time Wayland calls `registry_handle_global()`, `interface` and `version` give the name and version of a component or feature of the Wayland system. Here is the list of the `interface` value for all protocols and system features FLTK uses:

interface	use
<code>wl_compositor</code>	create <code>wl_surface</code> objects
<code>wl_subcompositor</code>	create subwindows
<code>wl_shm</code>	create buffers and buffer factories
<code>wl_seat</code>	create the unique "seat"
<code>wl_data_device</code>	support of copy/paste/drag-n-drop
<code>wl_output</code>	received once for each display
<code>xdg_wm_base</code>	create mapped windows
<code>gtk_shell1</code>	signals Mutter is in use + titlebar gestures
<code>weston_desktop_shell</code>	signals Weston is in use
<code>org_kde_plasma_shell</code>	signals KDE/Plasma is in use
<code>zwp_text_input_manager_v3</code>	interface with Text Input Methods
<code>zxdg_decoration_manager_v1</code>	select between CSD and SSD modes

Wayland compositors typically support several other protocols (e.g., `zxdg_output_manager_v1`) that FLTK does not use.

Each time `registry_handle_global` runs with an `interface` from the table above, FLTK calls `wl_registry_bind()` which returns a pointer to a Wayland structure that will be the client's access point to the corresponding Wayland protocol or system feature. This pointer is stored in a dedicated member variable of the unique `Fl_Wayland_Screen_Driver` object of an FLTK app, or of another object accessible from this object. For example, when `interface` equals `"wl_compositor"`, the value returned by `wl_registry_bind()` is stored as member `wl_compositor` of the `Fl_Wayland_Screen_Driver` object. `registry_handle_global()` also identifies whether the Mutter, Weston, or KWin compositor is connected and stores this information in static member variable `Fl_Wayland_Screen_Driver::compositor`.

Wayland calls `registry_handle_global()` with its parameter `interface` equals to `"wl_output"` once for each screen connected to the system. Each time, an object of type `struct wl_output` is created, to which a 4-member listener is associated by function `wl_output_add_listener()`. The 3rd member of this 4-function listener, `output_done()`, runs after all initialization steps of the screen have completed and turns to `true` member `done` of a record of type `struct Fl_Wayland_Screen_Driver::output` associated to the screen. Function `sync_done()` mentioned above therefore also calls `wl_display_dispatch()` until the `done` member of all `Fl_Wayland_Screen_Driver::output` records are `true`. Overall, after return from function `sync_done()`, FLTK has been made aware of all optional protocols and features of its connected Wayland compositor, and has initialized all screens of the system.

Finally, function `wl_display_get_fd()` is called to obtain the file descriptor of the Wayland socket and a call to `Fl::add_fd()` makes FLTK listen to this descriptor in `FL_READ` mode and associates function `wayland_socket_callback()` from file `Fl_Wayland_Screen_Driver.cxx` with it. This function calls `wl_display_dispatch()` which reads and interprets data available from the file descriptor, and calls corresponding listeners. The `wl_display_dispatch()` call is repeated as long as data are available for reading.

The event loop is run by function `Fl_Unix_System_Driver::wait()` which is used by both the Wayland and X11 FLTK backends. Among various tasks, this function waits for data arriving on the file descriptors FLTK is listening. Overall, the event loop of the Wayland backend is nearly exactly the same as that used by the X11 backend. The Wayland backend differs only in the callback function handling data read from the Wayland connection socket, and in overridden functions `Fl_Wayland_Screen_Driver::poll_or_select_with_delay()` and `Fl_Wayland_Screen_Driver::poll_or_select()`.

3.1.6 Wayland windows and surfaces

Wayland defines objects called surfaces of type `struct wl_surface`. A Wayland surface "has a rectangular area which may be displayed on zero or more displays, present buffers, receive user input, and define a local coordinate system". In short, surface is the name Wayland uses for a window. Buffers allow the client app to define

the graphical content of surfaces (see [Wayland buffers](#)). FLTK creates a surface each time an [Fl_Window](#) is shown() calling function `wl_compositor_create_surface()`. Static member function `Fl_Wayland_Window_Driver::surface_to_window(struct wl_surface *)` gives the `Fl_Window*` corresponding to the surface given in argument.

FLTK recognizes 4 mutually exclusive kinds of surfaces :

- DECORATED are toplevel windows with a titlebar;
- UNFRAMED are toplevel windows without titlebar;
- POPUP correspond to menus and tooltips;
- SUBWINDOW correspond to an [Fl_Window](#) embedded in another [Fl_Window](#).

Function `Fl_Wayland_Window_Driver::makeWindow()` creates all these surfaces, creates for each a record of type `struct wld_window` (see [wld_window](#)), and stores the window kind in member variable `kind` of this record. Member variable `xid` of the window's `Fl_X` record stores the address of this record.

Except for SUBWINDOW's, each surface needs a Wayland object of type `struct xdg_surface` used to make it become a mapped window and stored in member `xdg_surface` of the window's `wld_window` record. For DECORATED windows, this object is created inside libdecor and transmitted to FLTK by function `libdecor_frame_get_xdg_surface()`. For UNFRAMED and POPUP windows, it's created by function `xdg_wm_base_get_xdg_surface()`. Finally, each surface is also associated to one more Wayland object whose type varies with the window's kind. These explain this part of the `wld_window` record:

```
union {
    struct libdecor_frame *frame; // created for DECORATED by libdecor_decorate()
    struct wl_subsurface *subsurface; // created for SUBWINDOW by wl_subcompositor_get_subsurface()
    struct xdg_popup *xdg_popup; // created for POPUP by xdg_surface_get_popup()
    struct xdg_toplevel *xdg_toplevel; // created for UNFRAMED by xdg_surface_get_toplevel()
};
```

Except for SUBWINDOW's, each surface is associated to a 'configure' function that Wayland calls one or more times when the window is going to be mapped on the display. The 'configure' function of DECORATED surfaces is `handle_configure()` which is the 1st member of a 4-member listener named `libdecor_frame_iface` associated to a decorated window when it's created calling `libdecor_decorate()`. Finally, a call to `libdecor_frame_map()` triggers the process of mapping the newly created DECORATED surface on a display. Wayland calls `handle_configure()` twice during this process. The first `handle_configure()` run allows to set the window's `xdg_surface` object which is returned by function `libdecor_frame_get_xdg_surface()`. FLTK distinguishes the first from the second run of `handle_configure()` by looking at the `xdg_surface` member variable that's NULL at the beginning of the 1st run and not NULL later. Wayland calls `handle_configure()` also during operations such as resizing, minimizing (see below). With the help of a few calls to libdecor functions, FLTK obtains in this function all needed information about the size and state of the mapped window. The 'configure' functions of UNFRAMED and POPUP surfaces are `xdg_surface_configure()`, `xdg_toplevel_configure()` and `popup_configure()`. The mapping process of these surfaces is triggered by a call to `wl_surface_commit()`. These 'configure' functions transmit effective window size information to FLTK. Also, they are where the window's `Fl_Window_Driver::wait_for_expose_value` member variable is set to 0 to indicate that the window has been mapped to display. **Caution**: there are some small differences between how and when the various Wayland compositors call `handle_configure()`.

When a decorated window changes size, whatever the cause of it, Wayland calls `handle_configure()` which sets member variable `Fl_Wayland_Window_Driver::in_handle_configure` to true and calls the window's virtual `resize()` function which ultimately runs `Fl_Wayland_Window_Driver::resize()` which calls [Fl_Group::resize\(\)](#) to perform FLTK's resize operations and `Fl_Wayland_Graphics_Driver::buffer_release()` to delete the existing window buffer that's not adequate for the new window size. At the end of the run of `handle_configure()`, `in_handle_configure` is set back to false. When the window size change is caused by the app itself calling the window's `resize()` function, `Fl_Wayland_Window_Driver::in_handle_configure` is false. This allows `Fl_Wayland_Window_Driver::resize()` to detect that Wayland needs be informed of the desired size change, which gets done by a call to `libdecor_frame_commit()`. Wayland later calls `handle_configure()` and events described above unfold.

Wayland generally does not provide a way to control where the compositor should map a window in the system displays. Nevertheless, for multi-display systems, Wayland allows to control on what display should the compositor map a fullscreen window. That is done inside function `handle_configure()` which calls `libdecor_frame_set_fullscreen()` for DECORATED windows and inside function `xdg_toplevel_configure()` which calls `xdg_toplevel_set_fullscreen()` for UNFRAMED. The `struct wl_output` pointer for the targeted display is transmitted as 2nd argument of these calls.

3.1.7 Menu windows and other popups

Menu windows, tiny menu title windows, and tooltip windows are implemented using Wayland's popup mechanism which allows to position a popup window relatively to a previously mapped window, itself a popup or another kind of window, with the restriction that any popup must overlap or at least touch that other window. Member function `Fl_Wayland_Window_Driver::makeWindow` calls member function `Fl_Wayland_Window_Driver::process_menu_or_tooltip` to create all popups.

This function gets called after FLTK has computed using a given algorithm the desired (x,y) position of the popup window's top-left corner, using coordinates centered on the top-left corner of the toplevel window from which the popup originates. This algorithm is able to prevent popups from being positioned beyond the screen borders under the assumption that the position of a toplevel window inside a screen is known. While this assumption holds for other platforms, it does not for the Wayland platform. The FLTK code for the Wayland platform therefore modifies the algorithm that FLTK uses to compute the position of menu windows. The key information used by this algorithm is obtained by member function `Fl_Window_Driver::menu_window_area` which computes the coordinates of the rectangle where menu windows are allowed to be positioned. Under other platforms, this function just returns the origin and size of the work area of the screen in use. In contrast, the Wayland platform handles two situations differently :

- For menu windows that are not taller than the display in use, the Wayland-overridden member function `Fl_Wayland_Window_Driver::menu_window_area` returns large negative origin and large width and height values. This lets the standard FLTK algorithm position the menu relatively to its window of origin without concern about screen limits, and relies on Wayland's constraint mechanism described below to prevent the menu from going beyond these limits, without FLTK having to know where they are.
- Menu windows taller than the screen where they are mapped need special handling described in detail in a comment above the source code of function `Fl_Wayland_Window_Driver::process_menu_or_tooltip`.

Function `Fl_Wayland_Window_Driver::process_menu_or_tooltip` first computes `origin_win`, pointer to the `Fl_Window` relatively to which the popup is to be positioned. Window `origin_win` is the parent menu window when the popup is a sub-menu; it's the tiny windowtitle when the popup is a menu with a title; otherwise, it's the window containing the point of origin of the popup. An object of type `struct xdg_positioner` created by function `xdg_wm_base_create_positioner()` is used to express the rules that will determine the popup position relatively to `origin_win` as follows:

- Function `xdg_positioner_set_anchor_rect()` determines a rectangle in `origin_win` relatively to which the popup is to be positioned. When the popup to be created is a menu window spawned by an `Fl_Menu_Bar`, that rectangle is the full area of the menu title window. Otherwise, that rectangle is an adequately located point.
- Function `xdg_positioner_set_size()` sets the popup size.
- The `xdg_positioner_set_anchor(positioner, XDG_POSITIONER_ANCHOR_BOTTOM_LEFT);` and `xdg_positioner_set_gravity(positioner, XDG_POSITIONER_GRAVITY_BOTTOM_RIGHT);` calls position the popup so that its top-left corner is initially below and at right of the bottom-left corner of the `origin_win`'s anchor rectangle.
- The call to `xdg_positioner_set_offset()` further changes the popup vertical position.
- The call to `xdg_positioner_set_constraint_adjustment()` uses constraint flags `XDG_POSITIONER_CONSTRAINT_ADJUSTMENT_SLIDE_X` and `XDG_POSITIONER_CONSTRAINT_ADJUSTMENT_SLIDE_Y` which mean that the compositor will move the popup horizontally and vertically if its initial position would make it expand beyond the edges of the screen. Furthermore, flag `XDG_POSITIONER_CONSTRAINT_ADJUSTMENT_FLIP_Y` is added when the popup is a menu window spawned by an `Fl_Menu_Bar`; this has the popup flipped above the `Fl_Menu_Bar` if there's not enough screen room below it for the popup.
- Finally, a call to function `xdg_surface_get_popup()` creates the popup accounting for position rules listed above. The positioner is then deleted by `xdg_positioner_destroy()`, a listener is associated to the popup surface with `xdg_popup_add_listener()`, and a call to `wl_surface_commit()` triggers the mapping of the popup on the display.

Overall, the expected coordinates of the top-left corner of the popup relatively to `origin_win` are `popup_x`, `popup_y`. They are memorized in a record of FLTK-defined type `struct win_positioner` that's associated to the popup listener. When the compositor maps the popup, function `popup_configure`, the first element of the popup listener, runs and receives as arguments the coordinates of the popup top left and its size. These values account for the positioning constraints of the popup which may have moved it to avoid screen borders. This function can therefore detect whether constraints applied have modified the effective popup location in comparison to the expected coordinates which are available as member variables of the `struct win_positioner` record mentioned above. That's key to the handling by FLTK of tall menu windows.

Groups of popups containing a menutitle, the associated menuwindow, and optionally a submenu window and that don't belong to an `Fl_Menu_Bar` are mapped in a different order: the menuwindow is mapped first, and the menutitle is mapped second above it as a child popup. Function `Fl_Window_Driver::is_floating_title()` detects when such a menutitle is created, static member variable `previous_floatingtitle` is assigned the value of this menutitle, and the menutitle is mapped only after the menuwindow has been mapped, as a child of it. This positions better the popup group in the display relatively to where the popup was created.

3.1.8 Fl_Wayland_Graphics_Driver and Fl_Cairo_Graphics_Driver

The Wayland platform of FLTK uses an `Fl_Wayland_Graphics_Driver` object for all its on-screen drawing operations. This object is created by function `Fl_Graphics_Driver::newMainGraphicsDriver()` called by `Fl_Display_Device::display_device()` when the library opens the display. New `Fl_Wayland_Graphics_Driver` objects are also created for each `Fl_Image_Surface` and each `Fl_Copy_Surface` used, and deleted when these objects are deleted.

Class `Fl_Wayland_Graphics_Driver` derives from class `Fl_Cairo_Graphics_Driver` which implements all the FLTK drawing API for a Cairo surface. Function `Fl_Wayland_Graphics_Driver::cairo_init()` creates the Cairo surface used by each `Fl_Wayland_Graphics_Driver` object by calling `cairo_image_surface_create_for_data()` for the window's or offscreen's `draw_buffer` (see below).

Class `Fl_Cairo_Graphics_Driver` is also used by the X11 leg of the hybrid Wayland-X11 platform because this leg draws to the display with an `Fl_X11_Cairo_Graphics_Driver` object which derives from class `Fl_Cairo_Graphics_Driver`. Finally, `Fl_Cairo_Graphics_Driver` is also used, in the form of an object from its derived class `Fl_PostScript_Graphics_Driver`, when the hybrid Wayland-X11 platform draws PDF or PostScript, or when the classic X11 platform uses Pango and draws PDF or PostScript. This happens when classes `Fl_PDF_File_Surface`, `Fl_PostScript_File_Device` and `Fl_Printer` are used.

3.1.9 Wayland buffers

Wayland uses buffers, objects of type `struct wl_buffer`, to draw to surfaces. In principle, one or more buffers can be associated to a surface, and functions `wl_surface_attach()` and `wl_surface_commit()` are called to first attach one such buffer to the surface and then inform the compositor to map this buffer's graphics content on the display. Wayland buffers can use various memory layouts. FLTK uses `WL_SHM_FORMAT_ARGB8888`, which is the same layout as what Cairo calls `CAIRO_FORMAT_ARGB32`.

FLTK calls function `Fl_Wayland_Window_Driver::make_current()` before drawing to any `Fl_Window`. Member `buffer` of this `Fl_Window`'s `struct wld_window` (see `wld_window`) is NULL when the window has just been created or resized. In that case, FLTK calls `Fl_Wayland_Graphics_Driver::create_wld_buffer()` which returns a pointer to a `struct wld_buffer` containing

- a Wayland buffer, member `wl_buffer`;
- a Cairo image surface, created by a call to `Fl_Wayland_Graphics_Driver::cairo_init()`.

Each of these two objects encapsulates a byte array of the same size and the same memory layout destined to contain the `Fl_Window`'s graphics. The Cairo image surface object is where FLTK draws. The Wayland buffer is what Wayland maps on the display. FLTK copies the Cairo surface's byte array to the Wayland buffer's byte array before beginning the mapping operation. If `width` and `height` are a window's dimensions in pixels,

```
int stride = cairo_format_stride_for_width(CAIRO_FORMAT_ARGB32, width);
int size = stride * height;
```

give `size`, the common size of both byte arrays.

The effective creation of the `wl_buffer` object is delayed until function `Fl_Wayland_Graphics_Driver::buffer_commit()` gets called. Section [Buffer factories](#) below details how FLTK creates `wl_buffer` objects.

The `struct Fl_Wayland_Graphics_Driver::wld_buffer` (see `wld_buffer`) contains a pointer to the byte array of the Cairo image surface (member `draw_buffer.buffer`), information about the Wayland buffer

(members `wl_buffer` and `data`), the common size of the Cairo surface's and Wayland buffer's byte arrays (member `draw_buffer.data_size`), and other information. A pointer to this struct `Fl_Wayland_Graphics_Driver::wld_buffer` is memorized as member `buffer` of the `Fl_Window`'s `wld_window`. All drawing operations to the `Fl_Window` then modify the content of the Cairo image surface.

Function `Fl_Wayland_Window_Driver::flush()` is in charge of sending FLTK graphics data to the display. That is done by calling function `Fl_Wayland_Graphics_Driver::buffer_commit()` which creates the struct `wl_buffer` object calling `create_shm_buffer()` if that was not done before, copies the byte array of the Cairo surface to the Wayland buffer's starting memory address, and calls functions `wl_surface_attach()` and `wl_surface_commit()`. Before calling `Fl_Window::flush()`, FLTK has computed a damaged region. If that region is not null, `Fl_Wayland_Graphics_Driver::buffer_commit()` copies only the damaged parts of the Cairo surface to the Wayland buffer and calls function `wl_surface_damage_buffer()` for these parts to inform the compositor of what parts of the surface need its attention.

Wayland buffer deletion

Each `wld_buffer` record contains boolean member `in_use` which is set to `true` just before the buffer gets committed, and boolean member `released` which is set to `true` when FLTK no longer needs the buffer and calls `Fl_Wayland_Graphics_Driver::buffer_release()`. FLTK's buffer-creating function, `Fl_Wayland_Graphics_Driver::create_shm_buffer()`, attaches a 1-member listener to each buffer which Wayland calls after a commit operation to indicate the client is allowed to re-use the buffer. This listener's member function, `buffer_release_listener()`, turns to false member `in_use` of the buffer's `wld_buffer` record. Since the two events 'FLTK no longer needs the buffer' and 'the client is allowed to re-use the buffer' can arrive in any order, FLTK deletes the struct `wl_buffer` object by running `do_buffer_release()` only after both events happened, that is, when `in_use` is false and `released` is true. That's why function `do_buffer_release()` is called by both functions `Fl_Wayland_Graphics_Driver::buffer_release()` and `buffer_release_listener()`.

3.1.10 Throttling window redraws

FLTK uses Wayland's synchronization mechanism to make sure any committed `wl_buffer` is not changed while the compositor is using it and to refrain from calling `wl_surface_commit()` more frequently than the system can process it. Firstly, as seen above, Wayland calls function `buffer_release_listener()` when the client is free to reuse or destroy a given `wl_buffer`. FLTK won't change or destroy a committed `wl_buffer` before that call. Second, this 2-step mechanism prevents Wayland clients from committing new buffer states too frequently:

- `Fl_Wayland_Graphics_Driver::buffer_commit()` first calls function `wl_surface_frame()` to obtain a pointer to a struct `wl_callback` object and stores it as member `frame_cb` of the surface's `wld_window`. Then it calls `wl_callback_add_listener()` to associate this object to the FLTK-defined, callback function `surface_frame_done()`. It next calls `wl_surface_commit()`. Together, these 3 calls instruct Wayland to start mapping the buffer content to the display and to call `surface_frame_done()` later, when it will have become ready for another mapping operation.
- Later, `surface_frame_done()` runs and destroys the `wl_callback` object by function `wl_callback_destroy()` and sets member `frame_cb` to `NULL`.

Member variable `draw_buffer_needs_commit` of the `wld_buffer` is also important in this mechanism : it informs FLTK that the graphics buffer has changed and needs being committed. This variable is turned `true` every time a graphics operation changes the buffer content and turned `false` when the buffer gets committed.

This procedure ensures that FLTK never calls `wl_surface_commit()` before the compositor becomes ready for a new commit because `Fl_Wayland_Window_Driver::flush()` calls `Fl_Wayland_Graphics_Driver::buffer_commit()` only if `frame_cb` is `NULL`. If it's not `NULL`, the exact content of function `surface_frame_done()` :

```
static void surface_frame_done(void *data, struct wl_callback *cb, uint32_t time) {
    struct wld_window *window = (struct wld_window *)data;
    wl_callback_destroy(cb);
    window->frame_cb = NULL;
    if (window->buffer && window->buffer->draw_buffer_needs_commit) {
        Fl_Wayland_Graphics_Driver::buffer_commit(window);
    }
}
```

has the effect that when the mapping operation eventually completes, Wayland runs `surface_frame_done()`, which, if the buffer's `draw_buffer_needs_commit` member is true, calls `Fl_Wayland_Graphics_Driver::buffer_commit()` anew. The net result is that the screen shows the most recent surface content. This synchronization mechanism is also used when performing an interactive window resize operation. During such operation, the compositor informs the client an interactive resize is being performed and sends window resize commands at high rate (~60 Hz) to the client via the socket. Libdecor turns on flag `LIBDECOR_WINDOW_STATE_RESIZING` to inform the client, and runs function `handle_configure()` for each received resize command. Before calling `Fl_Group::resize()` and later `Fl_Window::draw()`, `handle_configure()` tests whether `window->frame_cb` is NULL. When it's not because a previous resize operation is being performed, the current resize command is skipped. At the end of the interactive resize, flag `LIBDECOR_WINDOW_STATE_RESIZING` is off and Wayland sends a final resize command which is not skipped. Overall, this ensures the client program resizes its window as frequently as it can without falling behind resize commands sent by the compositor. To account for a bug in Mutter (issue #878), the `window->frame_cb` object is not created when a toplevel window is being resized and is entirely covered by one subwindow.

Progressive window drawing

FLTK supports progressive drawing when an app calls function `Fl_Window::make_current()` at any time and then calls the FLTK drawing API. This is made possible in function `Fl_Wayland_Window_Driver::make_current()` with

```
// to support progressive drawing
if ( (!Fl_Wayland_Window_Driver::in_flush_) && window->buffer && (!window->frame_cb)
    && window->buffer->draw_buffer_needs_commit && (!wait_for_expose_value) ) {
    Fl_Wayland_Graphics_Driver::buffer_commit(window);
}
```

Thus, `buffer_commit()` runs only when `frame_cb` is NULL. If an app rapidly performs calls to `Fl_Window::make_current()` and to drawing functions, FLTK will copy `draw_buffer` to the Wayland buffer and instruct Wayland to map it to the display when `frame_cb` is NULL which means that the compositor is ready to start performing a mapping operation. This occurs when the progressive drawing operation begins. Later, `frame_cb` is generally found non NULL when `Fl_Wayland_Window_Driver::make_current()` runs because the compositor is busy processing the previous Wayland buffer. When the compositor has completed this processing, the client app runs `surface_frame_done()` which, provided member variable `draw_buffer_needs_commit` is true, calls `Fl_Wayland_Graphics_Driver::buffer_commit()`. This makes the compositor map the Wayland buffer in its new, more advanced, state.

An example of progressive drawing is given by FLTK's mandelbrot test app. When set to fullscreen, this app can be seen to progressively fill its window from top to bottom by blocks of lines, each block appearing when the compositor is ready to map a new buffer. When the compositor is not ready, the app does not block but continues computing and drawing in memory but not on display more lines of the desired Mandelbrot graph.

3.1.11 Buffer factories

Wayland names *buffer factory* a software procedure that constructs objects of type `struct wl_buffer` for use by a client application. FLTK creates a `wl_buffer` object each time an `Fl_Window` is mapped on a display or resized. That's done by member function `Fl_Wayland_Graphics_Driver::create_shm_buffer()` which follows this 3-step procedure to create a "buffer factory" for FLTK and to construct Wayland buffers from it:

- Libdecor function `libdecor_os_create_anonymous_file(off_t size)` creates an adequate file and mmap's it. This file lives in RAM because it is created by function `memfd_create()`. FLTK sets this file size to 10 MB unless the size of the buffer to be created is larger; in that case the anonymous file is sized to twice the buffer size.
- Wayland function `wl_shm_create_pool()` shares this mmap'ed memory with the Wayland compositor and returns an object of type `struct wl_shm_pool` which encapsulates this memory. A record of type `struct Fl_Wayland_Graphics_Driver::wld_shm_pool_data` is created and associated to the newly created `wl_shm_pool` by `wl_shm_pool_set_user_data()`. This record stores the starting address (`pool_memory`) and size (`pool_size`) of the pool's encapsulated memory. The record also contains member buffers of type `struct wl_list` which stores the access point to the linked list of `wl_buffer` objects that will be created from the `wl_shm_pool`.
- A variable named `chunk_offset` represents the offset within the pool's shared memory available for the buffer being constructed. It equals 0 when the pool has just been created and is updated as detailed below

each time a buffer is created from the pool. A record of type `struct Fl_Wayland_Graphics_Driver::wld_buffer` is created. This record will contain (member `wl_buffer`) the address of a `wl_buffer` object that's created by function `wl_shm_pool_create_buffer()`. This `wl_buffer` object encapsulates a section of a given size of the pool's shared memory beginning at offset `chunk_offset` in it. Quantity `pool_memory + chunk_offset` is therefore the address of the beginning of the mmap'ed memory section encapsulated by this `wl_buffer`. Member `shm_pool` of the newly constructed `Fl_Wayland_Graphics_Driver::wld_buffer` object is set to the address of the current `wl_shm_pool` object. This record is added to the head of the linked list of current pool's buffers by a call to `wl_list_insert()`. At that point, a `struct Fl_Wayland_Graphics_Driver::wld_buffer` record is part of the linked list of all such records corresponding to `wl_buffer` objects created from the same `wl_shm_pool` object, and member `shm_pool` of this record gives the address of this `wl_shm_pool`. When a new `struct Fl_Wayland_Graphics_Driver::wld_buffer` record is to be created,

```
struct wld_shm_pool_data *pool_data =
    (struct wld_shm_pool_data *)wl_shm_pool_get_user_data(pool);
struct Fl_Wayland_Graphics_Driver::wld_buffer *record = wl_container_of(pool_data->buffers.next,
    record, link);
int chunk_offset = ((char*)record->data - pool_data->pool_memory) + record->data_size;
```

gives the offset within the current pool's mmap'ed memory available for a new `wl_buffer`. Macro `wl_container_of()` gives the address of a record belonging to a linked list of records of the same type.

A window's `wl_buffer` is re-filled by graphics data and committed each time the window gets redrawn, and is set to be destroyed by function `Fl_Wayland_Graphics_Driver::buffer_release()` when `Fl_Window::hide()` runs or the window is resized. When the `wl_buffer` is no longer in use, function `do_buffer_release()` gets called as explained above. It destroys the `wl_buffer` with `wl_buffer->_destroy()`, and removes the corresponding `Fl_Wayland_Graphics_Driver::wld_buffer` record from the linked list of buffers from the same `wl_shm_pool`. Since new `Fl_Wayland_Graphics_Driver::wld_buffer` records are added at the head of the linked list, and since the record at the head of this list is used to compute the offset within the pool's mmap'ed memory available for a new `wl_buffer`, destruction of the last created `wl_buffer` allows to re-use the destroyed buffer's pool's memory for a new `wl_buffer`.

When function `do_buffer_release()` finds the list of buffers from a given pool empty, two situations can occur. 1) This pool is the current pool. Its mmap'ed memory will be re-used from offset 0 to create future `wl_buffer` objects. 2) This pool is not current. It gets destroyed with `wl_shm_pool_destroy()`, the pool's mmap'ed memory is munmap'ed, and the pool's associated `struct wld_shm_pool_data` is freed. In situation 1) above, the next `wl_buffer` to be created can need more memory than the current pool's memory size. If so, the current pool gets destroyed and replaced by a new, larger pool.

If the sum of `chunk_offset` plus the buffer size is larger than the current pool's size when function `create_shm_buffer()` is called, `chunk_offset` is reset to 0, and a new `wl_shm_pool` object is created and used by FLTK's "buffer factory". This mechanism allows to access new mmap'ed memory when `chunk_offset` reaches the end of the previous mmap'ed section.

Wayland uses also `wl_buffer` objects to support cursors. FLTK uses the "buffer factory" described here when creating custom cursors (see [custom-cursor](#)) with function `Fl_Wayland_Window_Driver::set_cursor(const Fl_RGB_Image *, ...)` which calls `create_shm_buffer()` via `set_cursor_4args()`, `custom_offscreen()` and `create_wld_buffer()`. In contrast, standard shaped-cursors (e.g., `FL_CURSOR_INSERT`) use their own "buffer factory" inside Wayland functions such as `wl_cursor_theme_get_cursor()`. Therefore, the fact that the `wl_buffer` objects behind standard cursors are never destroyed doesn't prevent disused `struct wl_shm_pool` objects from being freed because those buffers come from a distinct "buffer factory". The "buffer factory" described here is also used by function `offscreen_from_text()` when displaying dragged text in a DnD operation.

3.1.12 Displays and HighDPI support

Wayland uses the concept of *seat* of type `struct wl_seat` which encompasses displays, a keyboard, a mouse, and a trackpad. Although Wayland may be in principle able to deal with several seats, FLTK's Wayland platform is conceived for one seat only. That seat may contain one or more displays, which Wayland calls *outputs*, of type `struct wl_output`.

As written above, function `registry_handle_global()` discovers the available seat at start-up time. This function also associates a listener to each display connected to the system by calling function `wl_output->add_listener()`. This listener's member functions run at program startup when Wayland discovers its displays (see [Opening a Wayland connection](#)). Member `output_mode` runs also when the display is resized and mem-

ber `output_scale` also when the Wayland scale factor (see below) is changed. FLTK defines type `struct Fl_Wayland_Screen_Driver::output` to store display size and scaling information. One such record is created for each display. These records are put in a `struct wl_list` accessible from member `outputs` of the single `Fl_Wayland_Screen_Driver` object.

FLTK uses 2 distinct scaling parameters for each display:

- `int wld_scale;` This member variable of the `struct Fl_Wayland_Screen_Driver::output` record typically equals 1 for standard, and 2 for HighDPI displays. The effect of value `n` of variable `wld_scale` is that 1 Wayland graphics unit represents a block of $n \times n$ pixels. Another effect is that a drawing buffer for a surface of size $W \times H$ units contains $W * n * H * n * 4$ bytes. Member function `output_scale()` mentioned above sets this value for each system's display at startup time. Member function `Fl_Wayland_Graphics_Driver::buffer_commit()` informs the Wayland compositor of the value of `wld_scale` calling `wl_surface_set_buffer_scale()` which is enough to make FLTK apps HighDPI-aware. Under the gnome and KDE desktops, this parameter is visible in the "Settings" app, "Displays" section, "Scale" parameter which is 200% on HighDPI displays.
- `float gui_scale;` This other member variable is where FLTK's own GUI scaling mechanism with `ctrl+/-/0/` keystrokes and with environment variable `FLTK_SCALING_FACTOR` operates: when FLTK is scaled at 150%, `gui_scale` is assigned value 1.5. Function `Fl_Wayland_Screen_Driver::scale(int n, float f)` assigns value `f` to the `gui_scale` member variable of display # `n`. This variable is used by function `Fl_Wayland_Window_Driver::make_current()` when it calls `Fl_Wayland_Graphics_Driver::set_buffer()` that scales the graphics driver by this factor with `cairo_scale()`.

Overall, an FLTK object, say an `Fl_Window`, of size $W \times H$ FLTK units occupies `int(W * gui_scale) * wld_scale x int(H * gui_scale) * wld_scale` pixels on the display.

When an `Fl_Window` is to be shown, `Fl_Wayland_Window_Driver::makeWindow()` creates a `struct wl_surface` with `wl_compositor_create_surface()` and associates it calling `wl_surface_add_listener()` with a 2-member listener called `surface_listener` encharged of managing as follows the list of displays where this `wl_surface` will map. The `Fl_Window` possesses an initially empty linked list of displays accessible at member `outputs` of the window's `wld_window` record. When the `Fl_Window`, or more exactly its associated `struct wl_surface` is mapped on a display, member `surface_enter()` of `surface_listener` runs. This function adds the display where the surface belongs to the end of the linked list of displays for this surface. When a surface is dragged or enlarged across the edge of a display in a multi-display system and expands on a second display, `surface_enter()` runs again, and this surface's list of displays contains 2 items. When a surface leaves a display, member `surface_leave()` of `surface_listener` runs. It removes that display from the surface's list of displays. Each time the first item of a surface's list of displays changes, function `change_scale()` is called and applies that display's `gui_scale` value to that surface calling `Fl_Window_Driver::screen_num(int)`. When a window is unmapped by function `Fl_Wayland_Window_Driver::hide()`, the surface's list of displays is emptied.

Fractional scaling

The KWin and gnome compositors allow to use *fractional scaling* that can take values between 100% and 400% that are not a multiple of 100%. Wayland implements this rendering all `wl_surface`'s as if the scaling had the next value above that is a multiple of 100% (e.g., 175% --> 200%), and downsizing them to the desired fractional scale value at the compositing stage. Seen from FLTK, everything runs with `wld_scale` having an integer value (1, 2, 3 or 4).

Some gnome versions may natively support fractional scaling. Others require to use these commands to make them accept/refuse fractional scaling:

```
gsettings set org.gnome.mutter experimental-features "[ 'scale-monitor-framebuffer' ]"
gsettings reset org.gnome.mutter experimental-features
```

3.1.13 Mouse and trackpad handling

FLTK receives information about mouse and pointer events via a 'listener' made up of 5 pointers to functions which Wayland calls when events listed in table below occur. These functions receive from Wayland enough information in their parameters to generate corresponding FLTK events, that is, calls to `Fl::handle(int event_type, Fl_Window *)`.

listener function	called by Wayland when	resulting FLTK events
pointer_enter	pointer enters a window	FL_ENTER
pointer_leave	pointer leaves a window	FL_LEAVE
pointer_motion	pointer moves inside a window	FL_MOVE
pointer_button	state of mouse buttons changes	FL_PUSH, FL_RELEASE
pointer_axis	trackpad is moved vertically or horizontally	FL_MOUSEWHEEL

pointer_listener is installed by a call to function `wl_pointer_add_listener()` made by function `seat_capabilities()` which is itself another 'listener' made up of 2 function pointers

```
static struct wl_seat_listener seat_listener = {
    seat_capabilities,
    seat_name
};
```

installed by a call to function `wl_seat_add_listener()` made by function `registry_handle_global()` when it receives a "wl_seat" interface.

Handling middle mouse button clicks on window titlebars

The gnome desktop, via its `gnome-tweaks` application, allows to determine what happens when a middle mouse button click occurs on a window titlebar. To obey this setting, FLTK implements part of the `GTK Shell` protocol as follows. Mutter, gnome's Wayland compositor, declares its support of the `GTK Shell` protocol calling `registry_handle_global()` with its interface argument equal to "gtk_shell". FLTK initializes then member variable `seat->gtk_shell` of type `struct gtk_shell*`.

Member functions of `pointer_listener` mentioned above run for all mouse events on all `wl_surface` objects. The table above describes what these functions do for mouse events on FLTK-created `wl_surface` objects. But they also run for the `libdecor`-created `wl_surface` objects corresponding to window titlebars. Thus, member function `pointer_enter()` runs when the mouse enters a titlebar. It calls `Fl_Wayland_Screen_Driver::event_coords_from_surface()` which calls `Fl_Wayland_Window_Driver::surface_to_window()` which, as mentioned above, can distinguish FLTK-created from non FLTK-created `wl_surface` objects. This allows `pointer_enter()` to identify the entered surface as a titlebar and to assign static global variable `gtk_shell_surface` with the titlebar's `wl_surface` when the mouse enters a titlebar. Similarly, member function `pointer_leave()` sets `gtk_shell_surface` to NULL when the mouse leaves this titlebar. When there's a click on a titlebar, member function `pointer_button()` runs this code

```
if (gtk_shell_surface && state == WL_POINTER_BUTTON_STATE_PRESSED && button == BTN_MIDDLE) {
    struct gtk_surface* gtk_surface = gtk_shell_get_gtk_surface(seat->gtk_shell, gtk_shell_surface);
    gtk_surface->titlebar_gesture(gtk_surface, serial, seat->wl_seat, GTK_SURFACE_GESTURE_MIDDLE_CLICK);
    gtk_surface->release(gtk_surface);
    return;
}
```

which ensures that what `gnome-tweaks` has assigned to middle-click events is executed. At this point, FLTK obeys what `libdecor` decides for right-click (display the window menu) and double-click (maximize the window) events on titlebars which may diverge from `gnome-tweaks` settings.

3.1.14 Wayland cursors

Wayland defines types `struct wl_cursor` and `struct wl_cursor_theme` to hold cursor-related data. FLTK uses function `init_cursors()` from file `Fl_Wayland_Screen_Driver.cxx` to obtain the 'cursor theme' name using function `libdecor_get_cursor_settings()` of library `libdecor`. Function `wl_cursor_theme_load()` then returns a pointer to an object of type `struct wl_cursor_theme` stored in member variable `cursor_theme` of the `Fl_Wayland_Screen_Driver::seat` record. Function `init_cursors()` is itself called by a 'listener' called `seat_capabilities()` installed when function `registry_handle_global()` receives a "wl_seat" interface, at program startup. It is also called when the value of the Wayland scaling factor changes: `output_done()` calls `try_update_cursor()` calls `init_cursors()`. Function `output_done()` belongs to a 'listener' installed when function `registry_handle_global()` receives a "wl_output" interface.

Each time `Fl_Window::cursor(Fl_Cursor)` runs, FLTK calls `Fl_Wayland_Window_Driver::set_cursor(Fl_Cursor)` which calls `wl_cursor_theme_get_cursor()` to set the current cursor shape to one of the standard shapes from the `Fl_Cursor` enumeration. This Wayland function selects a cursor shape based on the current `wl_cursor_theme` object and a cursor name and returns a pointer to a `struct wl_cursor`. Under the gnome desktop, cursor names are the files of directory `/usr/share/icons/XXXX/cursors/` where XXXX is the 'gnome cursor theme' (default= Adwaita). For example, what FLTK calls

`FL_CURSOR_INSERT` corresponds to file `xterm` therein. The full correspondance between `Fl_Cursor` values and names of files therein is found in function `Fl_Wayland_Window_Driver::set_cursor(Fl_Cursor)`. FLTK stores in member variable `default_cursor` of the `Fl_Wayland_Screen_Driver::seat` record a pointer to the currently used `wl_cursor` object, and the current `Fl_Cursor` value in member `standard_cursor` of the `Fl_Wayland_Window_Driver` object.

Finally, function `do_set_cursor()` of file `Fl_Wayland_Screen_Driver.cxx` makes the system pointer use the current `wl_cursor` object to draw its shape on screen. That's done with a call to `wl_pointer_set_cursor()` and a few other functions.

Custom cursor shapes

To support custom cursors, FLTK presently uses a non-public type, `struct cursor_image`, defined in file `Fl_Wayland_Window_Driver.cxx` as follows:

```
struct cursor_image {
    struct wl_cursor_image image;
    struct wl_cursor_theme *theme;
    struct wl_buffer *buffer;
    int offset;
};
```

This definition has been copied to the FLTK source code from file `wayland-cursor.c` of the Wayland project source code because it's not accessible via Wayland header files. It shows that a pointer to a `cursor_image` object can also be viewed as a pointer to the embedded `struct wl_cursor_image` object, this one being part of the public Wayland API. It also shows that a `struct cursor_image` object has an associated `struct wl_buffer` object used to contain the cursor's graphics.

Function `Fl_Wayland_Window_Driver::set_cursor(const Fl_RGB_Image *rgb, int hotx, int hoty)` gives FLTK support of custom cursor shapes. It calls `Fl_Wayland_Window_Driver::set_cursor_4args()` that creates a `cursor_image` object, allocates the corresponding `wl_buffer` by a call to `Fl_Wayland_Graphics_Driver::create_shm_buffer()` via `custom_offscreen()` and `create_wld_buffer()` and draws the cursor shape into that buffer using the offscreen-drawing method of FLTK.

The public type `struct wl_cursor` is essentially an array of `wl_cursor_image` objects and a name:

```
struct wl_cursor {
    unsigned int image_count;
    struct wl_cursor_image **images;
    char *name;
};
```

Function `Fl_Wayland_Window_Driver::set_cursor_4args()` also creates a `struct wl_cursor` object containing a single `wl_cursor_image`, which is in fact the `cursor_image`. Finally, a `struct Fl_Wayland_Window_Driver::custom_cursor` (see [wld_window](#)) is allocated and used to memorize the `struct wl_cursor` and the cursor's image and hotspot. A pointer to this `struct Fl_Wayland_Window_Driver::custom_cursor` object is stored in member `custom_cursor` of the window's [wld_window](#).

Function `Fl_Wayland_Window_Driver::set_cursor_4args()` is also called when a window with a custom cursor is moved between distinct displays or when a display is rescaled to adapt the cursor size to the new display's scale factor.

Static member function `Fl_Wayland_Window_Driver::delete_cursor()` is used to delete any custom cursor shape. This occurs when a window associated to a custom cursor is un-mapped and when such a window gets associated to a standard cursor or to a new custom cursor.

3.1.15 Keyboard support

The "Mouse handling" section above mentioned function `seat_capabilities()` that Wayland calls when the app discovers its "seat". Presence of flag `WL_SEAT_CAPABILITY_KEYBOARD` in argument `capabilities` of this function indicates that a keyboard is available. In that case, a call to `wl_seat_get_keyboard()` returns a pointer stored in member `wl_keyboard` of the `Fl_Wayland_Screen_Driver::seat` object, and a call to `wl_keyboard_add_listener()` installs a 6-member listener of type `struct wl_keyboard_listener`. These 6 FLTK-defined, callback functions are used as follows.

1) Function `wl_keyboard_keymap()` runs when the app starts and also if the keyboard layout is changed during run-time. It allows initialization of access to this keyboard. Noticeably, member `xkb_state` of type `struct xkb_state*` of the current `Fl_Wayland_Screen_Driver::seat` record is adequately initialized.

2-3) Functions `wl_keyboard_enter()` and `wl_keyboard_leave()`, called when focus enters and leaves a surface, send `FL_FOCUS` and `FL_UNFOCUS` events to the [Fl_Window](#) object corresponding to this surface.

4) Function `wl_keyboard_key()` runs each time a keyboard key is pressed or released. Its argument `key`, to which 8 must be added, provides the keycode via function `xkb_state_key_get_one_sym()` and then the corresponding text via function `xkb_state_key_get_utf8()` which is put in `Fl::e_text`. Then, a few calls to functions whose name begin with `xkb_compose_` are necessary to support dead and compose keys. Finally a call to `Fl::handle()` sends an `FL_KEYDOWN` or `FL_KEYUP` event to the appropriate `Fl_Window`. Also, function `wl_keyboard_key()` uses global variable `Fl_Int_Vector key_vector` to record all currently pressed keys. This is the base of the implementation of `Fl_Wayland_Screen_Driver::event_key(int)`.

5) Function `wl_keyboard_modifiers()` runs when a modifier key (e.g., shift, control) is pressed or released. Calls to functions `xkb_state_update_mask()` and `xkb_state_mod_name_is_active()` allow FLTK to set `Fl::e_state` adequately.

6) Function `wl_keyboard_repeat_info()` does not run, for now, because this would require version 4 of the `wl_keyboard` object which is at version 2 in all tested Wayland compositors.

3.1.16 Support of text input methods

When the connected Wayland compositor supports text input methods, function `registry_handle_global()` gets called with its interface argument equal to `zwp_text_input_manager_v3` interface.name. The following call to `wl_registry_bind()` returns a pointer to type struct `zwp_text_input_manager_v3` that is stored as member `text_input_base` of the `Fl_Wayland_Screen_Driver` object.

Later, when function `seat_capabilities()` runs, `text_input_base` is found not NULL, which triggers a call to function `zwp_text_input_manager_v3_get_text_input()` returning a value of type struct `zwp_text_input_v3` * and stored as member `text_input` of the `Fl_Wayland_Screen_Driver::seat` object. Next, a call to `zwp_text_input_v3_add_listener()` associates this `text_input` with a 6-member listener of type struct `zwp_text_input_v3_listener`. These 6 FLTK-defined, callback functions are used as follows.

1-2) Functions `text_input_enter()` and `text_input_leave()` run when text input enters or leaves a surface.

3-4) Functions `text_input_preedit_string()` and `text_input_commit_string()` are called when the text input method prepares the client app to later insert 'marked' text or regular text, respectively. Complex text input often begins by inserting temporary text which is said to be 'marked' before replacing it with the text that will stay in the document. FLTK underlines marked text to distinguish it from regular text.

5) Function `text_input_done()` runs when it's time to send either regular or marked text to the client app. This is done by function `send_text_to_fltk()` which uses static variables `current_pre_edit`, `pending_pre_edit` and `pending_commit` to determine the sent text.

6) Function `text_input_delete_surrounding_text()` has no effect at present, without this preventing input methods that have been tested with FLTK from working satisfactorily.

It's necessary to inform the running text input method of the current location of the insertion point in the active surface. This information allows the input method to map its auxiliary window close to the insertion point. The flow of information on this topic is as follows:

- The two FLTK widgets supporting text input, `Fl_Input` and `Fl_Text_Display`, transmit to FLTK the window coordinates of the bottom of the current insertion point and the line height each time they change calling function `fl_set_spot()`.
- `fl_set_spot()` calls the platform override of virtual member function `Fl_Screen_Driver::set_spot()`. Under Wayland, this just calls `Fl_Wayland_Screen_Driver::insertion_point_location(int x, int y, int height)` which calls `zwp_text_input_v3_set_cursor_rectangle()` to inform the text input method about the surface position and size of the insertion point and also memorizes this information in static member variables of class `Fl_Wayland_Screen_Driver`.
- Callback function `text_input_enter()` calls `Fl_Wayland_Screen_Driver::insertion_point_location(int *x, int *y, int *height)` which gives it the stored position information, and then calls `zwp_text_input_v3_set_cursor_rectangle()` to inform the text input method about the position of the insertion point.

3.1.17 Interface with libdecor

FLTK uses a library called `libdecor` to determine whether the Wayland compositor uses CSD or SSD mode, and also to draw window titlebars when in CSD mode (see [libdecor](#)). `libdecor` is conceived to be present in a shared library linked to the Wayland client application which itself, and if the running Wayland compositor uses CSD mode, loads another shared library intended to draw titlebars in a way that best matches the Desktop. As of late 2023, `libdecor` contains two titlebar-drawing plugins:

- `libdecor-gtk` intended for the Gnome desktop;
- `libdecor-cairo` for other situations.

On recent Linux distributions, FLTK uses the system `libdecor` shared library available via packages `libdecor-0-dev` and `libdecor-0-plugin-1-gtk`. On earlier Linux versions, or if CMake option `FLTK_USE_SYSTEM_LIBDECOR` is set to OFF, FLTK bundles the most recent source code of `libdecor` and its plugins. The `libdecor` code bundled inside FLTK is compiled and put in `libfltk`. FLTK uses `libdecor-gtk` when software package `libgtk-3-dev` is present in the build system, and `libdecor-cairo` otherwise. FLTK prefixes all symbols of its bundled `libdecor` with `fl_`. This allows an FLTK client app to link to other libraries which may use the system version of `libdecor`.

`libdecor` uses the Wayland protocol `XDG decoration` to request being decorated by a supporting compositor. If the running compositor supports SSD, `libdecor` doesn't draw window titlebars because the compositor does it. That is what happens with the `KWin` and `Sway` compositors. However, if environment variable `LIBDECOR_FORCE_CSD` is defined to value 1 when an FLTK app runs, `libdecor` instructs an SSD-able compositor to refrain from decorating its windows and decorates windows itself.

Whatever the value of `FLTK_USE_SYSTEM_LIBDECOR`, FLTK and `libdecor` use environment variable `LIBDECOR_PLUGIN_DIR` as follows: if this variable is defined and points to the name of a directory, this directory is searched for a potential `libdecor` plugin in the form of a shared library; if one is found, FLTK and `libdecor` load it and use it.

The `libdecor` source code bundled in FLTK is identical to that of the `libdecor` repository. Nevertheless, FLTK uses this code with some minor changes. For example, except if `FLTK_USE_SYSTEM_LIBDECOR` is 1, FLTK needs to modify function `libdecor_new()` charged of loading the plugin, to make it use the plugin code that is included in `libfltk` if none is found as a dynamic library. This is done as follows in file `libdecor/build/fl_`

```
libdecor.c:
#define libdecor_new libdecor_new_orig
#include "../src/libdecor.c"
#undef libdecor_new

void libdecor_new() { // FLTK rewrite of this function
    .....
}
```

FLTK compiles file `fl_libdecor.c` which includes `libdecor.c` to the effect that all of the `libdecor` code becomes part of `libfltk` except that function `libdecor_new()` is substituted by its FLTK rewrite, without file `libdecor.c` being modified at all. This trick is also used to modify function `libdecor_frame_set_minimized()` to bypass a bug in the Weston compositor before version 10. Similarly, FLTK compiles file `fl_`
`libdecor-plugins.c` which includes either `libdecor-gtk.c` or `libdecor-cairo.c` to the effect that the desired plugin becomes part of `libfltk`.

To support function `Fl_Widget_Surface::draw_decorated_window()` that draws a mapped window and its titlebar, FLTK needs to perform two operations: 1) identify what plugin is operating, and 2) call a function that is specific of that plugin and that returns the pixels of the drawn titlebar.

FLTK performs operation 1) above using its function `get_libdecor_plugin_description()` of file `fl_`
`_libdecor-plugins.c` that returns a human readable string describing the running plugin. Each plugin puts its own string in member `description` of a record of type `struct libdecor_plugin_description`. Although this type is public in header file `libdecor-plugin.h`, accessing the symbol defined by the plugin to store a pointer to a value of this type is complicated for a reason and solved by a method detailed in a comment before function `get_libdecor_plugin_description()`.

Operation 2) above is done by FLTK-defined function `fl_libdecor_titlebar_buffer()` from file `fl_`
`_libdecor-plugins.c`. This function calls `get_libdecor_plugin_description()` seen above to get the running plugin's descriptive string. That is "GTK3 plugin" with `libdecor-gtk`. FLTK function `gtk_titlebar_buffer()` is then called, and returns a pointer to the start of a byte buffer containing the titlebar graphics. That is, again, not possible with the public `libdecor` API. Therefore, FLTK copies to `fl_libdecor-plugins.c` the definitions of several types given in `libdecor-gtk.c` or `libdecor-cairo.c` such as type `struct border_component`.

3.1.18 Copy/Paste/Drag-n-Drop

FLTK follows the procedure that is very well described in item "Wayland clipboard and drag & drop" of the [Documentation resources](#). All corresponding source code is in file `src/drivers/Wayland/fl_wayland_↵_clipboard_dnd.cxx`.

This part of the `Fl_Wayland_Screen_Driver::seat` record stores pointers to Wayland objects used for clipboard and D-n-D operations:

```
struct wl_data_device_manager *data_device_manager;
struct wl_data_device *data_device;
struct wl_data_source *data_source;
```

FLTK can copy or paste plain UTF-8 text or image data to/from the clipboard. Images are copied to the clipboard as image/bmp mime type. Images in image/bmp or image/png mime types from the clipboard can be pasted to FLTK apps.

Files dropped are received one pathname per line, with no '\n' after the last pathname.

3.1.19 EGL as support for OpenGL

Wayland uses [EGL™](#) to interface OpenGL with the underlying native platform window system. OpenGL-using FLTK apps are therefore linked to `libwayland-egl.so` and `libEGL.so` in addition to `libGL.so` and `libGLU.↵so`.

EGL completely hides the `wl_buffer` objects it uses to draw to GL windows. The `wld_buffer` structure and the 'buffer factory' described previously are not used for `Fl_Gl_Window`'s: the `buffer` member of an `Fl_Gl_Window`'s `wld_window` structure is always NULL.

EGL is initialized calling member function `Fl_Wayland_Gl_Window_Driver::init()` once, the first time the `Fl_Wayland_Gl_Window_Driver` c'tor runs. That is done with calls to `eglGetDisplay()`, `egl↵Initialize()`, and `eglBindAPI()`.

Member function `Fl_Wayland_Gl_Window_Driver::find()` calls `eglChooseConfig()` to filter the set of GL configurations that match the `Fl_Gl_Window`'s `mode()`, and puts in the returned `Fl_Gl_Choice` object the first matching configuration. The filtering gets done with bits `EGL_WINDOW_BIT`, to support the creation of window surfaces, and `EGL_OPENGL_BIT`, to support the creation of OpenGL contexts.

EGL needs 2 more objects created for each `Fl_Gl_Window`. They have types `struct wl_egl_window` and `EGLSurface`, and are created by member function `Fl_Wayland_Gl_Window_Driver::make_↵current_before()` which runs at the beginning of `Fl_Gl_Window::make_current()`. The first argument of the call to `wl_egl_window_create()` therein has type `struct wl_surface *` and is what connects EGL with the targeted Wayland window.

EGL creates with `eglCreateContext()` an object of type `EGLContext` via member function `Fl_↵Wayland_Gl_Window_Driver::create_gl_context()` called by `Fl_Gl_Window::make_current()`. Types `EGLContext` and `GLContext` are 2 names for the same object. The call to `eglCreateContext()` is made asking for a GL context of version at least 2. This does not prevent from obtaining contexts of higher versions, namely above 3.2, which are compatible with version 2 (the so-called compatibility profile) under all tested Linux systems.

FLTK function `Fl_Gl_Window::make_current()` calls overridden function `Fl_Wayland_Gl_Window_↵Driver::set_gl_context()` which calls EGL function `eglMakeCurrent()` when the cached context changes.

FLTK calls function `Fl_Wayland_Gl_Window_Driver::swap_buffers()` each time it wants a GL context to be sent to the display. This function contains some pure GL code to emulate an overlay buffer to support `Fl_Gl_Window` objects overriding their `draw_overlay()` member function. Then, it calls function `eglSwap_↵Buffers()`.

The overridden `Fl_Wayland_Gl_Window_Driver::resize()` function is implemented with calls to `wl_↵_egl_window_get_attached_size()` and `wl_egl_window_resize()`.

Class `Fl_Wayland_Gl_Plugin` exists to allow `libfltk` to call functions from `libfltk_gl`, `libwayland-egl.so` or `libEGL.so` and without having `libfltk` force linking any FLTK app with these GL-related libraries. For example, `Fl_Wayland_Window_Driver::flush()` needs to call `Fl_Gl_↵Window::valid(0)`.

Throttling GL window redraws

Although no documentation covering this subject was found, the EGL library internally uses `wl_callback` objects to throttle GL window redraws, and FLTK needs not interfere with these operations. Nevertheless FLTK creates and uses `wl_callback` objects for GL windows in 2 cases:

- when a decorated GL window is being interactively resized. Function `Fl_Wayland_Gl_Window_Driver::resize()` creates a `wl_callback` object, assigns it to `xid->frame_cb` and calls `wl_callback_add_listener()` before calling `wl_egl_window_resize()`. This allows the mechanism described above that prevents surfaces from being resized too frequently to operate with decorated `Fl_Gl_Window`'s too.
- when a GL subwindow is being refreshed by `Fl_Wayland_Gl_Window_Driver::swap_buffers()`. FLTK checks that `xid->frame_cb` is NULL and if so creates a `wl_callback` calling `wl_surface_frame()` before calling `eglSwapBuffers()`. This is useful if the GL subwindow becomes entirely out from the screen area. In that case, the Mutter compositor stops signaling that the subwindow is ready for new commits which FLTK detects because `xid->frame_cb` remains non-NULL. If the subwindow eventually re-appears partially on-screen, `xid->frame_cb` becomes NULL and FLTK calls `eglSwapBuffers()` to redraw the GL scene.

3.1.20 FLTK-defined, Wayland-specific types

struct `wld_window`

Defined in `Fl_Wayland_Window_Driver.H`. One such record is created for each `show()` `Fl_Window` by `Fl_Wayland_Window_Driver::makeWindow()`. Function `fl_wl_xid(Fl_Window*)` returns a pointer to the struct `wld_window` of its argument.

```
struct wld_window {
    Fl_Window *fl_win;
    struct wl_list outputs; // linked list of displays where part or whole of window maps
    struct wl_surface *wl_surface; // the window's surface
    struct wl_callback *frame_cb; // non-NULL until Wayland can process new surface commit
    struct Fl_Wayland_Graphics_Driver::wld_buffer *buffer; // see \ref wld_buffer
    struct xdg_surface *xdg_surface;
    enum Fl_Wayland_Window_Driver::kind kind; // DECORATED or POPUP or SUBWINDOW or UNFRAMED
    union {
        struct libdecor_frame *frame; // for DECORATED windows
        struct wl_subsurface *subsurface; // for SUBWINDOW windows
        struct xdg_popup *xdg_popup; // for POPUP windows
        struct xdg_toplevel *xdg_toplevel; // for UNFRAMED windows
    };
    struct Fl_Wayland_Window_Driver::custom_cursor {
        struct wl_cursor *wl_cursor;
        const Fl_RGB_Image *rgb;
        int hotx, hoty;
    } *custom_cursor; // non-null when using custom cursor
    int configured_width; // used when negotiating window size with the compositor
    int configured_height;
    int floating_width; // helps restoring size after un-maximizing
    int floating_height;
    int state; // indicates whether window is fullscreen, maximized. Used otherwise for POPUPS
    bool covered; // specially for Mutter and issue #878
}
```

struct `Fl_Wayland_Graphics_Driver::draw_buffer`

Defined in file `Fl_Wayland_Graphics_Driver.H`. One such record is created when an `Fl_Image_Surface` object is created. One such record is also embedded inside each struct `Fl_Wayland_Graphics_Driver::wld_buffer` record (see `wld_buffer`).

```
struct Fl_Wayland_Graphics_Driver::draw_buffer {
    unsigned char *buffer; // address of the beginning of the Cairo image surface's byte array
    cairo_t *cairo; // used when drawing to the Cairo image surface
    size_t data_size; // of buffer and wl_buffer, in bytes
    int stride; // bytes per line
    int width; // in pixels
};
```

FLTK gives offscreen buffers the platform-dependent type `Fl_Offscreen` which is in fact member `cairo_` of struct `Fl_Wayland_Graphics_Driver::draw_buffer`. Thus, a variable with type `Fl_Offscreen` needs be cast to type `cairo_t*`. Static member function `struct draw_buffer *offscreen_buffer(Fl_Offscreen)` of class `Fl_Wayland_Graphics_Driver` returns the `draw_buffer` record corresponding to an `Fl_Offscreen` value.

struct `Fl_Wayland_Graphics_Driver::wld_buffer`

Defined in file `Fl_Wayland_Graphics_Driver.H`. One such record is created by `Fl_Wayland_Graphics_Driver::create_wld_buffer()` when an `Fl_Window` is `show()` or resized, when a custom cursor shape is created, or when text is dragged.

```

struct Fl_Wayland_Graphics_Driver::wld_buffer {
    struct draw_buffer draw_buffer; // see draw_buffer
    struct wl_list link; // links all buffers from the same wl_shm_pool
    struct wl_buffer *wl_buffer; // the Wayland buffer
    void *data; // address of the beginning of the Wayland buffer's byte array
    struct wl_shm_pool *shm_pool; // pter to wl_shm_pool from which this wl_buffer comes
    bool draw_buffer_needs_commit; // true when draw_buffer has been modified but not yet committed
    bool in_use; // true while being committed
    bool released; // true after buffer_release() was called
};

```

struct Fl_Wayland_Screen_Driver::output

Defined in Fl_Wayland_Screen_Driver.H. One such record is created for each display of the system by function `registry_handle_global()` when it receives a "wl_output" interface. These records are kept in a linked list of them all, and an identifier of this linked list is stored in member `outputs` of the unique `Fl_Wayland_Screen_Driver` object `FLTK` uses. Thus,

```

Fl_Wayland_Screen_Driver *scr_driver = (Fl_Wayland_Screen_Driver*)Fl::screen_driver();
struct wl_list list_of_all_displays = scr_driver->outputs;

```

gives access, the Wayland way, to the linked list of displays in the system.

```

struct Fl_Wayland_Screen_Driver::output { // one record for each display
    uint32_t id; // an identifier of the display
    int x, y; // logical position of the top-left of display
    int width; // nber of horizontal pixels
    int height; // nber of vertical pixels
    float dpi; // at this point, always 96.
    struct wl_output *wl_output; // the Wayland object for this display
    int wld_scale; // Wayland scale factor
    float gui_scale; // FLTK scale factor
    bool done; // true means record members have been initialized
    struct wl_list link; // links these records together
};

```

It's possible to get the FLTK-defined record associated to a display from the Wayland-associated object for the same display, say `struct wl_output *wl_output`, by this call: `(struct Fl_Wayland_Screen_Driver::output *)wl_output_get_user_data(wl_output)`.

struct Fl_Wayland_Screen_Driver::seat

Defined in file `Fl_Wayland_Screen_Driver.H`. One record is created by function `registry_handle_global()` when it receives a "wl_seat" or `wl_data_device_manager_interface.name` interface. A pointer to this struct is stored in member `seat` of the client's unique `Fl_Wayland_Screen_Driver` object.

```

struct Fl_Wayland_Screen_Driver::seat {
    struct wl_seat *wl_seat;
    struct wl_pointer *wl_pointer;
    struct wl_keyboard *wl_keyboard;
    uint32_t keyboard_enter_serial;
    struct wl_surface *keyboard_surface;
    struct wl_list pointer_outputs;
    struct wl_cursor_theme *cursor_theme;
    struct wl_cursor *default_cursor;
    struct wl_surface *cursor_surface;
    struct wl_surface *pointer_focus;
    int pointer_scale;
    uint32_t serial;
    uint32_t pointer_enter_serial;
    struct wl_data_device_manager *data_device_manager;
    struct wl_data_device *data_device;
    struct wl_data_source *data_source;
    struct xkb_state *xkb_state;
    struct xkb_context *xkb_context;
    struct xkb_keymap *xkb_keymap;
    struct xkb_compose_state *xkb_compose_state;
    char *name;
    struct zwf_text_input_v3 *text_input;
};

```

3.1.21 Documentation resources

The Wayland book	Extensive introduction to Wayland programming written by the author of the sway compositor, unfortunately unachieved.	

Wayland Explorer	Documentation of all Wayland protocols, both stable and unstable. A language-independent syntax is used which makes function names usable from C or C++ not always obvious. Some useful functions seem undocumented here for an unclear reason.	
Wayland Protocol Specification	Documentation for all functions of the Wayland core protocol.	
Wayland clipboard and drag & drop	Detailed explanation of how clipboard and drag-and-drop work under Wayland.	
Wayland and input methods	Blog article introducing to the issue of text input methods under Wayland.	
Input Method Hub	Entry page for input method support giving newcomers a first understanding of what input methods are and how they are implemented in Wayland.	

3.2 Developer info for bundled libs

This chapter details the procedure to update the libraries which are bundled inside FLTK.

3.2.1 Introduction

This file is mainly intended for FLTK developers and contains information about the current versions of all bundled libraries and about how to upgrade these bundled libraries.

Starting with FLTK 1.4.0 the bundled libraries jpeg, png, and zlib use "symbol prefixing" with the prefix 'ftk_' for all external symbols to distinguish the bundled libraries from existing system libraries and to avoid runtime errors.

User code compiled correctly with the header files provided by the bundled image libraries need not be changed.

The nanosvg library is not affected.

3.2.2 Current status

Current versions of bundled libraries (as of March 9, 2025):

Library	Version/git commit	Release date	FLTK Version
jpeg	jpeg-9f	2024-01-14	1.4.0
nanosvg	7aeda550a8 [1]	2023-12-02	1.4.0
png	libpng-1.6.44	2024-09-12	1.4.1
zlib	zlib-1.3.1	2024-01-22	1.4.0
libdecor	2d5d80ab [2]	2025-03-06	1.5.0

Previous versions of bundled libraries (FLTK 1.3.x):

Library	Version	Release date	FLTK Version
jpeg	jpeg-9e	2022-01-16	1.3.9
png	libpng-1.6.40	2023-06-21	1.3.9
zlib	zlib-1.3	2023-08-18	1.3.9
jpeg	jpeg-9d	2020-01-12	1.3.6 - 1.3.8
png	libpng-1.6.37	2019-04-14	1.3.6 - 1.3.8
zlib	zlib-1.2.11	2017-01-15	1.3.6 - 1.3.8

[1] Git commit in branch 'ftk_' of <https://github.com/ftk/nanosvg>
See also git tag 'ftk_yyyy-mm-dd' where yyyy-mm-dd == "Release date"
and file nanosvg/README.txt.

[2] Git commit in <https://gitlab.freedesktop.org/libdecor/libdecor>

General information:

FLTK does not include the entire library distributions. We only provide the source files necessary to build the FLTK library and some README and/or CHANGELOG files. There are no test programs or other contributed files.

We use our own build files, hence a few files MUST NOT be upgraded when the library source files are upgraded. We strive to keep changes to the library source files as small as possible. Patching library code to work with FLTK should be a rare exception. Symbol prefixing with prefix 'ftk_' is one such exception to the rule.

If patches are necessary all changes in the library files should be marked with "FLTK" in a comment so a developer who upgrades the library later is aware of changes in the source code for FLTK. Look for 'FLTK' and/or 'fltk_' to find the differences.

Additional comments should be added to show the rationale, i.e. why a particular change was necessary. If applicable, add a reference to a Software Trouble Report, GitHub Issue or Pull Request (PR) like "STR 3456", "Issue #123", or "PR #234".

3.2.3 How to update the bundled libraries

It is generally advisable to use a graphical merge program. I'm using 'meld' under Linux, but YMMV.

Do not add any source files unless they are required to build the library.

Some config header files may be pre-generated in the FLTK sources. These header files should be left untouched, but it may be necessary to update these files if new items were added to the new library version. In this case the new header should be pre-generated on a Linux system with default options unless otherwise mentioned below for a specific library. Currently there are no known exceptions.

Merging source files:

Please check if some source and header files contain "FLTK" comments and/or 'fltk_' symbol prefixing to be aware of necessary merges. It is also good to download the distribution tar ball or Git source files of the previous version and to run a (graphical) diff or merge tool on the previous version and the bundled version of FLTK to see the "previous" differences.

Files that were not patched in previous versions should be copied to the new version w/o changes. Files that had FLTK specific patches must be merged manually. FLTK patches should be verified (if still necessary) and should be kept in the new source files.

Source and header files that have been added in the new library version should be added in FLTK as well if they are necessary to build the library. A simple "trial and error" should be sufficient to find files that need to be added. Added files must be added to FLTK's build files as well, usually to both 'Makefile' and 'CMakeLists.txt' to be used in configure/make and in CMake based builds, respectively.

Upgrade order:

There is only one dependency between all bundled libraries: libpng depends on zlib. Hence zlib should be upgraded first, then all other libs can be upgraded in arbitrary order.

Tests after merge:

Tests should be done on as many platforms as possible, both with autotools (configure/make) and CMake. Windows (Visual Studio) and macOS (Xcode) builds need CMake to generate the IDE files.

Upgrade notes for specific libraries:

The following chapters contain informations about specific files and how they are upgraded. Since the changes in all bundled libraries are not known in advance this information may change in the future. Please verify that no other changes are necessary.

3.2.4 zlib:

Website: <https://zlib.net/>

Download: See website and follow links.

Repository: git clone <https://github.com/madler/zlib.git>

zlib should be upgraded first because libpng depends on zlib.

Download the latest zlib sources, 'cd' to /path-to/zlib and run

```
$ ./configure --zprefix
```

This creates the header file 'zconf.h' with definitions to enable the standard 'z_' symbol prefix.

Unfortunately zlib requires patching some source and header files to convert this 'z_' prefix to 'fltk_z_' to be more specific. As of this writing (Nov. 2021) three files need symbol prefix patches:

- gzread.c

- zconf.h
- zlib.h

You may want to compare these files and/or the previous version to find out which changes are required. The general rule is to change all occurrences of 'z_' to 'ftk_z_' but there are exceptions.

The following files need special handling:

- CMakeLists.txt: Keep FLTK version, update manually if necessary.
- Makefile: Same as CMakeLists.txt.
- gzread.c: Merge changes (see above, manual merge recommended).
- zconf.h: Merge changes (see above, manual merge recommended).
- zlib.h: Merge changes (see above, manual merge recommended).
- makedepend: Keep this file.

Run 'make depend' in the zlib folder on a Linux system after the upgrade to update this file.

3.2.5 png:

Website: <http://libpng.org/pub/png/libpng.html>

Download: See website and follow links.

Repository: git clone <https://git.code.sf.net/p/libpng/code> libpng
libpng should be upgraded after zlib because it depends on zlib.

Download the latest libpng sources, 'cd' to /path-to/libpng and run

```
$ ./configure --with-libpng-prefix=ftk_
$ make
```

This creates the header files 'pnglibconf.h' and 'pngprefix.h' with the 'ftk_' symbol prefix.

The following files need special handling:

- CMakeLists.txt: Keep FLTK version, update manually if necessary.
- Makefile: Same as CMakeLists.txt.
- pnglibconf.h: Generate on a Linux system and merge (see above).
- pngprefix.h: Generate on a Linux system and merge (see above).
- makedepend: Keep this file.

Run 'make depend' in the png folder on a Linux system after the upgrade to update this file.

3.2.6 jpeg:

Website: <https://ijg.org/>

Download: See website and follow links.

Repository: N/A

Download the latest jpeg-xy sources on a Linux (or Unix) system, 'cd' to /path-to/jpeg-xy and run

```
$ ./configure
$ make [-jN]
```

This builds the library and should create the static library file '.libs/libjpeg.a'.

Execute the following command to extract the libjpeg symbol names used to build the 'prefixed' libftk_jpeg library:

```
$ nm --extern-only --defined-only .libs/libjpeg.a | awk '{print $3}' \
  | sed '/^$/d' | sort -u | awk '{print "#define \"$1\" ftk_\"$1\"}' \
  > ftk_jpeg_prefix.h
```

This creates the header file 'ftk_jpeg_prefix.h' with the '# define' statements using the 'ftk_' symbol prefix.

The following files need special handling:

- CMakeLists.txt: Keep FLTK version, update manually if necessary.
- Makefile: Same as CMakeLists.txt.

- fltk_jpeg_prefix.h: Generate on a Linux system and merge (see above).
- jconfig.h: keep changes flagged with

```
/* FLTK */
```

Note: more to come...

- makedepend: Keep this file.

Run 'make depend' in the jpeg folder on a Linux system after the upgrade to update this file.

3.2.7 nanosvg:

Website: <https://github.com/memononen/nanosvg>

Download: See website and follow links.

Repository: git clone <https://github.com/memononen/nanosvg.git>

FLTK Fork: git clone <https://github.com/fltk/nanosvg.git>

FLTK has its own GitHub fork of the original repository (see above).

The intention is to update this fork from time to time so the FLTK specific patches are up-to-date with the original library. Hopefully the FLTK patches will be accepted upstream at some time in the future so we no longer need our own patches. AlbrechtS, 04 Feb 2018.

Update (Feb 22, 2021): The upstream library is officially no longer maintained (see README.md) although updates appear from time to time.

Use this fork (branch 'fltk') to get the nanosvg library with FLTK specific patches:

```
$ git clone https://github.com/fltk/nanosvg.git nanosvg-fltk
$ cd nanosvg-fltk
$ git checkout fltk
$ cd src
$ cp nanosvg.h nanosvgrast.h /path/to/fltk-1.4/nanosvg/
```

This library does not have its own build files since it is a header-only library. The headers are included in FLTK where necessary.

The following files need special handling:

nanosvg.h: Merge or download from FLTK's fork (see above).

nanosvgrast.h: Merge or download from FLTK's fork (see above).

Maintaining branch 'fltk' in FLTK's fork of nanosvg (fltk/nanosvg):

Only maintainers with write access on fltk/nanosvg can do this. Others can fork our fltk/nanosvg fork in their own GitHub account and either open a PR on fltk/nanosvg or tell us about their changes in fltk.development.

Use something similar to the following commands to update FLTK's fork of nanosvg to the latest version. Commands are only examples, you may need to change more or less, depending on the outstanding updates.

Step 1: clone the fltk/nanosvg fork, set the remote 'upstream', and update the 'master' branch:

```
$ cd /to/your/dev/dir
$ git clone https://github.com/fltk/nanosvg.git nanosvg-fltk
$ cd nanosvg-fltk
$ git remote add upstream https://github.com/memononen/nanosvg
$ git checkout master
$ git pull upstream master
```

Note: the 'master' branch must never be changed, i.e. it must always be the same as 'upstream/master'. Never commit your own (FLTK specific) changes to branch 'master'.

Step 2: rebase branch 'fltk' on the new master (upstream/master), fix potential conflicts, and tag the new branch.

It is important to keep the individual FLTK specific patches intact (one commit per patch) because this will preserve the history and the committer and make it easier to skip single patches when they are accepted upstream.

```
$ git checkout fltk
$ git rebase upstream/master
```

At this point you may need to fix conflicts! Do whatever is necessary to update the branch 'fltk'.

Now 'git tag' the 'fltk' branch for later reference.

Hint: use 'git show <any-older-tag-name>' to see its contents. I like to write a summary of commits in the tag comment.

```
$ git tag -a fltk_yyyy-mm-dd fltk
```

Replace 'yyyy-mm-dd' with the current date and add a comment when asked for it (your editor will open an empty file).

Step 3: at this point it is recommended to copy the changed header files to your working copy of the FLTK library and test the changes. If anything is wrong, go back, fix the bugs and change the git tag (delete and create a new one).

Step 4: push the new branch 'fltk' and the tag to the fltk/nanosvg repository:

```
$ git push -f origin fltk
$ git push origin fltk_yyyy-mm-dd
```

Step 5: copy the changed files to your working copy of the FLTK repository (if not done already), update this file accordingly, and commit/push the update to the fltk/fltk repository.

3.2.8 libdecor:

Website: <https://gitlab.freedesktop.org/libdecor/libdecor>

Download: See website and follow links.

Repository: git clone <https://gitlab.freedesktop.org/libdecor/libdecor.git>

libdecor is used by the Wayland/X11 hybrid platform to draw window titlebars when FLTK apps run as Wayland clients and the running Wayland compositor uses client-side decoration. In the future, when libdecor will have made its way into Linux packages, FLTK will use the system version of libdecor. libdecor will remain as an FLTK bundle to support Linux configurations where the libdecor package is not available or not installed.

FLTK uses libdecor source files without any modification. This part of the libdecor source tree is copied to directory libdecor/ of the FLTK source tree:

```
LICENSE
README.md
src/      ... and files below except meson.build files
```

Furthermore, directory libdecor/build/ of the FLTK source tree does not originate from the libdecor source tree but contains 3 FLTK-created files. File build/Makefile may need changes if a libdecor update adds or renames source files.

3.3 Developer Information

This chapter describes FLTK development and documentation.

Example

```
/** \file
    Fl_Clock, Fl_Clock_Output widgets. */

/**
    \class Fl_Clock_Output
    \brief This widget can be used to display a program-supplied time.

    The time shown on the clock is not updated. To display the current time,
    use Fl_Clock instead.

    \image html clock.png
    \image latex clock.png "" width=10cm
    \image html round_clock.png
    \image latex clock.png "" width=10cm
    \image html round_clock.png "" width=10cm */

/**
    Returns the displayed time.
    Returns the time in seconds since the UNIX epoch (January 1, 1970).
    \see value(ulong)
    */
ulong value() const {return value_;}

/**
    Set the displayed time.
    Set the time in seconds since the UNIX epoch (January 1, 1970).
    \param[in] v seconds since epoch
    \see value()
    */
void Fl_Clock_Output::value(ulong v) {
    [...]
}

/**
    Create an Fl_Clock widget using the given position, size, and label string.
```

```

The default boxtype is \c FL_NO_BOX.
\param[in] X, Y, W, H position and size of the widget
\param[in] L widget label, default is no label
*/
Fl_Clock::Fl_Clock(int X, int Y, int W, int H, const char *L)
    : Fl_Clock_Output(X, Y, W, H, L) {}

/**
Create an Fl_Clock widget using the given boxtype, position, size, and
label string.
\param[in] t boxtype
\param[in] X, Y, W, H position and size of the widget
\param[in] L widget label, default is no label
*/
Fl_Clock::Fl_Clock(uchar t, int X, int Y, int W, int H, const char *L)
    : Fl_Clock_Output(X, Y, W, H, L) {
    type(t);
    box(t==FL_ROUND_CLOCK ? FL_NO_BOX : FL_UP_BOX);
}

```

Note

From Duncan: (will be removed later, just for now as a reminder)

I've just added comments for the `fl_color_chooser()` functions, and in order to keep them and the general Function Reference information for them together, I created a new doxygen group, and used `\ingroup` in the three comment blocks. This creates a new Modules page (which may not be what we want) with links to it from the File Members and [Fl_Color_Chooser.H](#) pages. It needs a bit more experimentation on my part unless someone already knows how this should be handled. (Maybe we can add it to a `functions.dox` file that defines a functions group and do that for all of the function documentation?)

Update: the trick is not to create duplicate entries in a new group, but to move the function information into the doxygen comments for the class, and use the navigation links provided. Simply using `\relatesalso` as the first doxygen command in the function's comment puts it in the appropriate place. There is no need to have `\defgroup` and `\ingroup` as well, and indeed they don't work. So, to summarize:

```

Gizmo.H
/** \class Gizmo
    A gizmo that does everything
 */
class Gizmo {
    etc
};
extern int popup_gizmo(...);

Gizmo.cxx:
/** \relatesalso Gizmo
    Pops up a gizmo dialog with a Gizmo in it
 */
int popup_gizmo(...);

```

Comments Within Doxygen Comment Blocks

You can use HTML comment statements to embed comments in doxygen comment blocks. These comments will not be visible in the generated document.

```

The following text is a developer comment.
<!-- *** This *** is *** invisible *** -->
This will be visible again.

```

will be shown as:

```

The following text is a developer comment.
This will be visible again.

```

Note

Since an unknown Doxygen version indentation by four or more bytes is automatically displayed as if it was framed by `\code` and `\endcode`. You need to take care that "normal" text is not indented that much to avoid mis-interpretation and formatting as code.

Different Headlines

You can use HTML tags `<H1> ... <H4>` for headlines with different sizes. As of doxygen 1.8.x there must not be more than three spaces at the beginning of the line for this to work. Currently (doxygen 1.8.6) there seems to be no difference in the font sizes of `<H3>` and `<H4>` in the pdf output, whereas the html output uses different font sizes.

```
<H1>Headline in big text (H1)</H1>
<H2>Headline in big text (H2)</H2>
<H3>Headline in big text (H3)</H3>
<H4>Headline in big text (H4)</H4>
```

Headline in big text (H1)

Headline in big text (H2)

Headline in big text (H3)

Headline in big text (H4)

3.3.1 Non-ASCII Characters

Doxygen understands many HTML quoting characters like `"`, `ü`, `ç`, `Ç`, but not all HTML quoting characters.

This will appear in the document:

Doxygen understands many HTML quoting characters like ", ü, ç, Ç, but not all HTML quoting characters.

For further informations about HTML quoting characters see

<http://www.doxygen.org/manual/htmlcmds.html>

Alternatively you can use **UTF-8** encoding within Doxygen comments.

3.3.2 Document Structure

- `\page` creates a named page
- `\section` creates a named section within that page
- `\subsection` creates a named subsection within the current section
- `\subsubsection` creates a named subsubsection within the current subsection

All these statements take a "name" as their first argument, and a title as their second argument. The title can contain spaces.

The page, section, and subsection titles are formatted in blue color and a size like "`<H1>`", "`<H2>`", and "`<H3>`", and "`<H4>`", respectively.

By **FLTK documentation convention**, a file like this one with a doxygen documentation chapter has the name "`<chapter>.dox`". The `\page` statement at the top of the page is "`\page <chapter> This is the title`". Sections within a documentation page must be called "`<chapter>_<section>`", where "`<chapter>`" is the name part of the file, and "`<section>`" is a unique section name within the page that can be referenced in links. The same for subsections and subsubsections.

These doxygen page and section commands work only in special documentation chapters, not within normal source or header documentation blocks. However, links **from** normal (e.g. class) documentation **to** documentation sections **do work**.

This page has

```
\page development Developer Information
```

at its top.

This section is

```
\section development_structure Document Structure
```

The following section is

```
\section development_links Creating Links
```

3.3.3 Creating Links

Links to other documents and external links can be embedded with

- doxygen `\ref` links to other doxygen `\page`, `\section`, `\subsection` and `\anchor` locations
- HTML links without markup - doxygen creates "`http://...`" links automatically

- standard, non-Doxygen, HTML links
- see chapter `\ref unicode` creates a link to the named chapter unicode that has been created with a `\\page` statement.
- For further informations about quoting see <http://www.doxygen.org/manual/htmlcmds.html>
- see `<a href="https://www.fltk.org/">FLTK Library</a>` creates a standard HTML link

appears as:

- see chapter [Unicode and UTF-8 Support](#) creates a link to the named chapter unicode that has been created with a `\page` statement.
- For further informations about quoting see <http://www.doxygen.org/manual/htmlcmds.html>
- see [FLTK Library](#) creates a standard HTML link

3.3.4 Paragraph Layout

There is no real need to use HTML `<P>` and `</P>` tags within the text to tell doxygen to start or stop a paragraph. In most cases, when doxygen encounters a blank line or some, but not all, `\commands` in the text it knows that it has reached the start or end of a paragraph. Doxygen also offers the `\par` command for special paragraph handling. It can be used to provide a paragraph title and also to indent a paragraph. Unfortunately `\par` won't do what you expect if you want to have doxygen links and sometimes html tags don't work either.

```
\par Normal Paragraph with title
```

This paragraph will have a title, but because there is a blank line between the `\par` and the text, it will have the normal layout.

```
\par Indented Paragraph with title
```

This paragraph will also have a title, but because there is no blank line between the `\par` and the text, it will be indented.

```
\par
```

It is also possible to have an indented paragraph without title. This is how you indent subsequent paragraphs.

```
\par No link to Fl_Widget::draw()
```

Note that the paragraph title is treated as plain text.

Doxygen type links will not work.

HTML characters and tags may or may not work.

```
Fl_Widget::draw() links and &quot;html&quot; tags work<br>
```

```
\par
```

Use a single line ending with `<br>` for complicated paragraph titles.

The above code produces the following paragraphs:

Normal Paragraph with title

This paragraph will have a title, but because there is a blank line between the `\par` and the text, it will have the normal layout.

Indented Paragraph with title

This paragraph will also have a title, but because there is no blank line between the `\par` and the text, it will be indented.

It is also possible to have an indented paragraph without title. This is how you indent subsequent paragraphs.

No link to `Fl_Widget::draw()`

Note that the paragraph title is treated as plain text. Doxygen type links will not work. HTML characters and tags may or may not work.

[Fl_Widget::draw\(\)](#) links and "html" tags work

Use a single line ending with `<br>` for complicated paragraph titles.

3.3.5 Navigation Elements

Each introduction (tutorial) page ends with navigation elements. These elements must only be included in the html documentation, therefore they must be separated with `\htmlonly` and `\endhtmlonly`.

The following code creates the navigation bar at the bottom of the [FLTK Basics](#) HTML page as an example of all pages in the "FLTK Programming Manual" section. Note that **this page** is one of the appendices w/o navigation bar.

```
\htmlonly


---



|                                                                                   |                                             |                                                                                             |
|-----------------------------------------------------------------------------------|---------------------------------------------|---------------------------------------------------------------------------------------------|
| <a class="el" href="intro.html">       [Prev]       Introduction to FLTK     </a> | <a class="el" href="index.html">[Index]</a> | <a class="el" href="common.html">       Common Widgets and Attributes       [Next]     </a> |
|-----------------------------------------------------------------------------------|---------------------------------------------|---------------------------------------------------------------------------------------------|


\endhtmlonly
```

Chapter 4

Todo List

Page [Adding and Extending Widgets](#)

Clarify [FI_Window::damage\(uchar\)](#) handling - seems confused/wrong? ORing value doesn't match setting behavior in [FI_Widget.H](#)!

Member [FI::now](#) (double offset=0)

[FI::system_driver\(\)](#)->gettime() was implemented for the Forms library and has a limited resolution (on Windows: milliseconds). On POSIX platforms it uses gettimeofday() with microsecond resolution. A new function could use a better resolution on Windows with its multimedia timers which requires a new dependency: winmm.lib (dll). This could be a future improvement, maybe set as a build option or generally (requires Win95 or 98?).

Member [FI_Browser::scrollbar_width](#) () const

This method should eventually be removed in 1.4+

Member [FI_Browser::scrollbar_width](#) (int width)

This method should eventually be removed in 1.4+

Class [FI_Chart](#)

Refactor [FI_Chart::type\(\)](#) information.

Member [FI_File_Input::errorcolor](#) () const

Remove [FI_File_Input::errorcolor\(\)](#) in FLTK 1.5.0 or higher.

Member [FI_File_Input::errorcolor](#) (FI_Color c)

Remove [FI_File_Input::errorcolor\(FI_Color\)](#) in FLTK 1.5.0 or higher.

Member [fl_filename_list](#) (const char *d, struct dirent ***l, FI_File_Sort_F *s=fl_numeric_sort)

should support returning OS error messages

Class [FI_Grid](#)

This (relative group coordinates of nested groups of [FI_Grid](#)) needs explanation and maybe an example.

Member [FI_Grid::Cell::~~Cell](#) ()

[FI_Grid](#)'s cell destructor should remove the cell from the grid. Currently it does nothing!

Member [FI_Grid::clear_layout](#) ()

[FI_Grid::clear\(\)](#) needs to be implemented as documented above!

Member `FI_Grid::debug` (int level=127)

Add more information about cells and children.

Control output by using `level`.

Member `FI_Grid::FI_Grid` (int X, int Y, int W, int H, const char *L=0)

More documentation of `FI_Grid` constructor?

Member `FI_Grid::layout` (int rows, int cols, int margin=-1, int gap=-1)

Document when and why to call `layout()` w/o args. See `FI_Flex::layout()`

Member `FI_Grid::layout` ()

Document when and why to call `layout()` w/o args. See `FI_Flex::layout()`

Member `FI_Group::delete_child` (int n)

Reimplementation of `FI_Group::delete_child(int)` in more FLTK subclasses. This is not yet complete.

Member `fl_height` (int font, int size)

In the future, when the XFT issues are resolved, this function should simply return the 'size' value.

Member `FI_Input::handle_mouse` (int, int, int, int, int keepmark=0)

Add comment and parameters

Member `FI_Input::handletext` (int e, int, int, int, int)

Add comment and parameters

Struct `FI_Label`

There is an aspiration that the `FI_Label` type will become a widget by itself. That way we will be avoiding a lot of code duplication by handling labels in a similar fashion to widgets containing text. We also provide an easy interface for very complex labels, containing html or vector graphics. However, this re-factoring is not in place in this release.

Member `FI_Menu::add` (const char *, int shortcut, FI_Callback *, void *=0, int=0)

Raw integer shortcut needs examples. Dependent on responses to <https://www.fltk.org/newsgroups.php?gfltk.coredev+v:10086> and results of STR#2344

Member `FI_Shortcut`

Discuss and decide whether we can "shift" these special keyboard flags to the upper byte to enable full 21-bit Unicode characters (U+0000 . . . U+10FFFF) plus the keyboard indicator bits as this was originally intended. This would be possible if we could rely on **all** programs being coded with symbolic names and not hard coded bit values.

Member `FI_Terminal::scrollbar`

Support scrollbar_left/right() - See `FI_Browser::scrollbar` docs

Support new ScrollbarStyle

Member `FI_Text_Display::extend_range_for_styles` (int *start, int *end)

Unicode?

Member `FI_Text_Display::handle_vline` (int mode, int lineStart, int lineLen, int leftChar, int rightChar, int topClip, int bottomClip, int leftClip, int rightClip) const

we need to handle hidden hyphens and tabs here!

we handle all styles and selections

we must provide code to get pixel positions of the middle of a character as well

Member `FI_Text_Display::overstrike` (const char *text)

Unicode? Find out exactly what we do here and simplify.

Member `FI_Text_Display::position_to_linecol` (int pos, int *lineNum, int *column) const

a column number makes little sense in the UTF-8/variable font width environment. We will have to further define what exactly we want to return. Please check the functions that call this particular function.

Member `FI_Text_Display::scroll` (int topLineNum, int horizOffset)

Column numbers make little sense here.

Member `FI_Text_Display::scrollbar_width` () const

This method should eventually be removed.

Member `FI_Text_Display::scrollbar_width` (int width)

This method should eventually be removed

Member `FI_Text_Display::shortcut` () const

FIXME : get set methods pointing on shortcut_ have no effects as shortcut_ is unused in this class and derived!

Member `FI_Text_Display::shortcut` (int s)

FIXME : get set methods pointing on shortcut_ have no effects as shortcut_ is unused in this class and derived!

Member `FI_Text_Display::wrapped_column` (int row, int column) const

What does this do and how is it useful? Column numbers mean little in this context. Which functions depend on this one? Function TextDXYToUnconstrainedPosition does not exist (nedit port?)

Unicode?

Member `FI_Text_Display::wrapped_row` (int row) const

What does this do and how is it useful? Column numbers mean little in this context. Which functions depend on this one? Function TextDXYToUnconstrainedPosition does not exist (nedit port?)

Member `FI_Tiled_Image::FI_Tiled_Image` (FI_Image *i, int W=0, int H=0)

Fix `FI_Tiled_Image` as background image for widgets and windows and fix the implementation of `FI::scheme(const char *)`.

Member `FI_Tree::handle` (int e) FL_OVERRIDE

add `FI_Widget_Tracker` (see `FI_Browser_.cxx::handle()`)

Member `FI_Tree::is_scrollbar` (FI_Widget *w)

should be const

Member [Fl_Tree::show_self\(\)](#)

should be const

Member [Fl_Window::show\(\)](#) `FL_OVERRIDE`

Check if we can remove resetting the current group in a later FLTK version (after 1.3.x). This may break "already broken" programs though if they rely on this "feature".

Page [FLTK Basics](#)

This section needs a major rework. Add a chapter "Building FLTK with CMake".

Page [Handling Events](#)

Add details on how to detect repeating keys, since on some X servers a repeating key will generate both `FL_KEYUP` and `FL_KEYDOWN`, such that to tell if a key is held, you need [Fl::event_key\(int\)](#) to detect if the key is being held down during `FL_KEYUP` or not.

Page [Unicode and UTF-8 Support](#)

FLTK 1.3 and later supports the full Unicode range (21 bits), but there are a few exceptions, for instance binary shortcut values in menus ([Fl_Shortcut](#)) can only be used with characters from the BMP (16 bits). This may be extended in a future FLTK version.

Work through the code and this documentation to harmonize the `[OksiD]` and `[fltk2]` functions.

Verify 16/24 bit Unicode limit for different character sets? OksiD's code appears limited to 16-bit whereas the FLTK2 code appears to handle a wider set. What about illegal characters? See comments in `fl_utf8fromwc()` and `fl_utf8toUtf16()`.

Chapter 5

Deprecated List

Member `Fl::release ()`

Use `Fl::grab(0)` instead.

Member `Fl::set_idle (Fl_Old_Idle_Handler cb)`

This method is obsolete - use the `add_idle()` method instead.

Member `Fl::version ()`

Use `int Fl::api_version()` instead.

Member `fl_ask (const char *fmt,...)`

`fl_ask()` is deprecated since it uses "Yes" and "No" for the buttons which does not conform to the current FLTK Human Interface Guidelines. Use `fl_choice()` with the appropriate verbs instead.

Member `Fl_Browser::position () const`

"since 1.4.0 - use `vposition()` instead"

Member `Fl_Browser::position (int pos)`

"since 1.4.0 - use `vposition(pos)` instead"

Member `Fl_Browser::scrollbar_width () const`

Use `scrollbar_size()` instead.

Member `Fl_Browser::scrollbar_width (int width)`

Use `scrollbar_size()` instead.

Member `fl_clip (int x, int y, int w, int h)`

Please use `fl_push_clip(int x, int y, int w, int h)` instead. `fl_clip(int, int, int, int)` will be removed in FLTK 1.↔5.

Member `Fl_File_Input::errorcolor () const`

Will be removed in FLTK 1.5.0 or higher.

Member `Fl_File_Input::errorcolor (Fl_Color c)`

Will be removed in FLTK 1.5.0 or higher.

Member `fl_find` (Window `xid`)

Kept in the X11, Windows, and macOS platforms for compatibility with FLTK versions before 1.4. Please use `fl_x11_find(Window)`, `fl_wl_find(struct wld_window*)`, `fl_win32_find(HWND)` or `fl_mac_find(FLWindow*)` with FLTK 1.4.0 and above.

Member `FI_GIF_Image::FI_GIF_Image` (const char *`imagename`, const unsigned char *`data`)

Please use `FI_GIF_Image(const char *imagename, const unsigned char *data, const size_t length)` instead.

Member `FI_Group::focus` (FI_Widget *`W`)

This is for backwards compatibility only. You should use `W->take_focus()` instead.

Member `FI_Group::sizes` ()

Deprecated since 1.4.0. Please use `bounds()` instead.

Member `FI_Image::draw_scaled` (int `X`, int `Y`, int `W`, int `H`)

Only for API compatibility with FLTK 1.3.4.

Member `FI_Image::label` (FI_Widget *`w`)

Please use `FI_Widget::image()` or `FI_Widget::deimage()` instead.

Member `FI_Image::label` (FI_Menu_Item *`m`)

Please use `FI_Menu_Item::image()` instead.

Member `FI_Image_Surface::highres_image` ()

Use `image()` instead.

Member `FI_Input::position` () const

"since 1.4.0 - use `insert_position()` instead"

Member `FI_Input::position` (int `p`, int `m`)

"since 1.4.0 - use `insert_position(p, m)` or `FI_Widget::position(x, y)` instead"

Member `FI_Input::position` (int `p`)

"since 1.4.0 - use `insert_position(p)` instead"

Member `FI_Menu_Item::check` ()

Please use `FI_Menu_Item::set()` instead. This method will be removed in FLTK 1.5.0 or later.

Member `FI_Menu_Item::checked` () const

Please use `FI_Menu_Item::value()` instead. This method will be removed in FLTK 1.5.0 or later.

Member `FI_Menu_Item::uncheck` ()

Please use `FI_Menu_Item::clear()` instead. This method will be removed in FLTK 1.5.0 or later.

Member `FI_Multi_Label::label` (FI_Menu_Item *)

since 1.4.0: please use `FI_Menu_Item::label(FI_Multi_Label *)`

Member `FI_Preferences::FI_Preferences` (const char *`path`, const char *`vendor`, const char *`application`)

"since 1.4.0 - use `FI_Preferences(path, vendor, application, flags)` instead"

Member `FI_Text_Display::scrollbar_width ()` const

Use `scrollbar_size()` instead.

Member `FI_Text_Display::scrollbar_width (int width)`

Use `scrollbar_size()` instead.

Member `FI_Text_Selection::position (int *startpos, int *endpos)` const

"since 1.4.0 - use `selected(startpos, endpos)` instead"

Member `FI_Tile::position (int oldx, int oldy, int newx, int newy)`

"since 1.4.0 - use `move_intersection(p)` instead"

Member `FI_Tree::first_visible ()`

in 1.3.3 ABI – use `first_visible_item()` instead.

Member `FI_Tree::item_clicked (FI_Tree_Item *val)`

in 1.3.3 ABI – use `callback_item()` instead.

Member `FI_Tree::item_clicked ()`

in 1.3.3 ABI – use `callback_item()` instead.

Member `FI_Tree::last_visible ()`

in 1.3.3 – use `last_visible_item()` instead.

Member `FI_Tree_Item::FI_Tree_Item (const FI_Tree_Prefs &prefs)`

in 1.3.3 ABI – you must use `FI_Tree_Item(FI_Tree*)` for proper horizontal scrollbar behavior.

Member `FI_Tree_Item::next_displayed (FI_Tree_Prefs &prefs)`

in 1.3.3 for confusing name, use `next_visible()` instead

Member `FI_Tree_Item::prev_displayed (FI_Tree_Prefs &prefs)`

in 1.3.3 for confusing name, use `prev_visible()`

Member `FL_VERSION`

This `double` version number is retained for compatibility with existing program code. New code should use `int FL_API_VERSION` instead. `FL_VERSION` is deprecated because comparisons of floating point values may fail due to rounding errors. However, there are currently no plans to remove this deprecated constant.

Member `FI_Widget::color2 ()` const

Use `selection_color()` instead.

Member `FI_Widget::color2 (unsigned a)`

Use `selection_color(unsigned)` instead.

Member `FI_Window::free_position ()`

please use `force_position(0)` instead

Member `FI_Window::icon ()` const

in 1.3.3

Member [Fl_Window::icon](#) (const void *ic)

in 1.3.3 in favor of platform-independent methods [Fl_Window::icon\(const Fl_RGB_Image *icon\)](#) and [Fl_Window::icons\(const Fl_RGB_Image *icons\[\], int count\)](#).

Page [Operating System Issues](#)

Kept for compatibility with FLTK versions before 1.4. Use preferentially [fl_x11_xid\(const Fl_Window *\)](#) with versions 1.4 and above.

Kept for compatibility with FLTK versions before 1.4. Use preferentially [fl_x11_find\(Window\)](#) with versions 1.4 and above.

Chapter 6

Topic Index

6.1 Topics

Here is a list of all topics with brief descriptions:

Callback Function Typedefs	231
Windows handling functions	232
Events handling functions	234
Selection & Clipboard functions	250
Screen functions	254
Color & Font functions	261
Drawing functions	275
Multithreading support functions	311
Safe widget deletion support functions	313
Cairo Support Functions and Classes	315
Unicode and UTF-8 functions	318
String handling functions	334
Mac OS X-specific symbols	335
Common Dialog Classes and Functions	336
File names and URI utility functions	350

Chapter 7

Hierarchical Index

7.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

FI_Grid::Cell	363
FI_Terminal::CharStyle	364
FI_GIF_Image::GIF_FRAME::CPAL	365
FI_Terminal::Cursor	365
FI_Preferences::Entry	365
FI_Terminal::EscapeSeq	366
FI	366
FI_Cairo_State	507
FI_Callback_User_Data	521
FL_CHART_ENTRY	535
FI_End	645
FI_File_Chooser	664
FI_File_Icon	671
FI_Gl_Choice	769
FI_Glut_Bitmap_Font	790
FI_Glut_StrokeChar	790
FI_Glut_StrokeFont	790
FI_Glut_StrokeStrip	791
FI_Glut_StrokeVertex	791
FI_Help_Block	848
FI_Help_Dialog	848
FI_Help_Font_Stack	850
FI_Help_Font_Style	851
FI_Help_Link	851
FI_Help_Target	852
FI_Image	919
FI_Bitmap	423
FI_XBM_Image	1973
FI_Pixmap	1195
FI_GIF_Image	764
FI_Anim_GIF_Image	408
FI_XPM_Image	1976
FI_RGB_Image	1313
FI_BMP_Image	428
FI_ICO_Image	915
FI_JPEG_Image	998
FI_PNG_Image	1206
FI_PNM_Image	1209
FI_SVG_Image	1489
FI_Shared_Image	1409

FI_Tiled_Image	1729
FI_Image_Reader	929
FI_Label	1002
FI_Mac_App_Menu	1021
FI_Menu_Item	1068
FI_Multi_Label	1108
FI_Native_File_Chooser	1132
FI_Plugin	1200
FI_Device_Plugin	619
FI_Preferences	1230
FI_Plugin_Manager	1201
FI_Rect	1292
FI_Scroll::FI_Region_LRTB	1295
FI_Scroll::FI_Region_XYWH	1295
FI_Scheme	1346
FI_Scroll::FI_Scrollbar_Data	1384
FI_Surface_Device	1483
FI_Display_Device	630
FI_Widget_Surface	1927
FI_Copy_Surface	604
FI_EPS_File_Surface	645
FI_Image_Surface	929
FI_Paged_Device	1182
FI_PDF_File_Surface	1188
FI_PostScript_File_Device	1221
FI_Printer	1252
FI_SVG_File_Surface	1485
FI_Text_Buffer	1620
FI_Text_Selection	1708
FI_Timeout	1734
FI_Tooltip	1757
FI_Tree_Item	1810
FI_Tree_Item_Array	1832
FI_Tree_Prefs	1835
FI_Widget	1884
FI_Box	433
FI_Button	495
FI_Light_Button	1004
FI_Check_Button	551
FI_Radio_Light_Button	1276
FI_Round_Button	1329
FI_Radio_Round_Button	1284
FI_Radio_Button	1268
FI_Repeat_Button	1296
FI_Return_Button	1304
FI_Shortcut_Button	1419
FI_Toggle_Button	1749
FI_Chart	522
FI_Clock_Output	581
FI_Clock	571
FI_Round_Clock	1337
FI_FormsBitmap	733
FI_FormsPixmap	740
FI_FormsText	748
FI_Free	755
FI_Group	828

FI_Browser_	471
FI_Browser	441
FI_File_Browser	649
FI_Hold_Browser	867
FI_Multi_Browser	1095
FI_Select_Browser	1395
FI_Check_Browser	536
FI_Color_Chooser	591
FI_Flex	705
FI_Grid	806
FI_Help_View	852
FI_Input_Choice	975
FI_Pack	1171
FI_Scroll	1358
FI_Spinner	1460
FI_Table	1512
FI_Table_Row	1539
FI_Tabs	1555
FI_Terminal	1577
FI_Text_Display	1637
FI_Text_Editor	1685
FI_Tile	1712
FI_Tree	1762
FI_Window	1932
FI_Double_Window	631
FI_Cairo_Window	507
FI_Overlay_Window	1157
FI_Gl_Window	769
FI_Glut_Window	791
FI_Single_Window	1437
FI_Menu_Window	1082
FI_Wizard	1964
FI_Input_	948
FI_Input	935
FI_File_Input	676
FI_Float_Input	723
FI_Int_Input	988
FI_Multiline_Input	1111
FI_Output	1146
FI_Multiline_Output	1121
FI_Secret_Input	1385
FI_Spinner::FI_Spinner_Input	1473
FI_Menu_	1023
FI_Choice	559
FI_Scheme_Choice	1347
FI_Menu_Bar	1046
FI_Sys_Menu_Bar	1497
FI_Menu_Button	1057
FI_Positioner	1212
FI_Progress	1260
FI_Timer	1740
FI_Valuator	1840
FI_Adjuster	398
FI_Counter	608
FI_Simple_Counter	1428
FI_Dial	620

FI_Fill_Dial	688
FI_Line_Dial	1013
FI_Roller	1320
FI_Slider	1450
FI_Fill_Slider	697
FI_Hor_Fill_Slider	881
FI_Hor_Nice_Slider	889
FI_Hor_Slider	898
FI_Nice_Slider	1138
FI_Scrollbar	1373
FI_Value_Slider	1872
FI_Hor_Value_Slider	906
FI_Value_Input	1850
FI_Value_Output	1862
FI_Widget_Tracker	1930
FI_GIF_Image::GIF_FRAME	1979
FI_ICO_Image::IconDirEntry	1979
FI_Text_Editor::Key_Binding	1980
FI_Terminal::Margin	1980
FI_Preferences::Name	1980
FI_Preferences::Node	1981
FI_Paged_Device::page_format	1982
FI_Terminal::PartialUtf8Buf	1983
FI_Terminal::RingBuffer	1983
FI_Preferences::RootNode	1984
FI_Scroll::ScrollInfo	1984
FI_Terminal::Selection	1985
FI_Tile::Size_Range	1986
FI_Text_Display::Style_Table_Entry	1986
FI_Terminal::Utf8Char	1986

Chapter 8

Class Index

8.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

FI_Grid::Cell	363
FI_Terminal::CharStyle	364
FI_GIF_Image::GIF_FRAME::CPAL	365
FI_Terminal::Cursor	365
FI_Preferences::Entry	365
FI_Terminal::EscapeSeq	366
FI	
The FI is the FLTK global (static) class containing state information and global methods for the current application	366
FI_Adjuster	
Was stolen from Prisms, and has proven to be very useful for values that need a large dynamic range	398
FI_Anim_GIF_Image	
Supports loading, caching, and drawing of animated Compuserve GIF SM images	408
FI_Bitmap	
Supports caching and drawing of mono-color (bitmap) images	423
FI_BMP_Image	
Supports loading, caching, and drawing of Windows Bitmap (BMP) image files	428
FI_Box	
This widget simply draws its box, and possibly its label	433
FI_Browser	
Displays a scrolling list of text lines, and manages all the storage for the text	441
FI_Browser_	
This is the base class for browsers	471
FI_Button	
Buttons generate callbacks when they are clicked by the user	495
FI_Cairo_State	
Contains all the necessary info on the current cairo context	507
FI_Cairo_Window	
This defines an FLTK window with Cairo support	507
FI_Callback_User_Data	
A class prototype that allows for additional data in callbacks	521
FI_Chart	
FI_Chart displays simple charts	522
FL_CHART_ENTRY	
For internal use only	535
FI_Check_Browser	
Displays a scrolling list of text lines that may be selected and/or checked by the user	536
FI_Check_Button	
A button with a "checkmark" to show its status	551

FI_Choice	A button that is used to pop up a menu	559
FI_Clock	This widget provides a round analog clock display	571
FI_Clock_Output	This widget can be used to display a program-supplied time	581
FI_Color_Chooser	Standard RGB color chooser	591
FI_Copy_Surface	Supports copying of graphical data to the clipboard	604
FI_Counter	Controls a single floating point value with button (or keyboard) arrows	608
FI_Device_Plugin	This plugin socket allows the integration of new device drivers for special window or screen types	619
FI_Dial	Circular dial to control a single floating point value	620
FI_Display_Device	The computer's display	630
FI_Double_Window	The FI_Double_Window provides a double-buffered window	631
FI_End	This is a dummy class that allows you to end a FI_Group in a constructor list of a class:	645
FI_EPS_File_Surface	Encapsulated PostScript drawing surface	645
FI_File_Browser	Displays a list of filenames, optionally with file-specific icons	649
FI_File_Chooser	Displays a standard file selection dialog that supports various selection modes	664
FI_File_Icon	Manages icon images that can be used as labels in other widgets and as icons in the FileBrowser widget	671
FI_File_Input	This widget displays a pathname in a text input field	676
FI_Fill_Dial	Draws a dial with a filled arc	688
FI_Fill_Slider	Widget that draws a filled horizontal slider, useful as a progress or value meter	697
FI_Flex	FI_Flex is a container (layout) widget for one row or one column of widgets	705
FI_Float_Input	Subclass of FI_Input that only allows the user to type floating point numbers (sign, digits, decimal point, more digits, 'E' or 'e', sign, digits)	723
FI_FormsBitmap	Forms compatibility Bitmap Image Widget	733
FI_FormsPixmap	Forms pixmap drawing routines	740
FI_FormsText		748
FI_Free	Emulation of the Forms "free" widget	755
FI_GIF_Image	Supports loading, caching, and drawing of Compuserve GIF SM images	764
FI_GI_Choice		769
FI_GI_Window	Sets things up so OpenGL works	769
FI_Glut_Bitmap_Font	Gtk glut font/size attributes used in the glutXXX functions	790
FI_Glut_StrokeChar		790
FI_Glut_StrokeFont		790

FI_Glut_StrokeStrip	791
FI_Glut_StrokeVertex	791
FI_Glut_Window	
GLUT is emulated using this window class and these static variables (plus several more static variables hidden in glut_compatibility.cxx):	791
FI_Grid	
FI_Grid is a container (layout) widget with multiple columns and rows	806
FI_Group	
Main FLTK container widget	828
FI_Help_Block	848
FI_Help_Dialog	
Displays a standard help dialog window using the FI_Help_View widget	848
FI_Help_Font_Stack	850
FI_Help_Font_Style	
FI_Help_View font stack element definition	851
FI_Help_Link	
Definition of a link for the html viewer	851
FI_Help_Target	
FI_Help_Target structure	852
FI_Help_View	
Displays HTML text	852
FI_Hold_Browser	
The FI_Hold_Browser is a subclass of FI_Browser which lets the user select a single item, or no items by clicking on the empty space	867
FI_Hor_Fill_Slider	881
FI_Hor_Nice_Slider	
Single thumb tab slider	889
FI_Hor_Slider	
Horizontal Slider class	898
FI_Hor_Value_Slider	906
FI_ICO_Image	
Supports loading, caching, and drawing of Windows icon (.ico) files	915
FI_Image	
Base class for image caching, scaling and drawing	919
FI_Image_Reader	929
FI_Image_Surface	
Directs all graphics requests to an FI_Image	929
FI_Input	
This is the FLTK text input widget	935
FI_Input_	
This class provides a low-overhead text input field	948
FI_Input_Choice	
A combination of the input widget and a menu button	975
FI_Int_Input	
Subclass of FI_Input that only allows the user to type decimal digits (or hex numbers of the form 0xae)	988
FI_JPEG_Image	
Supports loading, caching, and drawing of Joint Photographic Experts Group (JPEG) File Interchange Format (JFIF) images	998
FI_Label	
This struct stores all information for a text or mixed graphics label	1002
FI_Light_Button	
This subclass displays the "on" state by turning on a light, rather than drawing pushed in	1004
FI_Line_Dial	1013
FI_Mac_App_Menu	1021
FI_Menu_	
Base class of all widgets that have a menu in FLTK	1023

FI_Menu_Bar	This widget provides a standard menubar interface	1046
FI_Menu_Button	This is a button that when pushed pops up a menu (or hierarchy of menus) defined by an array of FI_Menu_Item objects	1057
FI_Menu_Item	The FI_Menu_Item structure defines a single menu item that is used by the FI_Menu_ class	1068
FI_Menu_Window	Window type used for menus	1082
FI_Multi_Browser	Subclass of FI_Browser which lets the user select any set of the lines	1095
FI_Multi_Label	Allows a mixed text and/or graphics label to be applied to an FI_Menu_Item or FI_Widget	1108
FI_Multiline_Input	This input field displays '\n' characters as new lines rather than ^J, and accepts the Return, Tab, and up and down arrow keys	1111
FI_Multiline_Output	This widget is a subclass of FI_Output that displays multiple lines of text	1121
FI_Native_File_Chooser	This class lets an FLTK application easily and consistently access the operating system's native file chooser	1132
FI_Nice_Slider		1138
FI_Output	This widget displays a piece of text	1146
FI_Overlay_Window	This window provides double buffering and also the ability to draw the "overlay" which is another picture placed on top of the main image	1157
FI_Pack	This widget was designed to add the functionality of compressing and aligning widgets	1171
FI_Paged_Device	Represents page-structured drawing surfaces	1182
FI_PDF_File_Surface	To send graphical output to a PDF file	1188
FI_Pixmap	Supports caching and drawing of colormap (pixmap) images, including transparency	1195
FI_Plugin	FI_Plugin allows link-time and run-time integration of binary modules	1200
FI_Plugin_Manager	FI_Plugin_Manager manages link-time and run-time plugin binaries	1201
FI_PNG_Image	Supports loading, caching, and drawing of Portable Network Graphics (PNG) image files	1206
FI_PNM_Image	Supports loading, caching, and drawing of Portable Anymap (PNM, PBM, PGM, PPM) image files	1209
FI_Positioner	This class is provided for Forms compatibility	1212
FI_PostScript_File_Device	To send graphical output to a PostScript file	1221
FI_Preferences	FI_Preferences store user settings between application starts	1230
FI_Printer	OS-independent print support	1252
FI_Progress	Displays a progress bar for the user	1260
FI_Radio_Button		1268
FI_Radio_Light_Button		1276
FI_Radio_Round_Button		1284
FI_Rect	Rectangle with standard FLTK coordinates (X, Y, W, H)	1292

FI_Scroll::FI_Region_LRTB	
A local struct to manage a region defined by left/right/top/bottom	1295
FI_Scroll::FI_Region_XYWH	
A local struct to manage a region defined by xywh	1295
FI_Repeat_Button	
The FI_Repeat_Button is a subclass of FI_Button that generates a callback when it is pressed and then repeatedly generates callbacks as long as it is held down	1296
FI_Return_Button	
The FI_Return_Button is a subclass of FI_Button that generates a callback when it is pressed or when the user presses the Enter key	1304
FI_RGB_Image	
Supports caching and drawing of full-color images with 1 to 4 channels of color information . . .	1313
FI_Roller	
"dolly" control commonly used to move 3D objects	1320
FI_Round_Button	
Buttons generate callbacks when they are clicked by the user	1329
FI_Round_Clock	
A clock widget of type FL_ROUND_CLOCK	1337
FI_Scheme	1346
FI_Scheme_Choice	1347
FI_Scroll	
This container widget lets you maneuver around a set of widgets much larger than your window	1358
FI_Scrollbar	
Displays a slider with arrow buttons at the ends of the scrollbar	1373
FI_Scroll::FI_Scrollbar_Data	
A local struct to manage a scrollbar's xywh region and tab values	1384
FI_Secret_Input	
Subclass of FI_Input that displays its input as a string of placeholders	1385
FI_Select_Browser	
The class is a subclass of FI_Browser which lets the user select a single item, or no items by clicking on the empty space	1395
FI_Shared_Image	
This class supports caching, loading, and drawing of image files	1409
FI_Shortcut_Button	
A button that allows the user to type a key combination to create shortcuts	1419
FI_Simple_Counter	
This widget creates a counter with only 2 arrow buttons	1428
FI_Single_Window	
This is the same as FI_Window	1437
FI_Slider	
Sliding knob inside a box	1450
FI_Spinner	
This widget is a combination of a numerical input widget and repeat buttons	1460
FI_Spinner::FI_Spinner_Input	1473
FI_Surface_Device	
A drawing surface that's susceptible to receive graphical output	1483
FI_SVG_File_Surface	
A drawing surface producing a Scalable Vector Graphics (SVG) file	1485
FI_SVG_Image	
Supports loading, caching and drawing of scalable vector graphics (SVG) images	1489
FI_Sys_Menu_Bar	
A class to create and modify menus that appear on macOS in the menu bar at the top of the screen	1497
FI_Table	
A table of widgets or other content	1512
FI_Table_Row	
A table with row selection capabilities	1539

Fl_Tabs	Container widget that displays a set of tabs, with each tab representing a different child widget	1555
Fl_Terminal	Terminal widget supporting Unicode/utf-8, ANSI/xterm escape codes with full RGB color control	1577
Fl_Text_Buffer	This class manages Unicode text displayed in one or more Fl_Text_Display widgets	1620
Fl_Text_Display	Rich text display widget	1637
Fl_Text_Editor	This is the FLTK text editor widget	1685
Fl_Text_Selection	This is an internal class for Fl_Text_Buffer to manage text selections	1708
Fl_Tile	Lets you resize its children by dragging the border between them	1712
Fl_Tiled_Image	This class supports tiling of images over a specified area	1729
Fl_Timeout	The internal class Fl_Timeout handles all timeout related functions	1734
Fl_Timer	This is provided only to emulate the Forms Timer widget	1740
Fl_Toggle_Button	The toggle button is a push button that needs to be clicked once to toggle on, and one more time to toggle off	1749
Fl_Tooltip	Tooltip support for all FLTK widgets	1757
Fl_Tree	Tree widget	1762
Fl_Tree_Item	Tree widget item	1810
Fl_Tree_Item_Array	Manages an array of Fl_Tree_Item pointers	1832
Fl_Tree_Prefs	Tree widget's preferences	1835
Fl_Valuator	Controls a single floating-point value and provides a consistent interface to set the value, range, and step, and insures that callbacks are done the same for every object	1840
Fl_Value_Input	Displays a numeric value	1850
Fl_Value_Output	Displays a floating point value	1862
Fl_Value_Slider	Fl_Slider widget with a box displaying the current value	1872
Fl_Widget	Fl_Widget is the base class for all widgets in FLTK	1884
Fl_Widget_Surface	A surface on which any FLTK widget can be drawn	1927
Fl_Widget_Tracker	This class should be used to control safe widget deletion	1930
Fl_Window	This widget produces an actual window	1932
Fl_Wizard	This widget is based off the Fl_Tabs widget, but instead of displaying tabs it only changes "tabs" under program control	1964
Fl_XBM_Image	Supports loading, caching, and drawing of X Bitmap (XBM) bitmap files	1973
Fl_XPM_Image	Supports loading, caching, and drawing of X Pixmap (XPM) images, including transparency	1976
Fl_GIF_Image::GIF_FRAME		1979

FI_ICO_Image::IconDirEntry	
Windows ICONDIRENTRY structure	
1979	
FI_Text_Editor::Key_Binding	
Simple linked list item associating a key/state to a function	1980
FI_Terminal::Margin	1980
FI_Preferences::Name	
'Name' provides a simple method to create numerical or more complex procedural names for entries and groups on the fly	1980
FI_Preferences::Node	1981
FI_Paged_Device::page_format	
Width, height and name of a page format	1982
FI_Terminal::PartialUtf8Buf	1983
FI_Terminal::RingBuffer	1983
FI_Preferences::RootNode	1984
FI_Scroll::ScrollInfo	
Structure to manage scrollbar and widget interior sizes	1984
FI_Terminal::Selection	1985
FI_Tile::Size_Range	1986
FI_Text_Display::Style_Table_Entry	
This structure associates the color, font, and font size of a string to draw with an attribute mask matching attr	1986
FI_Terminal::Utf8Char	1986

Chapter 9

File Index

9.1 File List

Here is a list of all documented files with brief descriptions:

Enumerations.H	
This file contains type definitions and general enumerations	1989
filename.H	
File names and URI utility functions	2023
FI.H	
FI static class	2026
FI_Adjuster.H	2034
FI_Anim_GIF_Image.H	2034
fl_ask.H	
API for common dialogs	2036
fl_attr.h	
This file defines compiler-specific macros	2039
FI_Bitmap.H	2042
FI_BMP_Image.H	2043
FI_Box.H	
FI_Box widget	2044
FI_Browser.H	2044
FI_Browser_.H	2046
FI_Button.H	2048
FI_Cairo.H	
Cairo is currently supported for the following platforms: Windows, macOS, Unix/Linux (X11 + Wayland)	2049
FI_Cairo_Window.H	
FI_Cairo_Window, an FLTK window incorporating a Cairo draw callback	2050
fl_callback_macros.H	
This file provides macros for easy function and method callbacks with multiple type safe arguments	2051
fl_casts.H	2060
FI_Chart.H	
FI_Chart widget	2060
FI_Check_Browser.H	2062
FI_Check_Button.H	2063
FI_Choice.H	2064
FI_Clock.H	2064
FI_Color_Chooser.H	
FI_Color_Chooser widget	2065
FI_Copy_Surface.H	2067
FI_Counter.H	2068
FI_Device.H	
Declaration of classes FI_Surface_Device , FI_Display_Device , FI_Device_Plugin	2069

FI_Dial.H	2070
FI_Double_Window.H	2071
fl_draw.H	
Utility header to pull drawing functions together	2071
FI_Export.H	2082
FI_File_Browser.H	2083
FI_File_Chooser.H	2084
FI_File_Icon.H	2086
FI_File_Input.H	2088
FI_Fill_Dial.H	2088
FI_Fill_Slider.H	2089
FI_Flex.H	2089
FI_Float_Input.H	2091
FI_FormsBitmap.H	2091
FI_FormsPixmap.H	2092
FI_Free.H	2092
FI_GIF_Image.H	2093
FI_GL_Window.H	2094
FI_Graphics_Driver.H	2095
FI_Grid.H	
FI_Grid container widget	2100
FI_Group.H	
FI_Group and FI_End classes	2104
FI_Help_Dialog.H	2106
FI_Help_View.H	2107
FI_Hold_Browser.H	2110
FI_Hor_Fill_Slider.H	2110
FI_Hor_Nice_Slider.H	2111
FI_Hor_Slider.H	2111
FI_Hor_Value_Slider.H	2112
FI_ICO_Image.H	2112
FI_Image.H	
FI_Image, FI_RGB_Image classes	2113
FI_Image_Surface.H	2115
FI_Input.H	2116
FI_Input_.H	2117
FI_Input_Choice.H	2120
FI_Int_Input.H	2122
FI_JPEG_Image.H	2122
FI_Light_Button.H	2123
FI_Line_Dial.H	2123
FI_Menu.H	2124
FI_Menu_.H	2124
FI_Menu_Bar.H	2125
FI_Menu_Button.H	2126
FI_Menu_Item.H	2126
FI_Menu_Window.H	2130
fl_message.H	2130
FI_Multi_Browser.H	2131
FI_Multi_Label.H	2131
FI_Multiline_Input.H	2132
FI_Multiline_Output.H	2132
FI_Native_File_Chooser.H	
FI_Native_File_Chooser widget	2132
FI_Nice_Slider.H	2135
FI_Object.H	2135
FI_Output.H	2136
FI_Overlay_Window.H	2136

Fl_Pack.H	2137
Fl_Paged_Device.H	
Declaration of class Fl_Paged_Device	2137
Fl_PDF_File_Surface.H	2139
Fl_Pixmap.H	2140
Fl_Plugin.H	2141
Fl_PNG_Image.H	2141
Fl_PNM_Image.H	2142
Fl_Positioner.H	2142
Fl_PostScript.H	
Declaration of classes Fl_PostScript_File_Device and Fl_EPS_File_Surface	2143
Fl_Preferences.H	2145
Fl_Printer.H	
Declaration of class Fl_Printer	2148
Fl_Progress.H	2149
Fl_Radio_Button.H	2149
Fl_Radio_Light_Button.H	2150
Fl_Radio_Round_Button.H	2150
Fl_Rect.H	2151
Fl_Repeat_Button.H	2152
Fl_Return_Button.H	2152
Fl_RGB_Image.H	2153
Fl_Roller.H	2153
Fl_Round_Button.H	2154
Fl_Round_Clock.H	2154
Fl_Scheme.H	2154
Fl_Scheme_Choice.H	2155
Fl_Scroll.H	2156
Fl_Scrollbar.H	2157
Fl_Secret_Input.H	2158
Fl_Select_Browser.H	2158
Fl_Shared_Image.H	
Fl_Shared_Image class	2158
Fl_Shortcut_Button.H	2161
fl_show_colormap.H	
The fl_show_colormap() function hides the implementation classes used to provide the popup window and color selection mechanism	2162
fl_show_input.H	2162
Fl_Simple_Counter.H	2162
Fl_Single_Window.H	2163
Fl_Slider.H	2163
Fl_Spinner.H	2164
fl_string_functions.h	
Public header for FLTK's platform-agnostic string handling	2166
Fl_SVG_File_Surface.H	2167
Fl_SVG_Image.H	2167
Fl_Sys_Menu_Bar.H	
Definition of class Fl_Sys_Menu_Bar	2168
Fl_Table.H	2169
Fl_Table_Row.H	2175
Fl_Tabs.H	2177
Fl_Terminal.H	
Fl_Terminal widget	2178
Fl_Text_Buffer.H	2188
Fl_Text_Display.H	2192
Fl_Text_Editor.H	2197
Fl_Tile.H	2198
Fl_Tiled_Image.H	2199

Fl_Timer.H	2200
Fl_Toggle_Button.H	2200
Fl_Toggle_Light_Button.H	2201
Fl_Toggle_Round_Button.H	2201
Fl_Tooltip.H	2201
Fl_Tree.H	
This file contains the definitions of the Fl_Tree class	2202
Fl_Tree_Item.H	
This file contains the definitions for Fl_Tree_Item	2206
Fl_Tree_Item_Array.H	
This file defines a class that manages an array of Fl_Tree_Item pointers	2210
Fl_Tree_Prefs.H	
This file contains the definitions for Fl_Tree 's preferences	2211
fl_types.h	
This file contains simple "C"-style type definitions	2217
fl_utf8.h	
Header for Unicode and UTF-8 character handling	2218
Fl_Valuator.H	2222
Fl_Value_Input.H	2223
Fl_Value_Output.H	2224
Fl_Value_Slider.H	2225
Fl_Widget.H	
Fl_Widget and Fl_Label classes	2225
Fl_Widget_Surface.H	2231
Fl_Window.H	
Fl_Window widget	2232
Fl_Wizard.H	2235
Fl_XBM_Image.H	2236
Fl_XPM_Image.H	2236
forms.H	2237
gl.h	
This file defines wrapper functions for OpenGL in FLTK	2246
gl2opengl.h	2251
gl_draw.H	2252
glu.h	2252
glut.H	2252
mac.H	
Mac OS X-specific symbols	2259
math.h	2261
names.h	
This file defines arrays of human readable names for FLTK symbolic constants	2262
platform.H	2263
platform_types.h	
Definitions of platform-dependent types	2264
wayland.H	
Definitions of functions specific to the Wayland platform	2268
win32.H	
Definitions of functions specific to the Windows platform	2269
x.H	2270
x11.H	
Definitions of functions specific to the X11 platform	2271
cgdebug.h	2273
fastarrow.h	2275
Fl.cxx	
Implementation of the member functions of class Fl	2276
fl_arc.cxx	
Utility functions for drawing arcs and circles	2278

fl_ask.cxx	
Utility functions for common dialogs	2278
fl_boxytype.cxx	
Drawing code for common box types	2280
fl_cmap.h	2281
fl_color.cxx	
Color handling	2285
FI_compose.cxx	
Utility functions to support text input	2285
fl_contrast.cxx	
Color contrast handling	2285
fl_curve.cxx	
Utility for drawing Bézier curves, adding the points to the current fl_begin/fl_vertex/fl_end path	2286
FI_Double_Window.cxx	
FI_Double_Window implementation	2286
FI_get_system_colors.cxx	
System color support	2286
FI_GI_Choice.H	2288
FI_GI_Window_Driver.H	2289
FI_Graphics_Driver.cxx	
Implementation of class FI_Graphics_Driver	2290
FI_Grid.cxx	
Implements the FI_Grid container widget	2290
FI_Image_Reader.h	2291
FI_Int_Vector.H	2292
FI_Message.h	2293
FI_Native_File_Chooser_Kdialog.H	2295
FI_Native_File_Chooser_Zenity.H	2295
fl_oxy.h	2296
FI_Paged_Device.cxx	
Implementation of class FI_Paged_Device	2296
fl_rect.cxx	
Drawing and clipping routines for rectangles	2296
FI_Screen_Driver.H	2297
FI_String.H	2299
FI_Sys_Menu_Bar_Driver.H	2301
FI_System_Driver.H	2302
FI_Timeout.cxx	2304
FI_Timeout.h	
FI_Timeout handling	2304
fl_vertex.cxx	
Portable drawing code for drawing arbitrary shapes with simple 2D transformations	2306
FI_Window_Driver.H	2306
fl_write_png.cxx	
PNG image support functions	2309
FI_XColor.H	2311
flstring.h	2311
freeglut_teapot_data.h	2312
mediumarrow.h	2314
numericstort.c	2314
print_button.h	2316
print_panel.h	2316
slowarrow.h	2317
utf8_internal.h	2317
vsnprintf.c	
Portable vsnprintf() implementation	2317
Xutf8.h	2318
case.h	2320

dingbats_.h	2341
spacing.h	2347
symbol_.h	2370
imKStoUCS.c	2383
armscii_8.h	2387
ascii.h	2388
big5.h	2388
big5_emacs.h	2436
cp1133.h	2438
cp1251.h	2439
cp1255.h	2440
cp1256.h	2442
cp936ext.h	2443
gb2312.h	2515
georgian_academy.h	2545
georgian_ps.h	2546
iso8859_1.h	2547
iso8859_10.h	2547
iso8859_11.h	2548
iso8859_13.h	2549
iso8859_14.h	2551
iso8859_15.h	2552
iso8859_16.h	2553
iso8859_2.h	2554
iso8859_3.h	2555
iso8859_4.h	2557
iso8859_5.h	2558
iso8859_6.h	2559
iso8859_7.h	2560
iso8859_8.h	2561
iso8859_9.h	2562
iso8859_9e.h	2563
jisx0201.h	2564
jisx0208.h	2565
jisx0212.h	2592
koi8_c.h	2617
koi8_r.h	2619
koi8_u.h	2620
ksc5601.h	2622
mulelao.h	2657
tatar_cyr.h	2658
tcvn.h	2659
tis620.h	2661
ucs2be.h	2662
utf8.h	2662
viscii.h	2664
mk_wcwidth.c	2665
ucs2fontmap.c	2669
utf8Utils.c	2673
Ximint.h	2676
Xlibint.h	2676

Chapter 10

Topic Documentation

10.1 Callback Function Typedefs

Typedefs

- typedef void(* **FI_Abort_Handler**) (const char *format,...)
Signature of set_abort functions passed as parameters.
- typedef int(* **FI_Args_Handler**) (int argc, char **argv, int &i)
Signature of args functions passed as parameters.
- typedef void(* **FI_Atclose_Handler**) ([FI_Window](#) *window, void *data)
Signature of set_atclose functions passed as parameters.
- typedef void(* **FI_Awake_Handler**) (void *data)
Signature of some wakeup callback functions passed as parameters.
- typedef void **FI_Box_Draw_F**(int x, int y, int w, int h, [FI_Color](#) color)
Signature of some box drawing functions passed as parameters.
- typedef void **FI_Box_Draw_Focus_F**([FI_Boxtype](#) bt, int x, int y, int w, int h, [FI_Color](#) fg, [FI_Color](#) bg)
Signature of box focus frame drawing functions.
- typedef void(* **FI_Clipboard_Notify_Handler**) (int source, void *data)
Signature of add_clipboard_notify functions passed as parameters.
- typedef int(* [FI_Event_Dispatch](#)) (int event, [FI_Window](#) *w)
Signature of event_dispatch functions passed as parameters.
- typedef int(* **FI_Event_Handler**) (int event)
Signature of add_handler functions passed as parameters.
- typedef void(* **FI_FD_Handler**) ([FL_SOCKET](#) fd, void *data)
Signature of add_fd functions passed as parameters.
- typedef void(* **FI_Idle_Handler**) (void *data)
Signature of add_idle callback functions passed as parameters.
- typedef void **FI_Label_Draw_F**(const [FI_Label](#) *label, int x, int y, int w, int h, [FI_Align](#) align)
Signature of some label drawing functions passed as parameters.
- typedef void **FI_Label_Measure_F**(const [FI_Label](#) *label, int &width, int &height)
Signature of some label measurement functions passed as parameters.
- typedef void(* **FI_Old_Idle_Handler**) ()
Signature of set_idle callback functions passed as parameters.
- typedef int(* **FI_System_Handler**) (void *event, void *data)
Signature of add_system_handler functions passed as parameters.
- typedef void(* [FI_Timeout_Handler](#)) (void *data)
Signature of timeout callback functions passed as parameters.

10.1.1 Detailed Description

Typedefs defined in <FL/Fl.H> for callback or handler functions passed as function parameters.

FLTK uses callback functions as parameters for some function calls, e.g. to set up global event handlers ([Fl::add_handler\(\)](#)), to add a timeout handler ([Fl::add_timeout\(\)](#)), and many more.

The typedefs defined in this group describe the function parameters used to set up or clear the callback functions and should also be referenced to define the callback function to handle such events in the user's code.

See also

[Fl::add_handler\(\)](#), [Fl::add_timeout\(\)](#), [Fl::repeat_timeout\(\)](#), [Fl::remove_timeout\(\)](#) and others

10.1.2 Typedef Documentation

10.1.2.1 Fl_Event_Dispatch

```
typedef int(* Fl_Event_Dispatch) (int event, Fl_Window *w)
```

Signature of event_dispatch functions passed as parameters.

See also

[Fl::event_dispatch\(Fl_Event_Dispatch\)](#)

10.1.2.2 Fl_Timeout_Handler

```
typedef void(* Fl_Timeout_Handler) (void *data)
```

Signature of timeout callback functions passed as parameters.

Please see [Fl::add_timeout\(\)](#) for details.

10.2 Windows handling functions

Windows and standard dialogs handling declared in <FL/Fl.H>

Functions

- static void [Fl::default_atclose](#) ([Fl_Window](#) *, void *)
Default callback for window widgets.
- static [Fl_Window](#) * [Fl::first_window](#) ()
Returns the first top-level window in the list of shown() windows.
- static void [Fl::first_window](#) ([Fl_Window](#) *)
Sets the window that is returned by [first_window\(\)](#).
- static [Fl_Window](#) * [Fl::grab](#) ()
Returns the window that currently receives all events.
- static void [Fl::grab](#) ([Fl_Window](#) *)
Selects the window to grab.
- static [Fl_Window](#) * [Fl::modal](#) ()
Returns the top-most [modal\(\)](#) window currently shown.
- static [Fl_Window](#) * [Fl::next_window](#) (const [Fl_Window](#) *)
Returns the next top-level window in the list of shown() windows.
- static void [Fl::set_abort](#) ([Fl_Abort_Handler](#) f)
For back compatibility, sets the void [Fl::fatal](#) handler callback.
- static void [Fl::set_atclose](#) ([Fl_Atclose_Handler](#) f)
For back compatibility, sets the [Fl::atclose](#) handler callback.

Variables

- static void(* [Fl::atclose](#))([Fl_Window](#) *, void *)
Back compatibility: default window callback handler.

10.2.1 Detailed Description

Windows and standard dialogs handling declared in `<FL/FL.H>`

10.2.2 Function Documentation

10.2.2.1 `default_atclose()`

```
void Fl::default_atclose (
    Fl_Window * window,
    void * v) [static]
```

Default callback for window widgets.

It hides the window and then calls the default widget callback.

10.2.2.2 `first_window()` [1/2]

```
Fl_Window * Fl::first_window () [static]
```

Returns the first top-level window in the list of `shown()` windows.

If a `modal()` window is shown this is the top-most modal window, otherwise it is the most recent window to get an event.

10.2.2.3 `first_window()` [2/2]

```
void Fl::first_window (
    Fl_Window * window) [static]
```

Sets the window that is returned by `first_window()`.

The window is removed from wherever it is in the list and inserted at the top. This is not done if `Fl::modal()` is on or if the window is not `shown()`. Because the first window is used to set the "parent" of modal windows, this is often useful.

10.2.2.4 `grab()` [1/2]

```
static Fl_Window * Fl::grab () [inline], [static]
```

Returns the window that currently receives all events.

Returns

The window that currently receives all events, or NULL if event grabbing is currently OFF.

10.2.2.5 `grab()` [2/2]

```
void Fl::grab (
    Fl_Window * win) [static]
```

Selects the window to grab.

This is used when pop-up menu systems are active.

Send all events to the passed window no matter where the pointer or focus is (including in other programs). The window *does not have to be shown()*, this lets the `handle()` method of a "dummy" window override all event handling and allows you to map and unmap a complex set of windows (under both X and Windows *some* window must be mapped because the system interface needs a window id).

If `grab()` is on it will also affect `show()` of windows by doing system-specific operations (on X it turns on override-redirect). These are designed to make menus popup reliably and faster on the system.

To turn off grabbing do `Fl::grab(0)`.

Be careful that your program does not enter an infinite loop while `grab()` is on. On X this will lock up your screen!

To avoid this potential lockup, all newer operating systems seem to limit mouse pointer grabbing to the time during which a mouse button is held down. Some OS's may not support grabbing at all.

10.2.2.6 `modal()`

```
static Fl_Window * Fl::modal () [inline], [static]
```

Returns the top-most `modal()` window currently shown.

This is the most recently shown() window with [modal\(\)](#) true, or NULL if there are no [modal\(\)](#) windows shown(). The [modal\(\)](#) window has its [handle\(\)](#) method called for all events, and no other windows will have [handle\(\)](#) called ([grab\(\)](#) overrides this).

10.2.2.7 next_window()

```
Fl_Window * Fl::next_window (
    const Fl_Window * window) [static]
```

Returns the next top-level window in the list of shown() windows.
You can use this call to iterate through all the windows that are shown().

Parameters

in	<i>window</i>	must be shown and not NULL

10.2.2.8 set_atclose()

```
static void Fl::set_atclose (
    Fl_Atclose_Handler f) [inline], [static]
```

For back compatibility, sets the [Fl::atclose](#) handler callback.
You can now simply change the callback for the window instead.

See also

[Fl_Window::callback\(Fl_Callback*\)](#)

10.2.3 Variable Documentation

10.2.3.1 atclose

```
void(* Fl::atclose)(Fl_Window *, void *)=default\_atclose [static], [default]
```

Back compatibility: default window callback handler.

See also

[Fl::set_atclose\(\)](#)

10.3 Events handling functions

[Fl](#) class events handling API declared in [<FL/Fl.H>](#)

Functions

- static void [Fl::add_handler](#) ([Fl_Event_Handler](#) ha)
Install a function to parse unrecognized events.
- static void [Fl::add_handler](#) ([Fl_Event_Handler](#) ha, [Fl_Event_Handler](#) before)
Install a function to parse unrecognized events with less priority than another function.
- static void [Fl::add_system_handler](#) ([Fl_System_Handler](#) h, void *data)
Install a function to intercept system events.
- static [Fl_Widget](#) * [Fl::belowmouse](#) ()
Gets the widget that is below the mouse.
- static void [Fl::belowmouse](#) ([Fl_Widget](#) *)
Sets the widget that is below the mouse.
- static [Fl_Callback_Reason](#) [Fl::callback_reason](#) ()
Give the reason for calling a callback.
- static int [Fl::compose](#) (int &del)
Any text editing widget should call this for each FL_KEYBOARD event.

- static void `Fl::compose_reset ()`
If the user moves the cursor, be sure to call `Fl::compose_reset()`.
- static void `Fl::disable_im ()`
Disables the system input methods facilities.
- static void `Fl::enable_im ()`
Enables the system input methods facilities.
- static int `Fl::event ()`
Returns the last event that was processed.
- static int `Fl::event_alt ()`
Returns non-zero if the Alt key is pressed.
- static int `Fl::event_button ()`
Gets which particular mouse button caused the current event.
- static int `Fl::event_button1 ()`
Returns non-zero if mouse button 1 is currently held down.
- static int `Fl::event_button2 ()`
Returns non-zero if mouse button 2 is currently held down.
- static int `Fl::event_button3 ()`
Returns non-zero if mouse button 3 is currently held down.
- static int `Fl::event_button4 ()`
Returns non-zero if mouse button 4 is currently held down.
- static int `Fl::event_button5 ()`
Returns non-zero if mouse button 5 is currently held down.
- static int `Fl::event_buttons ()`
Returns the mouse buttons state bits; if non-zero, then at least one button is pressed now.
- static int `Fl::event_clicks ()`
Returns non zero if we had a double click event.
- static void `Fl::event_clicks (int i)`
Manually sets the number returned by `Fl::event_clicks()`.
- static void * `Fl::event_clipboard ()`
During an `FL_PASTE` event of non-textual data, returns a pointer to the pasted data.
- static const char * `Fl::event_clipboard_type ()`
Returns the type of the pasted data during an `FL_PASTE` event.
- static int `Fl::event_command ()`
Returns non-zero if the `FL_COMMAND` key is pressed, either `FL_CTRL` or on OSX `FL_META`.
- static int `Fl::event_ctrl ()`
Returns non-zero if the Control key is pressed.
- static `Fl_Event_Dispatch Fl::event_dispatch ()`
Return the current event dispatch function.
- static void `Fl::event_dispatch (Fl_Event_Dispatch d)`
Set a new event dispatch function.
- static int `Fl::event_dx ()`
Returns the current horizontal mouse scrolling associated with the `FL_MOUSEWHEEL` event.
- static int `Fl::event_dy ()`
Returns the current vertical mouse scrolling associated with the `FL_MOUSEWHEEL` event.
- static int `Fl::event_inside (const Fl_Widget *)`
Returns whether or not the mouse event is inside a given child widget.
- static int `Fl::event_inside (int, int, int, int)`
Returns whether or not the mouse event is inside the given rectangle.
- static int `Fl::event_is_click ()`
Returns non-zero if the mouse has not moved far enough and not enough time has passed since the last `FL_PUSH` or `FL_KEYBOARD` event for it to be considered a "drag" rather than a "click".

- static void [Fl::event_is_click](#) (int i)
Clears the value returned by [Fl::event_is_click\(\)](#).
- static int [Fl::event_key](#) ()
Gets which key on the keyboard was last pushed.
- static int [Fl::event_key](#) (int key)
Returns true if the given `key` was held down (or pressed) during the last event.
- static int [Fl::event_length](#) ()
Returns the length of the text in [Fl::event_text\(\)](#).
- static int [Fl::event_original_key](#) ()
Returns the keycode of the last key event, regardless of the NumLock state.
- static int [Fl::event_shift](#) ()
Returns non-zero if the Shift key is pressed.
- static int [Fl::event_state](#) ()
Returns the keyboard and mouse button states of the last event.
- static int [Fl::event_state](#) (int mask)
Returns non-zero if any of the passed event state bits are turned on.
- static const char * [Fl::event_text](#) ()
Returns the text associated with the current event, including `FL_PASTE` or `FL_DND_RELEASE` events.
- static int [Fl::event_x](#) ()
Returns the mouse position of the event relative to the [Fl_Window](#) it was passed to.
- static int [Fl::event_x_root](#) ()
Returns the mouse position on the screen of the event.
- static int [Fl::event_y](#) ()
Returns the mouse position of the event relative to the [Fl_Window](#) it was passed to.
- static int [Fl::event_y_root](#) ()
Returns the mouse position on the screen of the event.
- static [Fl_Widget](#) * [Fl::focus](#) ()
Gets the current [Fl::focus\(\)](#) widget.
- static void [Fl::focus](#) ([Fl_Widget](#) *)
Sets the widget that will receive `FL_KEYBOARD` events.
- static int [Fl::get_key](#) (int key)
Returns true if the given `key` is held down now.
- static void [Fl::get_mouse](#) (int &, int &)
Return where the mouse is on the screen by doing a round-trip query to the server.
- static int [Fl::handle](#) (int, [Fl_Window](#) *)
Handle events from the window system.
- static int [Fl::handle_](#) (int, [Fl_Window](#) *)
Handle events from the window system.
- static [Fl_Event_Handler](#) [Fl::last_handler](#) ()
Returns the last function installed by a call to [Fl::add_handler\(\)](#).
- static [Fl_Widget](#) * [Fl::pushed](#) ()
Gets the widget that is being pushed.
- static void [Fl::pushed](#) ([Fl_Widget](#) *)
Sets the widget that is being pushed.
- static void [Fl::remove_handler](#) ([Fl_Event_Handler](#) h)
Removes a previously added event handler.
- static void [Fl::remove_system_handler](#) ([Fl_System_Handler](#) h)
Removes a previously added system event handler.
- static int [Fl::test_shortcut](#) ([Fl_Shortcut](#))
Tests the current event, which must be an `FL_KEYBOARD` or `FL_SHORTCUT`, against a shortcut value (described in [Fl_Button](#)).

Variables

- `const char *const fl_callback_reason_names []`
This is an array of callback reason names you can use to convert callback reasons into names.
- `const char *const fl_eventnames []`
This is an array of event names you can use to convert event numbers into names.
- `const char *const fl_fontnames []`
This is an array of font names you can use to convert font numbers into names.

10.3.1 Detailed Description

Fl class events handling API declared in `<FL/Fl.H>`

10.3.2 Function Documentation**10.3.2.1 add_handler() [1/2]**

```
void Fl::add_handler (
    Fl_Event_Handler ha) [static]
```

Install a function to parse unrecognized events.

If FLTK cannot figure out what to do with an event, it calls each of these functions (most recent first) until one of them returns non-zero. If none of them returns non-zero then the event is ignored. Events that cause this to be called are:

- `FL_SHORTCUT` events that are not recognized by any widget. This lets you provide global shortcut keys.
- `FL_SCREEN_CONFIGURATION_CHANGED` events. Under X11, this event requires the libXrandr.so shared library to be loadable at run-time and the X server to implement the RandR extension.
- `FL_ZOOM_EVENT` events.
- System events that FLTK does not recognize. See `fl_xevent`.
- *Some* other events when the widget FLTK selected returns zero from its `handle()` method. Exactly which ones may change in future versions, however.

See also

`Fl::remove_handler(Fl_Event_Handler)`
`Fl::event_dispatch(Fl_Event_Dispatch d)`
`Fl::handle(int, Fl_Window*)`

10.3.2.2 add_handler() [2/2]

```
void Fl::add_handler (
    Fl_Event_Handler ha,
    Fl_Event_Handler before) [static]
```

Install a function to parse unrecognized events with less priority than another function.

Install function `ha` to handle unrecognized events giving it the priority just lower than that of function `before` which was previously installed.

See also

`Fl::add_handler(Fl_Event_Handler)`
`Fl::last_handler()`

Since

1.4.0

10.3.2.3 `add_system_handler()`

```
void Fl::add_system_handler (
    Fl_System_Handler ha,
    void * data) [static]
```

Install a function to intercept system events.

FLTK calls each of these functions as soon as a new system event is received. The processing will stop at the first function to return non-zero. If all functions return zero then the event is passed on for normal handling by FLTK.

Each function will be called with a pointer to the system event as the first argument and `data` as the second argument. The system event pointer will always be `void *`, but will point to different objects depending on the platform:

- X11: XEvent
- Windows: MSG
- OS X: NSEvent
- Wayland: NULL (FLTK runs the event handler(s) just before calling `wl_display_dispatch()`)

Parameters

<i>ha</i>	The event handler function to register
<i>data</i>	User data to include on each call

See also

[Fl::remove_system_handler\(Fl_System_Handler\)](#)

10.3.2.4 `belowmouse()` [1/2]

```
static Fl_Widget * Fl::belowmouse () [inline], [static]
```

Gets the widget that is below the mouse.

See also

[belowmouse\(Fl_Widget*\)](#)

10.3.2.5 `belowmouse()` [2/2]

```
void Fl::belowmouse (
    Fl_Widget * o) [static]
```

Sets the widget that is below the mouse.

This is for highlighting buttons. It is not used to send `FL_PUSH` or `FL_MOVE` directly, for several obscure reasons, but those events typically go to this widget. This is also the first widget tried for `FL_SHORTCUT` events.

If you change the `belowmouse` widget, the previous one and all parents (that don't contain the new widget) are sent `FL_LEAVE` events. Changing this does *not* send `FL_ENTER` to this or any widget, because sending `FL_ENTER` is supposed to *test* if the widget wants the mouse (by it returning non-zero from [handle\(\)](#)).

10.3.2.6 `callback_reason()`

```
Fl_Callback_Reason Fl::callback_reason () [static]
```

Give the reason for calling a callback.

Returns

the reason for the current callback

See also

[Fl_Widget::when\(\)](#), [Fl_Widget::do_callback\(\)](#), [Fl_Widget::callback\(\)](#)

Since

1.4.0

10.3.2.7 compose()

```
int Fl::compose (
    int & del) [static]
```

Any text editing widget should call this for each FL_KEYBOARD event.

Use of this function is very simple.

If *true* is returned, then it has modified the [Fl::event_text\(\)](#) and [Fl::event_length\(\)](#) to a set of *bytes* to insert (it may be of zero length!). It will also set the "del" parameter to the number of *bytes* to the left of the cursor to delete, this is used to delete the results of the previous call to [Fl::compose\(\)](#).

If *false* is returned, the keys should be treated as function keys, and del is set to zero. You could insert the text anyways, if you don't know what else to do.

Text editing widgets can preferentially call [fl_set_spot\(\)](#) to indicate the window coordinates of the bottom of the current insertion point and the line height. This way, auxiliary windows that help choosing among alternative characters with some text input methods appear just below or above the insertion point. If widgets don't do that, such auxiliary windows appear at the widget's bottom.

On some platforms, text input can involve marked text, that is, temporary text replaced by other text during the input process. This occurs, e.g., under Wayland or macOS when using dead keys or when entering CJK characters. Text editing widgets should preferentially signal marked text, usually underlining it. Widgets can use `int Fl::compose_state` after having called [Fl::compose\(int&\)](#) to obtain the length in bytes of marked text that always finishes at the current insertion point. Widgets should also call void [fl_reset_spot\(\)](#) when processing FL_UNFOCUS events. The [Fl_Input](#) and [Fl_Text_Editor](#) widgets underline marked text. If none of this is done by a user-defined text editing widget, text input will work, but will not signal to the user what text is marked.

Finally, text editing widgets should call `set_flag(MAC_USE_ACCENTS_MENU)` ; in their constructor if they want to use, on the macOS platform, the feature introduced with Mac OS 10.7 "Lion" where pressing and holding certain keys on the keyboard opens a diacritic marks popup window.

Note

For compatibility with FLTK 1.3, text editing widgets can call `Fl::insertion_point_location(int x, int y, int height)` and `Fl::reset_marked_text()` only under the macOS platform to indicate/reset the coordinates of the current insertion point. This is deprecated in version 1.4 because redundant with the platform-independent [fl_set_spot\(\)](#) and [fl_reset_spot\(\)](#) functions.

10.3.2.8 compose_reset()

```
void Fl::compose_reset () [static]
```

If the user moves the cursor, be sure to call [Fl::compose_reset\(\)](#).

The next call to [Fl::compose\(\)](#) will start out in an initial state. In particular it will not set "del" to non-zero. This call is very fast so it is ok to call it many times and in many places.

10.3.2.9 disable_im()

```
void Fl::disable_im () [static]
```

Disables the system input methods facilities.

See also

[enable_im\(\)](#)

10.3.2.10 enable_im()

```
void Fl::enable_im () [static]
```

Enables the system input methods facilities.

This is the default.

See also

[disable_im\(\)](#)

10.3.2.11 event()

```
static int Fl::event () [inline], [static]
```

Returns the last event that was processed.

This can be used to determine if a callback is being done in response to a keypress, mouse click, etc.

10.3.2.12 event_button()

```
static int Fl::event_button () [inline], [static]
```

Gets which particular mouse button caused the current event.

This returns garbage if the most recent event was not a FL_PUSH or FL_RELEASE event.

Return values

<i>FL_LEFT_MOUSE</i>	
<i>FL_MIDDLE_MOUSE</i>	
<i>FL_RIGHT_MOUSE</i>	
<i>FL_BACK_MOUSE</i>	
<i>FL_FORWARD_MOUSE</i>	

See also

[Fl::event_buttons\(\)](#), [Fl::event_state\(\)](#)

10.3.2.13 event_button1()

```
static int Fl::event_button1 () [inline], [static]
```

Returns non-zero if mouse button 1 is currently held down.

For more details, see [Fl::event_buttons\(\)](#).

10.3.2.14 event_button2()

```
static int Fl::event_button2 () [inline], [static]
```

Returns non-zero if mouse button 2 is currently held down.

For more details, see [Fl::event_buttons\(\)](#).

10.3.2.15 event_button3()

```
static int Fl::event_button3 () [inline], [static]
```

Returns non-zero if mouse button 3 is currently held down.

For more details, see [Fl::event_buttons\(\)](#).

10.3.2.16 event_button4()

```
static int Fl::event_button4 () [inline], [static]
```

Returns non-zero if mouse button 4 is currently held down.

For more details, see [Fl::event_buttons\(\)](#).

10.3.2.17 event_button5()

```
static int Fl::event_button5 () [inline], [static]
```

Returns non-zero if mouse button 5 is currently held down.

For more details, see [Fl::event_buttons\(\)](#).

10.3.2.18 event_buttons()

```
static int Fl::event_buttons () [inline], [static]
```

Returns the mouse buttons state bits; if non-zero, then at least one button is pressed now.

This function returns the button state at the time of the event. During an FL_RELEASE event, the state of the released button will be 0. To find out, which button caused an FL_RELEASE event, you can use [Fl::event_button\(\)](#) instead.

Returns

a bit mask value like { [FL_BUTTON1] | [FL_BUTTON2] | ... | [FL_BUTTON5] }

10.3.2.19 event_clicks() [1/2]

```
static int Fl::event_clicks () [inline], [static]
```

Returns non zero if we had a double click event.

Return values

<i>Non-zero</i>	if the most recent FL_PUSH or FL_KEYBOARD was a "double click".
<i>N-1</i>	for N clicks. A double click is counted if the same button is pressed again while event_is_click() is true.

10.3.2.20 event_clicks() [2/2]

```
static void Fl::event_clicks (
    int i) [inline], [static]
```

Manually sets the number returned by [Fl::event_clicks\(\)](#).

This can be used to set it to zero so that later code does not think an item was double-clicked.

Parameters

in	<i>i</i>	corresponds to no double-click if 0, i+1 mouse clicks otherwise

See also

int [event_clicks\(\)](#)

10.3.2.21 event_clipboard()

```
static void * Fl::event_clipboard () [inline], [static]
```

During an FL_PASTE event of non-textual data, returns a pointer to the pasted data.

The returned data is an [Fl_RGB_Image](#) * when the result of [Fl::event_clipboard_type\(\)](#) is [Fl::clipboard_image](#).

10.3.2.22 event_clipboard_type()

```
static const char * Fl::event_clipboard_type () [inline], [static]
```

Returns the type of the pasted data during an FL_PASTE event.

This type can be [Fl::clipboard_plain_text](#) or [Fl::clipboard_image](#).

10.3.2.23 event_dispatch()

```
void Fl::event_dispatch (
    Fl_Event_Dispatch d) [static]
```

Set a new event dispatch function.

The event dispatch function is called after native events are converted to FLTK events, but before they are handled by FLTK. If the dispatch function `Fl_Event_Dispatch d` is set, it is up to the dispatch function to call `Fl::handle_(int, Fl_Window*)` or to ignore the event.

The dispatch function itself must return 0 if it ignored the event, or non-zero if it used the event. If you call `Fl::handle_()`, then this will return the correct value.

The event dispatch can be used to handle exceptions in FLTK events and callbacks before they reach the native event handler:

```
int myHandler(int e, Fl_Window *w) {
    try {
        return Fl::handle_(e, w);
    } catch () {
        ...
    }
}

main() {
    Fl::event_dispatch(myHandler);
    ...
    Fl::run();
}
```

Parameters

<code>d</code>	new dispatch function, or NULL

See also

[Fl::add_handler\(Fl_Event_Handler\)](#)

[Fl::handle\(int, Fl_Window*\)](#)

[Fl::handle_\(int, Fl_Window*\)](#)

10.3.2.24 event_dx()

```
static int Fl::event_dx () [inline], [static]
```

Returns the current horizontal mouse scrolling associated with the `FL_MOUSEWHEEL` event.

Right is positive.

10.3.2.25 event_dy()

```
static int Fl::event_dy () [inline], [static]
```

Returns the current vertical mouse scrolling associated with the `FL_MOUSEWHEEL` event.

Down is positive.

10.3.2.26 event_inside() [1/2]

```
int Fl::event_inside (
    const Fl_Widget * o) [static]
```

Returns whether or not the mouse event is inside a given child widget.

Returns non-zero if the current `Fl::event_x()` and `Fl::event_y()` put it inside the given child widget's bounding box.

This method can only be used to check whether the mouse event is inside a **child** widget of the window that handles the event, and there must not be an intermediate subwindow (i.e. the widget must not be inside a subwindow of the current window). However, it is valid if the widget is inside a nested `Fl_Group`.

You must not use it with the window itself as the `o` argument in a window's `handle()` method.

Note

The mentioned restrictions are necessary, because this method does not transform coordinates of child widgets, and thus the given widget `o` must be within the *same* window that is handling the current event. Otherwise the results are undefined.

You should always call this rather than doing your own comparison so you are consistent about edge effects.

See also

[Fl::event_inside\(int, int, int, int\)](#)

Parameters

<code>in</code>	<code>o</code>	child widget to be tested

Returns

non-zero, if mouse event is inside the widget

10.3.2.27 event_inside() [2/2]

```
int Fl::event_inside (
    int xx,
    int yy,
    int ww,
    int hh) [static]
```

Returns whether or not the mouse event is inside the given rectangle.

Returns non-zero if the current [Fl::event_x\(\)](#) and [Fl::event_y\(\)](#) put it inside the given arbitrary bounding box.

You should always call this rather than doing your own comparison so you are consistent about edge effects.

To find out, whether the event is inside a child widget of the current window, you can use [Fl::event_inside\(const Fl_Widget *\)](#).

Parameters

<code>in</code>	<code>xx,yy,ww,hh</code>	bounding box

Returns

non-zero, if mouse event is inside

10.3.2.28 event_is_click() [1/2]

```
static int Fl::event_is_click () [inline], [static]
```

Returns non-zero if the mouse has not moved far enough and not enough time has passed since the last `FL_PUSH` or `FL_KEYBOARD` event for it to be considered a "drag" rather than a "click".

You can test this on `FL_DRAG`, `FL_RELEASE`, and `FL_MOVE` events.

10.3.2.29 event_is_click() [2/2]

```
static void Fl::event_is_click (
    int i) [inline], [static]
```

Clears the value returned by [Fl::event_is_click\(\)](#).

Useful to prevent the *next* click from being counted as a double-click or to make a popup menu pick an item with a single click. Don't pass non-zero to this.

10.3.2.30 event_key() [1/2]

```
static int Fl::event_key () [inline], [static]
```

Gets which key on the keyboard was last pushed.

The returned integer 'key code' is not necessarily a text equivalent for the keystroke. For instance: if someone presses '5' on the numeric keypad with numlock on, [Fl::event_key\(\)](#) may return the 'key code' for this key, and NOT the character '5'. To always get the '5', use [Fl::event_text\(\)](#) instead.

Returns

an integer 'key code', or 0 if the last event was not a key press or release.

See also

int [event_key\(int\)](#), [event_text\(\)](#), [compose\(int&\)](#).

10.3.2.31 event_key() [2/2]

```
int Fl::event_key (
    int key) [static]
```

Returns true if the given *key* was held down (or pressed) *during* the last event.

This is constant until the next event is read from the server.

[Fl::get_key\(int\)](#) returns true if the given key is held down *now*. Under X this requires a round-trip to the server and is *much* slower than [Fl::event_key\(int\)](#).

Keys are identified by the *unshifted* values. FLTK defines a set of symbols that should work on most modern machines for every key on the keyboard:

- All keys on the main keyboard producing a printable ASCII character use the value of that ASCII character (as though shift, ctrl, and caps lock were not on). The space bar is 32.
- All keys on the numeric keypad producing a printable ASCII character use the value of that ASCII character plus FL_KP (e.g., FL_KP + '4', FL_KP + '/'). The highest possible value is FL_KP_Last so you can range-check to see if something is on the keypad.
- All numbered function keys use the number on the function key plus FL_F. The highest possible number is FL_F_Last, so you can range-check a value.
- Buttons on the mouse are considered keys, and use the button number (where the left button is 1) plus FL_Button.
- All other keys on the keypad have a symbol: FL_Escape, FL_BackSpace, FL_Tab, FL_Enter, FL_Print, FL_Scroll_Lock, FL_Pause, FL_Insert, FL_Home, FL_Page_Up, FL_Delete, FL_End, FL_Page_Down, FL_Left, FL_Up, FL_Right, FL_Down, FL_Iso_Key, FL_Shift_L, FL_Shift_R, FL_Control_L, FL_Control_R, FL_Caps_Lock, FL_Alt_L, FL_Alt_R, FL_Meta_L, FL_Meta_R, FL_Menu, FL_Num_Lock, FL_KP_Enter. Be careful not to confuse these with the very similar, but all-caps, symbols used by [Fl::event_state\(\)](#).

On X [Fl::get_key\(FL_Button+n\)](#) does not work.

On Windows [Fl::get_key\(FL_KP_Enter\)](#) and [Fl::event_key\(FL_KP_Enter\)](#) do not work.

10.3.2.32 event_length()

```
static int Fl::event_length () [inline], [static]
```

Returns the length of the text in [Fl::event_text\(\)](#).

There will always be a nul at this position in the text. However there may be a nul before that if the keystroke translates to a nul character or you paste a nul character.

10.3.2.33 event_original_key()

```
static int Fl::event_original_key () [inline], [static]
```

Returns the keycode of the last key event, regardless of the NumLock state.

If NumLock is deactivated, FLTK translates events from the numeric keypad into the corresponding arrow key events. [event_key\(\)](#) returns the translated key code, whereas [event_original_key\(\)](#) returns the keycode before NumLock translation.

10.3.2.34 event_state() [1/2]

```
static int Fl::event_state () [inline], [static]
```

Returns the keyboard and mouse button states of the last event.

This is a bitfield of what shift states were on and what mouse buttons were held down during the most recent event.

Note

FLTK platforms differ in what [Fl::event_state\(\)](#) returns when it is called while a modifier key or mouse button is being pressed or released.

- Under X11 and Wayland, [Fl::event_state\(\)](#) indicates the state of the modifier keys and mouse buttons just **prior** to the event. Thus, during the `FL_KEYDOWN` event generated when pressing the shift key, for example, the `FL_SHIFT` bit of [event_state\(\)](#) is 0 and becomes 1 only at the next event which can be any other event, including e.g. `FL_MOVE`.
- Under other platforms the reported state of modifier keys or mouse buttons includes that of the key or button being pressed or released.
- [Fl::event_state\(\)](#) returns the same value under all platforms when it's called while a non-modifier key (e.g. a letter or function key) is being pressed or released.
- X servers do not agree on shift states, and `FL_NUM_LOCK`, `FL_META`, and `FL_SCROLL_LOCK` may not work.
- The values were selected to match the XFree86 server on Linux.

Note

This inconsistency **may** be fixed (on X11 and Wayland) in a later release.

The legal event state bits are:

Device	State Bit	Key or Button	Since
Keyboard	<code>FL_SHIFT</code>	Shift	
Keyboard	<code>FL_CAPS_LOCK</code>	Caps Lock	
Keyboard	<code>FL_CTRL</code>	Ctrl	
Keyboard	<code>FL_ALT</code>	Alt	
Keyboard	<code>FL_NUM_LOCK</code>	Num Lock	
Keyboard	<code>FL_META</code>	Meta, e.g. "Windows"	
Keyboard	<code>FL_SCROLL_LOCK</code>	Scroll Lock	
Mouse	<code>FL_BUTTON1</code>	left button	
Mouse	<code>FL_BUTTON2</code>	middle button	
Mouse	<code>FL_BUTTON3</code>	right button	
Mouse	<code>FL_BUTTON4</code>	side button 1 (back)	1.3.10
Mouse	<code>FL_BUTTON5</code>	side button 2 (forward)	1.3.10

10.3.2.35 event_state() [2/2]

```
static int Fl::event_state (
    int mask) [inline], [static]
```

Returns non-zero if any of the passed event state bits are turned on.

Use `mask` to pass the event states you're interested in. The legal event state bits are defined in [Fl::event_state\(\)](#).

10.3.2.36 `event_text()`

```
static const char * Fl::event_text () [inline], [static]
```

Returns the text associated with the current event, including `FL_PASTE` or `FL_DND_RELEASE` events.

This can be used in response to `FL_KEYUP`, `FL_KEYDOWN`, `FL_PASTE`, and `FL_DND_RELEASE`.

When responding to `FL_KEYUP`/`FL_KEYDOWN`, use this function instead of `Fl::event_key()` to get the text equivalent of keystrokes suitable for inserting into strings and text widgets.

The returned string is guaranteed to be NULL terminated. However, see `Fl::event_length()` for the actual length of the string, in case the string itself contains NULLs that are part of the text data.

Returns

A NULL terminated text string equivalent of the last keystroke.

10.3.2.37 `event_x_root()`

```
static int Fl::event_x_root () [inline], [static]
```

Returns the mouse position on the screen of the event.

To find the absolute position of an `Fl_Window` on the screen, use the difference between `event_x_root()`, `event_y_root()` and `event_x()`, `event_y()`.

10.3.2.38 `event_y_root()`

```
static int Fl::event_y_root () [inline], [static]
```

Returns the mouse position on the screen of the event.

To find the absolute position of an `Fl_Window` on the screen, use the difference between `event_x_root()`, `event_y_root()` and `event_x()`, `event_y()`.

10.3.2.39 `focus()` [1/2]

```
static Fl_Widget * Fl::focus () [inline], [static]
```

Gets the current `Fl::focus()` widget.

See also

`Fl::focus(Fl_Widget*)`

10.3.2.40 `focus()` [2/2]

```
void Fl::focus (
    Fl_Widget * o) [static]
```

Sets the widget that will receive `FL_KEYBOARD` events.

Use this function inside the `handle(int)` member function of a widget of yours to give focus to the widget, for example when it receives the `FL_FOCUS` or the `FL_PUSH` event. Otherwise, use `Fl_Widget::take_focus()` to give focus to a widget;

If you change `Fl::focus()`, the previous widget and all parents (that don't contain the new widget) are sent `FL_↔UNFOCUS` events. Changing the focus does *not* send `FL_FOCUS` to this or any widget, because sending `FL_↔FOCUS` is supposed to *test* if the widget wants the focus (by it returning non-zero from `handle()`).

Since FLTK 1.4.0 widgets can set the `NEEDS_KEYBOARD` flag to indicate that a keyboard is essential for the widget to function. Touchscreen devices will be sent a request to show an on-screen keyboard if no hardware keyboard is connected.

See also

`Fl_Widget::take_focus()`

`Fl_Widget::needs_keyboard() const`

`Fl_Widget::needs_keyboard(bool)`

10.3.2.41 get_key()

```
int Fl::get_key (
    int key) [static]
```

Returns true if the given `key` is held down *now*.

Under X this requires a round-trip to the server and is *much* slower than [Fl::event_key\(int\)](#).

See also

[event_key\(int\)](#)

10.3.2.42 get_mouse()

```
void Fl::get_mouse (
    int & x,
    int & y) [static]
```

Return where the mouse is on the screen by doing a round-trip query to the server.

You should use [Fl::event_x_root\(\)](#) and [Fl::event_y_root\(\)](#) if possible, but this is necessary if you are not sure if a mouse event has been processed recently (such as to position your first window). If the display is not open, this will open it.

10.3.2.43 handle()

```
int Fl::handle (
    int e,
    Fl_Window * window) [static]
```

Handle events from the window system.

This is called from the native event dispatch after native events have been converted to FLTK notation. This function calls [Fl::handle_\(int, Fl_Window*\)](#) unless the user sets a dispatch function. If a user dispatch function is set, the user must make sure that [Fl::handle_\(\)](#) is called, or the event will be ignored.

Parameters

<i>e</i>	the event type (Fl::event_number() is not yet set)
<i>window</i>	the window that caused this event

Returns

0 if the event was not handled

See also

[Fl::add_handler\(Fl_Event_Handler\)](#)

[Fl::event_dispatch\(Fl_Event_Dispatch\)](#)

10.3.2.44 handle_()

```
int Fl::handle_ (
    int e,
    Fl_Window * window) [static]
```

Handle events from the window system.

This function is called from the native event dispatch, unless the user sets another dispatch function. In that case, the user dispatch function must decide when to call [Fl::handle_\(int, Fl_Window*\)](#)

Callbacks can set `FL_REASON_CLOSED` and `FL_REASON_CANCELLED`.

Parameters

<i>e</i>	the event type (Fl::event_number() is not yet set)
<i>window</i>	the window that caused this event

Returns

0 if the event was not handled

See also

[Fl::event_dispatch\(Fl_Event_Dispatch\)](#)

10.3.2.45 last_handler()

```
Fl_Event_Handler Fl::last_handler () [static]
```

Returns the last function installed by a call to [Fl::add_handler\(\)](#).

Since

1.4.0

10.3.2.46 pushed() [1/2]

```
static Fl_Widget * Fl::pushed () [inline], [static]
```

Gets the widget that is being pushed.

See also

[void pushed\(Fl_Widget*\)](#)

10.3.2.47 pushed() [2/2]

```
void Fl::pushed (
    Fl_Widget * o) [static]
```

Sets the widget that is being pushed.

FL_DRAG or FL_RELEASE (and any more FL_PUSH) events will be sent to this widget.

If you change the pushed widget, the previous one and all parents (that don't contain the new widget) are sent FL_RELEASE events. Changing this does *not* send FL_PUSH to this or any widget, because sending FL_PUSH is supposed to *test* if the widget wants the mouse (by it returning non-zero from [handle\(\)](#)).

10.3.2.48 remove_handler()

```
void Fl::remove_handler (
    Fl_Event_Handler ha) [static]
```

Removes a previously added event handler.

See also

[Fl::handle\(int, Fl_Window*\)](#)

10.3.2.49 remove_system_handler()

```
void Fl::remove_system_handler (
    Fl_System_Handler ha) [static]
```

Removes a previously added system event handler.

Parameters

<i>ha</i>	The event handler function to remove

See also

[Fl::add_system_handler\(Fl_System_Handler\)](#)

10.3.2.50 test_shortcut()

```
int Fl::test_shortcut (
    Fl_Shortcut shortcut) [static]
```

Tests the current event, which must be an FL_KEYBOARD or FL_SHORTCUT, against a shortcut value (described in [Fl_Button](#)).

Not to be confused with [Fl_Widget::test_shortcut\(\)](#).

Returns

non-zero if there is a match.

10.3.3 Variable Documentation

10.3.3.1 fl_callback_reason_names

```
const char* const fl_callback_reason_names[]
```

Initial value:

```
=
{
    "FL_REASON_UNKNOWN",
    "FL_REASON_SELECTED",
    "FL_REASON_DESELECTED",
    "FL_REASON_RESELECTED",
    "FL_REASON_OPENED",
    "FL_REASON_CLOSED",
    "FL_REASON_DRAGGED",
    "FL_REASON_CANCELLED",
    "FL_REASON_CHANGED",
    "FL_REASON_GOT_FOCUS",
    "FL_REASON_LOST_FOCUS",
    "FL_REASON_RELEASED",
    "FL_REASON_ENTER_KEY",
    NULL, NULL, NULL,
    NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL,
    NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL,
    "FL_REASON_USER", "FL_REASON_USER+1", "FL_REASON_USER+2", "FL_REASON_USER+3",
}
```

This is an array of callback reason names you can use to convert callback reasons into names.

The array gets defined inline wherever your '#include <[FL/names.h](#)>' appears.

10.3.3.2 fl_eventnames

```
const char* const fl_eventnames[]
```

This is an array of event names you can use to convert event numbers into names.

The array gets defined inline wherever your '#include <[FL/names.h](#)>' appears.

Example:

```
#include <FL/names.h> // array will be defined here
int MyClass::handle(int e) {
    printf("Event was %s (%d)\n", fl_eventnames[e], e);
    // ..resulting output might be e.g. "Event was FL_PUSH (1)"..
    [...]
}
```

10.3.3.3 fl_fontnames

```
const char* const fl_fontnames[]
```

Initial value:

```
=
{
    "FL_HELVETICA",
    "FL_HELVETICA_BOLD",
    "FL_HELVETICA_ITALIC",
    "FL_HELVETICA_BOLD_ITALIC",
    "FL_COURIER",
    "FL_COURIER_BOLD",
    "FL_COURIER_ITALIC",
    "FL_COURIER_BOLD_ITALIC",
    "FL_TIMES",
    "FL_TIMES_BOLD",
    "FL_TIMES_ITALIC",
    "FL_TIMES_BOLD_ITALIC",
    "FL_SYMBOL",
    "FL_SCREEN",
    "FL_SCREEN_BOLD",
}
```

```
"FL_ZAPF_DINGBATS",
}
```

This is an array of font names you can use to convert font numbers into names. The array gets defined inline wherever your `#include <FL/names.h>` appears.

Example:

```
#include <FL/names.h>           // array will be defined here
int MyClass::my_callback(FL_Widget *w, void*) {
    int fnum = w->labelfont();
    // Resulting output might be e.g. "Label's font is FL_HELVETICA (0)"
    printf("Label's font is %s (%d)\n", fl_fontnames[fnum], fnum);
    // ..resulting output might be e.g. "Label's font is FL_HELVETICA (0)"..
    [...]
}
```

10.4 Selection & Clipboard functions

FLTK global copy/cut/paste functions declared in `<FL/Fl.H>`

Functions

- static void `Fl::add_clipboard_notify` (`Fl_Clipboard_Notify_Handler` h, void *data=0)
FLTK will call the registered callback whenever there is a change to the selection buffer or the clipboard.
- static int `Fl::clipboard_contains` (const char *type)
Returns non 0 if the clipboard contains data matching type.
- static void `Fl::copy` (const char *stuff, int len, int destination=0, const char *type=`Fl::clipboard_plain_text`)
Copies data to the selection buffer, the clipboard, or both.
- static int `Fl::dnd` ()
Initiate a Drag And Drop operation.
- static void `Fl::paste` (`Fl_Widget` &receiver)
Backward compatibility only.
- static void `Fl::paste` (`Fl_Widget` &receiver, int source, const char *type=`Fl::clipboard_plain_text`)
Pastes the data from the selection buffer (source is 0) or the clipboard (source is 1) into receiver.
- static void `Fl::remove_clipboard_notify` (`Fl_Clipboard_Notify_Handler` h)
Stop calling the specified callback when there are changes to the selection buffer or the clipboard.
- static void `Fl::selection` (`Fl_Widget` &owner, const char *, int len)
Changes the current selection.
- static `Fl_Widget` * `Fl::selection_owner` ()
back-compatibility only: Gets the widget owning the current selection
- static void `Fl::selection_owner` (`Fl_Widget` *)
Back-compatibility only: The single-argument call can be used to move the selection to another widget or to set the owner to NULL, without changing the actual text of the selection.
- static int `Fl::selection_to_clipboard` ()
Returns the current selection_to_clipboard mode.
- static void `Fl::selection_to_clipboard` (int mode)
Copies selections on X11 directly to the clipboard if enabled.

Variables

- static char const *const `Fl::clipboard_image` = "image"
Denotes image data.
- static char const *const `Fl::clipboard_plain_text` = "text/plain"
Denotes plain textual data.

10.4.1 Detailed Description

FLTK global copy/cut/paste functions declared in `<FL/Fl.H>`

10.4.2 Function Documentation

10.4.2.1 `add_clipboard_notify()`

```
void Fl::add_clipboard_notify (
    FL_Clipboard_Notify_Handler h,
    void * data = 0) [static]
```

FLTK will call the registered callback whenever there is a change to the selection buffer or the clipboard. The source argument indicates which of the two has changed. Only changes by other applications are reported.

Example:

```
void clip_callback(int source, void *data) {
    if ( source == 0 ) printf("CLIP CALLBACK: selection buffer changed\n");
    if ( source == 1 ) printf("CLIP CALLBACK: clipboard changed\n");
}
[...]
```

```
int main() {
    [...]
    Fl::add_clipboard_notify(clip_callback);
    [...]
}
```

Note

Some systems require polling to monitor the clipboard and may therefore have some delay in detecting changes.

10.4.2.2 `clipboard_contains()`

```
int Fl::clipboard_contains (
    const char * type) [static]
```

Returns non 0 if the clipboard contains data matching `type`.

The clipboard can contain both text and image data; in that situation this function returns non 0 to both requests. This function is *not* meant to check whether the clipboard is empty. This function does not allow to query the selection buffer because FLTK allows to copy/paste non-textual data only from/to the clipboard.

Parameters

<code>type</code>	can be Fl::clipboard_plain_text or Fl::clipboard_image .

10.4.2.3 `copy()`

```
void Fl::copy (
    const char * stuff,
    int len,
    int destination = 0,
    const char * type = FL::clipboard_plain_text) [static]
```

Copies data to the selection buffer, the clipboard, or both.

The destination can be:

- 0: selection buffer (see note below)
- 1: clipboard
- 2: both

The selection buffer exists only on the X11 platform and is used for middle-mouse pastes and for drag-and-drop selections. The clipboard is used for traditional copy/cut/paste operations. On all other platforms the selection buffer (`destination = 0`) is mapped to the clipboard, i.e. on platforms other than X11 all `destinations` are equivalent and the data is always copied to the clipboard.

Note

Please see [Fl::selection_to_clipboard\(\)](#) to enable duplication of the selection buffer to the clipboard on X11, i.e. if `destination = 0` (selection buffer) **and** [Fl::selection_to_clipboard\(\)](#) is enabled, then the data is copied to both the selection buffer and the clipboard. This makes the X11 behavior similar to other platforms but keeps the selection buffer for X11 specific inter process communication.

`type` should always be [Fl::clipboard_plain_text](#) which is the default. Other values are ignored and reserved for future extensions.

Note

This function is, at present, intended only to copy UTF-8 encoded textual data. To copy graphical data, use the [FL_Copy_Surface](#) class. The `type` argument may allow to copy other kinds of data in the future.

Parameters

in	<i>stuff</i>	text data to be copied
in	<i>len</i>	the number of relevant bytes in <i>stuff</i>
in	<i>destination</i>	0 = selection, 1 = clipboard, 2 = both (see description)
in	<i>type</i>	usually plain text (see description)

10.4.2.4 dnd()

```
int Fl::dnd () [static]
```

Initiate a Drag And Drop operation.

The selection buffer should be filled with relevant data before calling this method. FLTK will then initiate the system wide drag and drop handling. Dropped data will be marked as *text*.

Create a selection first using: [Fl::copy\(const char *stuff, int len, 0\)](#)

10.4.2.5 paste() [1/2]

```
void Fl::paste (
    Fl_Widget & receiver) [static]
```

Backward compatibility only.

This calls [Fl::paste\(receiver, 0\)](#);

See also

[Fl::paste\(Fl_Widget &receiver, int clipboard, const char* type\)](#)

10.4.2.6 paste() [2/2]

```
void Fl::paste (
    Fl_Widget & receiver,
    int source,
    const char * type = Fl::clipboard_plain_text) [static]
```

Pastes the data from the selection buffer (`source` is 0) or the clipboard (`source` is 1) into `receiver`.

The selection buffer (`source` is 0) is used for middle-mouse pastes and for drag-and-drop selections. The clipboard (`source` is 1) is used for copy/cut/paste operations.

If `source` is 1, the optional `type` argument indicates what type of data is requested from the clipboard. At present, [Fl::clipboard_plain_text](#) (requesting text data) and [Fl::clipboard_image](#) (requesting image data) are possible. Set things up so the handle function of the `receiver` widget will be called with an `FL_PASTE` event some time in the future if the clipboard does contain data of the requested type.

The handle function of `receiver` can process the `FL_PASTE` event as follows:

- If the `receiver` widget is known to only receive text data, the text string from the specified `source` is in [Fl::event_text\(\)](#) with UTF-8 encoding, and the number of bytes is in [Fl::event_length\(\)](#). If [Fl::paste\(\)](#) gets called during the drop step of a files-drag-and-drop operation, [Fl::event_text\(\)](#) contains a list of filenames (see [Drag and Drop Events](#)).

- If the `receiver` widget can potentially receive non-text data, use `Fl::event_clipboard_type()` to determine what sort of data is being sent. If `Fl::event_clipboard_type()` returns `Fl::clipboard_plain_text`, proceed as above. If it returns `Fl::clipboard_image`, the pointer returned by `Fl::event_clipboard()` can be safely cast to type `Fl_RGB_Image *` to obtain a pointer to the pasted image. If `receiver` accepts the clipboard image, `receiver.handle()` should return 1 and the application should take ownership of this image (that is, delete it after use). Conversely, if `receiver.handle()` returns 0, the application must not use the image.

The receiver should be prepared to be called *directly* by this, or for it to happen *later*, or possibly *not at all*. This allows the window system to take as long as necessary to retrieve the paste buffer (or even to screw up completely) without complex and error-prone synchronization code in FLTK.

Platform details for image data:

- Unix/Linux platform: Clipboard images in PNG or BMP formats are recognized. Requires linking with the `fltk_images` library.
- Windows platform: Both bitmap and vectorial (Enhanced metafile) data from clipboard can be pasted as image data.
- Mac OS X platform: Both bitmap (TIFF) and vectorial (PDF) data from clipboard can be pasted as image data.

10.4.2.7 selection()

```
void Fl::selection (
    Fl_Widget & owner,
    const char * text,
    int len) [static]
```

Changes the current selection.

The block of text is copied to an internal buffer by FLTK (be careful if doing this in response to an `FL_PASTE` as this may be the same buffer returned by `event_text()`). The `selection_owner()` widget is set to the passed owner.

10.4.2.8 selection_owner() [1/2]

```
static Fl_Widget * Fl::selection_owner () [inline], [static]
```

back-compatibility only: Gets the widget owning the current selection

See also

`Fl_Widget*` [selection_owner\(Fl_Widget*\)](#)

10.4.2.9 selection_owner() [2/2]

```
void Fl::selection_owner (
    Fl_Widget * owner) [static]
```

Back-compatibility only: The single-argument call can be used to move the selection to another widget or to set the owner to NULL, without changing the actual text of the selection.

`FL_SELECTIONCLEAR` is sent to the previous selection owner, if any.

Copying the buffer every time the selection is changed is obviously wasteful, especially for large selections. An interface will probably be added in a future version to allow the selection to be made by a callback function. The current interface will be emulated on top of this.

10.4.2.10 selection_to_clipboard() [1/2]

```
static int Fl::selection_to_clipboard () [inline], [static]
```

Returns the current `selection_to_clipboard` mode.

See also

`void` [selection_to_clipboard\(int\)](#)

10.4.2.11 selection_to_clipboard() [2/2]

```
static void Fl::selection_to_clipboard (
    int mode) [inline], [static]
```

Copies selections on X11 directly to the clipboard if enabled.

This method can be called on all platforms. Other platforms than X11 are not affected by this feature.

If this is switched on (`mode = 1`), [Fl::copy\(\)](#) copies all data to the clipboard regardless of its `destination` argument. If the destination is 0 (selection buffer) data is copied to both the selection buffer and the clipboard.

Drag and drop is also affected since drag-and-drop data is copied to the selection buffer.

You can use this to make the experience of data selection and copying more like that on other platforms (Windows, macOS, and even Wayland).

The default operation mode is the standard X11 behavior (disabled).

Note

This feature is experimental and enabling it may have unexpected side effects. It is your own responsibility if you enable it.

Since

1.4.0

Parameters

<code>in</code>	<code>mode</code>	1 = enable <code>selection_to_clipboard</code> , 0 = disable <code>selection_to_clipboard</code>

See also

[copy\(const char *, int, int, const char *\)](#)

10.5 Screen functions

[Fl](#) global screen functions declared in `<FL/Fl.H>`.

Functions

- static int [Fl::h](#) ()
Returns the height in pixels of the main screen work area.
- static void [Fl::keyboard_screen_scaling](#) (int value)
Controls the possibility to scale all windows by ctrl/+/-/0/ or cmd/+/-/0/.
- static int [Fl::screen_count](#) ()
Gets the total count of available screens.
- static void [Fl::screen_dpi](#) (float &h, float &v, int n=0)
Gets the screen resolution in dots-per-inch for the given screen.
- static int [Fl::screen_num](#) (int x, int y)
Gets the screen number of a screen that contains the specified screen position x, y.
- static int [Fl::screen_num](#) (int x, int y, int w, int h)
Gets the screen number of the screen which intersects the most with the rectangle defined by x, y, w, h.
- static float [Fl::screen_scale](#) (int n)
Gets the GUI scaling factor of screen number n.
- static void [Fl::screen_scale](#) (int n, float factor)
Sets the GUI scaling factor of screen number n.
- static int [Fl::screen_scaling_supported](#) ()
Returns whether scaling factors are supported by this platform.
- static void [Fl::screen_work_area](#) (int &X, int &Y, int &W, int &H)

- Gets the bounding box of the work area of the screen that contains the mouse pointer.*
- static void [Fl::screen_work_area](#) (int &X, int &Y, int &W, int &H, int mx, int my)
- Gets the bounding box of the work area of a screen that contains the specified screen position `mx`, `my`.*
- static void [Fl::screen_work_area](#) (int &X, int &Y, int &W, int &H, int n)
- Gets the bounding box of the work area of the given screen.*
- static void [Fl::screen_xywh](#) (int &X, int &Y, int &W, int &H)
- Gets the bounding box of a screen that contains the mouse pointer.*
- static void [Fl::screen_xywh](#) (int &X, int &Y, int &W, int &H, int mx, int my)
- Gets the bounding box of a screen that contains the specified screen position `mx`, `my`.*
- static void [Fl::screen_xywh](#) (int &X, int &Y, int &W, int &H, int mx, int my, int mw, int mh)
- Gets the screen bounding rect for the screen which intersects the most with the rectangle defined by `mx`, `my`, `mw`, `mh`.*
- static void [Fl::screen_xywh](#) (int &X, int &Y, int &W, int &H, int n)
- Gets the screen bounding rect for the given screen.*
- static int [Fl::w](#) ()
- Returns the width in pixels of the main screen work area.*
- static int [Fl::x](#) ()
- Returns the leftmost x coordinate of the main screen work area.*
- static int [Fl::y](#) ()
- Returns the topmost y coordinate of the main screen work area.*

10.5.1 Detailed Description

[Fl](#) global screen functions declared in `<FL/Fl.H>`.

FLTK supports high-DPI screens using a screen scaling factor. The scaling factor is initialized by the library to a value based on information obtained from the OS. If this initial value is not satisfactory, the `FLTK_SCALING_FACTOR` environment variable can be set to a value FLTK will multiply to the OS-given value. The 2 variants of functions [Fl::screen_scale\(\)](#) allow to programmatically get and set scaling factor values. The scaling factor value can be further changed at runtime by typing `Ctrl/+/-/0/` (`Cmd/+/-/0/` under macOS). See [FL_SHORTCUT](#) for more details about these shortcuts.

10.5.2 Function Documentation

10.5.2.1 keyboard_screen_scaling()

```
void Fl::keyboard_screen_scaling (
    int value) [static]
```

Controls the possibility to scale all windows by `ctrl/+/-/0/` or `cmd/+/-/0/`.

This function **should** be called before [fl_open_display\(\)](#) runs. If it is not called, the default is to handle these keys for window scaling.

Note

This function can currently only be used to switch the internal handler **off**, i.e. `value` must be 0 (zero) - all other values result in undefined behavior and are reserved for future extension.

Parameters

<i>value</i>	0 to stop recognition of <code>ctrl/+/-/0/</code> (or <code>cmd/+/-/0/</code> under macOS) keys as window scaling.

Since

1.4.0

10.5.2.2 screen_count()

```
int Fl::screen_count () [static]
```

Gets the total count of available screens.

Note

Screen numbers range from 0 to [Fl::screen_count\(\)](#) - 1 in the FLTK API.

10.5.2.3 screen_dpi()

```
void Fl::screen_dpi (
    float & h,
    float & v,
    int n = 0) [static]
```

Gets the screen resolution in dots-per-inch for the given screen.

Parameters

out	<i>h,v</i>	horizontal and vertical resolution
in	<i>n</i>	the screen number (0 to Fl::screen_count() - 1)

See also

`void screen\_xywh(int &x, int &y, int &w, int &h, int mx, int my)`

10.5.2.4 screen_num() [1/2]

```
int Fl::screen_num (
    int x,
    int y) [static]
```

Gets the screen number of a screen that contains the specified screen position *x*, *y*.

Parameters

in	<i>x,y</i>	the absolute screen position

Returns

a screen number [0 , [Fl::screen_count\(\)](#)-1]

Attention

When the running system contains screens with different scaling factors, this API may become ambiguous because a given value pair (*x*, *y*) may belong to distinct screens. In that situation other APIs should be preferred, e.g. [Fl_Window::screen_num\(\)](#) and [Fl::screen_scale\(int\)](#).

10.5.2.5 screen_num() [2/2]

```
int Fl::screen_num (
    int x,
    int y,
    int w,
    int h) [static]
```

Gets the screen number of the screen which intersects the most with the rectangle defined by *x*, *y*, *w*, *h*.

Parameters

in	<i>x,y,w,h</i>	the rectangle to search for intersection with

Returns

a screen number [0 , [Fl::screen_count\(\)](#)-1]

10.5.2.6 screen_scale() [1/2]

```
float Fl::screen_scale (
    int n) [static]
```

Gets the GUI scaling factor of screen number *n*.

The valid range of *n* is 0 .. [Fl::screen_count\(\)](#) - 1.

The return value is 1.0 if screen scaling is not supported or *n* is outside the valid range.

Returns

Current screen scaling factor (default: 1.0)

See also

[Fl::screen_count\(\)](#)

[Fl::screen_scaling_supported\(\)](#)

Since

1.4.0

10.5.2.7 screen_scale() [2/2]

```
void Fl::screen_scale (
    int n,
    float factor) [static]
```

Sets the GUI scaling factor of screen number *n*.

The valid range of *n* is 0 .. [Fl::screen_count\(\)](#) - 1.

This method does nothing if *n* is out of range or screen scaling is not supported by this platform.

Otherwise it also sets the scaling factor of all windows mapped to screen number *n* or all screens, depending on the type of screen scaling support on the platform.

Parameters

in	<i>n</i>	screen number
in	<i>factor</i>	scaling factor of screen <i>n</i>

See also

[Fl::screen_scaling_supported\(\)](#)

Since

1.4.0

10.5.2.8 screen_scaling_supported()

```
int Fl::screen_scaling_supported () [static]
```

Returns whether scaling factors are supported by this platform.

Return values

0	scaling factors are not supported by this platform
1	a single scaling factor is shared by all screens
2	each screen can have its own scaling factor

See also

[Fl::screen_scale\(int\)](#)

Since

1.4.0

10.5.2.9 screen_work_area() [1/3]

```
void Fl::screen_work_area (
    int & X,
    int & Y,
    int & W,
    int & H) [static]
```

Gets the bounding box of the work area of the screen that contains the mouse pointer.

Parameters

out	<i>X,Y,W,H</i>	the work area bounding box

See also

void [screen_work_area\(int &x, int &y, int &w, int &h, int n\)](#)

10.5.2.10 screen_work_area() [2/3]

```
void Fl::screen_work_area (
    int & X,
    int & Y,
    int & W,
    int & H,
    int mx,
    int my) [static]
```

Gets the bounding box of the work area of a screen that contains the specified screen position *mx*, *my*.

Parameters

out	<i>X,Y,W,H</i>	the work area bounding box
in	<i>mx,my</i>	the absolute screen position

See also

void [screen_work_area\(int &x, int &y, int &w, int &h, int n\)](#)

10.5.2.11 screen_work_area() [3/3]

```
void Fl::screen_work_area (
    int & X,
    int & Y,
    int & W,
    int & H,
    int n) [static]
```

Gets the bounding box of the work area of the given screen.

Parameters

out	<i>X,Y,W,H</i>	the work area bounding box
in	<i>n</i>	the screen number (0 to Fl::screen_count() - 1)

See also

void [screen_xywh](#)(int &x, int &y, int &w, int &h, int mx, int my)

Note

Under X11, the screen work area is given values that differ from that screen's bounding box only if the system contains a single screen. Under Wayland, a screen work area is always equal to that screen's bounding box.

Like all quantities accessible via public APIs of FLTK, values of *x,y,w,h* are given in FLTK units, that is, in drawing units divided by the scaling factor of screen *n*.

10.5.2.12 screen_xywh() [1/4]

```
void Fl::screen_xywh (
    int & X,
    int & Y,
    int & W,
    int & H) [static]
```

Gets the bounding box of a screen that contains the mouse pointer.

Parameters

out	<i>X,Y,W,H</i>	the corresponding screen bounding box

See also

void [screen_xywh](#)(int &x, int &y, int &w, int &h, int mx, int my)

10.5.2.13 screen_xywh() [2/4]

```
void Fl::screen_xywh (
    int & X,
    int & Y,
    int & W,
    int & H,
    int mx,
    int my) [static]
```

Gets the bounding box of a screen that contains the specified screen position *mx, my*.

Parameters

out	<i>X,Y,W,H</i>	the corresponding screen bounding box
in	<i>mx,my</i>	the absolute screen position

10.5.2.14 screen_xywh() [3/4]

```
void Fl::screen_xywh (
    int & X,
    int & Y,
    int & W,
```



```

    int & H,
    int mx,
    int my,
    int mw,
    int mh) [static]

```

Gets the screen bounding rect for the screen which intersects the most with the rectangle defined by `mx`, `my`, `mw`, `mh`.

Parameters

out	<i>X,Y,W,H</i>	the corresponding screen bounding box
in	<i>mx,my,mw,mh</i>	the rectangle to search for intersection with

See also

void [screen_xywh](#)(int &X, int &Y, int &W, int &H, int n)

10.5.2.15 screen_xywh() [4/4]

```

void Fl::screen_xywh (
    int & X,
    int & Y,
    int & W,
    int & H,
    int n) [static]

```

Gets the screen bounding rect for the given screen.

Under Windows, Mac OS X, and X11 + the Gnome desktop, screen #0 contains the menubar/taskbar

Parameters

out	<i>X,Y,W,H</i>	the corresponding screen bounding box
in	<i>n</i>	the screen number (0 to Fl::screen_count() - 1)

Note

Like all quantities accessible via public APIs of FLTK, values of `X,Y,W,H` are given in FLTK units, that is, in drawing units divided by the scaling factor of screen `n`.

See also

void [screen_xywh](#)(int &x, int &y, int &w, int &h, int mx, int my)

10.6 Color & Font functions

fl global color, font functions.

Functions

- [Fl_Color fl_color](#) ()
Return the last [fl_color\(\)](#) that was set.
- void [fl_color](#) ([Fl_Color](#) c)
Set the color for all subsequent drawing operations.
- void [fl_color](#) (int c)
for back compatibility - use [fl_color\(Fl_Color c\)](#) instead
- void [fl_color](#) (uchar r, uchar g, uchar b)

- Set the color for all subsequent drawing operations.*

 - `FI_Color fl_color_average` (`FI_Color` color1, `FI_Color` color2, float weight)

Returns the weighted average color between the two given colors.
- `FI_Color fl_contrast` (`FI_Color` fg, `FI_Color` bg, int context, int size)

Returns a color that contrasts with the background color.
- void `fl_contrast_function` (`FI_Contrast_Function` *f)

Register a custom contrast function.
- int `fl_contrast_level` ()

Get the contrast level (sensitivity) of the `fl_contrast()` method.
- void `fl_contrast_level` (int level)

Set the contrast level (sensitivity) of the `fl_contrast()` method.
- int `fl_contrast_mode` ()

Return the current contrast algorithm (mode).
- void `fl_contrast_mode` (int mode)

Set the contrast algorithm (mode).
- int `fl_descent` ()

Return the recommended distance above the bottom of a `fl_height()` tall box to draw the text at so it looks centered vertically in that box.
- `FI_Font fl_font` ()

Return the `face` set by the most recent call to `fl_font()`.
- void `fl_font` (`FI_Font` face, `FI_Fontsize` fsize)

Sets the current font, which is then used in various drawing routines.
- int `fl_height` ()

Return the recommended minimum line spacing for the current font.
- int `fl_height` (int font, int size)

This function returns the actual height of the specified `font` and `size`.
- `FI_Color fl_inactive` (`FI_Color` c)

Returns the inactive, dimmed version of the given color.
- const char * `fl_latin1_to_local` (const char *t, int n=-1)

Convert text from Windows/X11 latin1 character set to local encoding.
- double `fl_lightness` (`FI_Color` color)

Return the perceived lightness of a color.
- const char * `fl_local_to_latin1` (const char *t, int n=-1)

Convert text from local encoding to Windows/X11 latin1 character set.
- const char * `fl_local_to_mac_roman` (const char *t, int n=-1)

Convert text from local encoding to Mac Roman character set.
- double `fl_luminance` (`FI_Color` color)

Return the raw / physical luminance of a color.
- const char * `fl_mac_roman_to_local` (const char *t, int n=-1)

Convert text from Mac Roman character set to local encoding.
- `FI_Color fl_show_colormap` (`FI_Color` oldcol)

Pops up a window to let the user pick a colormap entry.
- `FI_Fontsize fl_size` ()

Return the `size` set by the most recent call to `fl_font()`.
- void `fl_text_extents` (const char *, int &dx, int &dy, int &w, int &h)

Determine the minimum pixel dimensions of a nul-terminated string using the current `fl_font()`.
- void `fl_text_extents` (const char *t, int n, int &dx, int &dy, int &w, int &h)

Determine the minimum pixel dimensions of a sequence of `n` characters (bytes) using the current `fl_font()`.
- double `fl_width` (const char *txt)

Return the typographical width of a nul-terminated string using the current font face and size.
- double `fl_width` (const char *txt, int n)

- Return the typographical width of a sequence of n characters using the current font face and size.*
- double `fl_width` (unsigned int `c`)
Return the typographical width of a single character using the current font face and size.
- static void `Fl::free_color` (`Fl_Color` `i`, int `overlay=0`)
Frees the specified color from the colormap, if applicable.
- static unsigned `Fl::get_color` (`Fl_Color` `i`)
Returns the RGB value(s) for the given FLTK color index.
- static void `Fl::get_color` (`Fl_Color` `i`, `uchar` &`red`, `uchar` &`green`, `uchar` &`blue`)
Returns the RGB value(s) for the given FLTK color index.
- static void `Fl::get_color` (`Fl_Color` `i`, `uchar` &`red`, `uchar` &`green`, `uchar` &`blue`, `uchar` &`alpha`)
Returns the RGBA value(s) for the given FLTK color index.
- static const char * `Fl::get_font` (`Fl_Font`)
Gets the string for this face.
- static const char * `Fl::get_font_name` (`Fl_Font`, int *`attributes=0`)
Get a human-readable string describing the family of this face.
- static int `Fl::get_font_sizes` (`Fl_Font`, int *`&sizep`)
Return an array of sizes in `sizep`.
- static void `Fl::set_color` (`Fl_Color` `i`, unsigned `c`)
Sets an entry in the `fl_color` index table.
- static void `Fl::set_color` (`Fl_Color`, `uchar`, `uchar`, `uchar`)
Sets an entry in the `fl_color` index table.
- static void `Fl::set_color` (`Fl_Color`, `uchar`, `uchar`, `uchar`, `uchar`)
Sets an entry in the `fl_color` index table.
- static void `Fl::set_font` (`Fl_Font`, const char *)
Changes a face.
- static void `Fl::set_font` (`Fl_Font`, `Fl_Font`)
Copies one face to another.
- static `Fl_Font` `Fl::set_fonts` (const char *`=0`)
FLTK will open the display, and add every fonts on the server to the face table.

10.6.1 Detailed Description

`fl` global color, font functions.

These functions are declared in `<FL/Fl.H>` or `<FL/fl_draw.H>`.

10.6.2 Function Documentation

10.6.2.1 `fl_color()` [1/3]

```
Fl_Color fl_color () [inline]
```

Return the last `fl_color()` that was set.

This can be used for state save/restore.

10.6.2.2 `fl_color()` [2/3]

```
void fl_color (
    Fl_Color c) [inline]
```

Set the color for all subsequent drawing operations.

For color-mapped displays, a color cell will be allocated out of `fl_colormap` the first time you use a color. If the colormap fills up then a least-squares algorithm is used to find the closest color. If no valid graphical context (`fl_gc`) is available, the foreground is not set for the current window.

Parameters

<code>in</code>	<code>c</code>	<code>color</code>

10.6.2.3 fl_color() [3/3]

```
void fl_color (
    uchar r,
    uchar g,
    uchar b) [inline]
```

Set the color for all subsequent drawing operations.

The closest possible match to the RGB color is used. The RGB color is used directly on TrueColor displays. For colormap visuals the nearest index in the gray ramp or color cube is used. If no valid graphical context (fl_gc) is available, the foreground is not set for the current window.

Parameters

in	<i>r,g,b</i>	color components

10.6.2.4 fl_color_average()

```
Fl_Color fl_color_average (
    Fl_Color color1,
    Fl_Color color2,
    float weight)
```

Returns the weighted average color between the two given colors.

The red, green and blue values are averages using the following formula:

```
color = color1 * weight + color2 * (1 - weight)
```

Thus, a *weight* value of 1.0 will return the first color, while a value of 0.0 will return the second color.

Parameters

in	<i>color1,color2</i>	boundary colors
in	<i>weight</i>	weighting factor

10.6.2.5 fl_contrast()

```
Fl_Color fl_contrast (
    Fl_Color fg,
    Fl_Color bg,
    int context,
    int size)
```

Returns a color that contrasts with the background color.

This will be the foreground color if it contrasts sufficiently with the background color. Otherwise, returns `FL_WHITE` or `FL_BLACK` depending on which color provides the best contrast.

FLTK 1.4.0 uses a different default contrast function than earlier releases (1.3.x) but you can use the old "legacy" contrast function by calling

```
fl_contrast_mode(FL_CONTRAST_LEGACY);
```

early in your main program.

Note

It is a known issue that static initialization using `fl_contrast()` may already have been executed before you call this function in `main()`. You should be aware of this and, if necessary, write your own (static) contrast initialization function. This should rarely be necessary.

You can change the behavior of `fl_contrast()` in several ways:

- Change the "level" (sensitivity) for contrast calculation, see `fl_contrast_level()`. Valid levels are 0 - 100, the default "medium" value depends on the contrast mode. If you raise the level above the default value the overall contrast will generally be higher, i.e. the required contrast to return the foreground color is raised and therefore the calculated color switches "earlier" to either black or white. In other words, using the following values:

- 0 always uses the foreground color
- the default, unmodified algorithm allows a sufficient contrast such that the text is readable
- 100 will always use black or white

Changing the `level` is particularly useful and intended for the "legacy mode" to improve the results partially. Values slightly above 50 (50 - 70) will likely return the best results (50 is the default, as used in FLTK 1.3.x).

Note

Different contrast modes (algorithms) can use their own values and defaults of `fl_contrast_level()`.

- Change the used contrast calculation function. You can either use the old (FLTK 1.3.x) function or use the better but slower function based on the CIELAB ($L^*a^*b^*$) color model, or you can define your own custom contrast function if you need even better contrast results.

The following contrast functions are available:

- `FL_CONTRAST_LEGACY`, the old FLTK 1.3.x compatible function. This is the fastest function (using integer arithmetic) but it provides worse results in border cases. You may want to use this on embedded or otherwise CPU constrained systems or if you need strict backwards compatibility. For slightly better results you may utilize the new `fl_contrast_level(int)` function (since 1.4.0) to increase the contrast sensitivity. This will provide slightly better results than FLTK 1.3.x and earlier but we recommend to use the new default function:
- `FL_CONTRAST_CIELAB`, based on the CIELAB ($L^*a^*b^*$) color model. This function is superior regarding the human contrast perception but may be slightly slower - which should not matter on a modern CPU. The default contrast level in this mode is 39 which results in a very similar experience as the old contrast function but avoids unreadable border cases. **This is the default since FLTK 1.4.0.**
- `FL_CONTRAST_CUSTOM`, your own contrast calculation function.

In the future we **may** provide even more (and superior) contrast algorithms.

The new parameters `context` and `size` (since 1.4.0) are defined for future extensions and are currently not used. Default values are 0.

- The `context` is intended to differentiate text and other kinds of objects, e.g. radio buttons, check marks, or icon types.
- The `size` parameter is an unspecified (object) size that may be used to calculate the required contrast. In text mode this must be the font size. Rule: the larger the object (font), the lower the required contrast.

Note

These new optional parameters must be provided in the custom contrast function which is the reason why they are added now. In the future we may use the (font) size to adjust the calculated contrast, and users defining their own contrast functions may use them in their functions.

Parameters

in	<i>fg</i>	foreground (text) color
in	<i>bg</i>	background color
in	<i>context</i>	graphical context (optional, default = 0 == text)
in	<i>size</i>	unspecified size (optional, default = 0 == undefined)

Returns

contrasting color: `fg`, `FL_BLACK`, or `FL_WHITE`

See also

[fl_contrast_level\(int\)](#)
[fl_contrast_mode\(int\)](#)
[fl_contrast_function\(\)](#)

10.6.2.6 fl_contrast_function()

```
void fl_contrast_function (
    Fl_Contrast_Function * f)
```

Register a custom contrast function.

Your custom contrast function will be called when [fl_contrast\(\)](#) is called if and only if you previously registered your function and called `fl_contrast_mode(FL_CONTRAST_CUSTOM)`.

Your custom contrast function must provide the signature

```
Fl_Color my_contrast_function(Fl_Color fg, Fl_Color bg, int context, int size)
```

The arguments are the same as for the full [fl_contrast\(\)](#) function since FLTK 1.4. You can use the supplied `size` to modify the result. Depending on the caller the `size` parameter can be 0 (default) or a valid size. In the context of text, i.e. `context == 0`, the `size` parameter is the fontsize.

The `context` parameter is not yet used and will always be 0 unless included in a call to [fl_contrast\(\)](#). The value 0 must be interpreted as text. In the future the `context` argument will be used to supply a different context than text (small icons, large icons, etc.). The exact usage is not yet specified.

Your function may also use [fl_contrast_level\(\)](#) to modify the result accordingly.

Since

1.4.0

See also

[fl_contrast_mode\(int\)](#)
[fl_contrast_level\(int\)](#)
[fl_contrast\(\)](#)

10.6.2.7 fl_contrast_level() [1/2]

```
int fl_contrast_level ()
```

Get the contrast level (sensitivity) of the [fl_contrast\(\)](#) method.

This returns the level of the currently selected contrast mode.

Returns

The current contrast level.

See also

[fl_contrast_level\(int level\)](#)
[fl_contrast_mode\(int mode\)](#)

Since

1.4.0

10.6.2.8 fl_contrast_level() [2/2]

```
void fl_contrast_level (
    int level)
```

Set the contrast level (sensitivity) of the [fl_contrast\(\)](#) method.

This can be used to tune the legacy [fl_contrast\(\)](#) function to achieve slightly better results. The default value is defined per contrast mode (see below). Values between 50 and 70 may be useful for the legacy contrast mode but you can raise it up to 100. Lower values than 50 are probably not useful.

The contrast `level` affects not only the legacy (1.3.x) [fl_contrast\(\)](#) function but also the new CIELAB contrast mode which is the default since FLTK 1.4.0. See default value below.

Other contrast modes are currently not affected by the contrast level.

You may use the contrast level if you define your own custom contrast function in mode `FL_CONTRAST_CUSTOM`.

Note

All contrast modes store their own contrast level because the behavior is slightly different. You must change the contrast mode `fl_contrast_mode()` **before** you set or get the contrast level.

The default contrast level is

- 50 in mode `FL_CONTRAST_LEGACY` (compatible with FLTK 1.3.x)
- 39 in mode `FL_CONTRAST_CIELAB` (similar threshold as in FLTK 1.3.x)
- 0 (undefined) for all other modes

See the description of `fl_contrast_mode(int mode)` for more information about the contrast level per mode.

Example:

```
fl_contrast_mode(FL_CONTRAST_LEGACY);
fl_contrast_level(60);
```

A level greater than 50 (probably best in the range 50 to 70) may achieve better results of the legacy `fl_contrast()` function in some border cases of low contrast between foreground and background colors but we recommend to use the new default algorithm `FL_CONTRAST_CIELAB` unless you need strict backwards compatibility or use a CPU constrained embedded system.

Parameters

in	<i>level</i>	valid range is 0 to 100

Since

1.4.0

10.6.2.9 fl_contrast_mode() [1/2]

```
int fl_contrast_mode ()
```

Return the current contrast algorithm (mode).

Returns

Contrast algorithm (mode).

Since

1.4.0

See also

[fl_contrast_mode\(int\)](#)

10.6.2.10 fl_contrast_mode() [2/2]

```
void fl_contrast_mode (
    int mode)
```

Set the contrast algorithm (mode).

You can use one of

- `FL_CONTRAST_NONE` (not recommended: returns the foreground color)
- `FL_CONTRAST_LEGACY` (same as in FLTK 1.3.x)
- `FL_CONTRAST_CIELAB` (better, this is the default since FLTK 1.4.0)
- `FL_CONTRAST_CUSTOM` (you must define your own contrast algorithm)

If you set `FL_CONTRAST_CUSTOM` you must also register your custom contrast function by calling `fl_contrast_function()`.

You may set the contrast level `fl_contrast_level(int)` after setting the contrast mode. This affects the contrast algorithm as described below:

- `FL_CONTRAST_LEGACY`: default level is 50 which is compatible with FLTK 1.3.x and older. This mode is no longer the default and is not recommended because it doesn't take human contrast perception into account and doesn't properly handle sRGB color values. You may get better contrasts if you set the level higher than 50. Values in the range 50 to 70 may be useful. Higher values result in higher contrast, i.e. the algorithm switches "earlier" to black or white mode.
- `FL_CONTRAST_CIELAB`: default level is 39 which appears to be a good value. The higher the level is, the more contrast is to be expected. Values in the range below 39 accept lower contrast and values above 39 switch "earlier" to black or white. Values between 36 and 46 may yield usable contrast experience.

Note

The goal of `fl_contrast()` is to achieve a "sufficient" contrast between text and background. Level 39 in CIELAB mode means that the accepted contrast is about 39% of the lightness difference between both colors. This can be perceived as very low contrast in some cases, but the text should at least be readable. Note that the highest possible contrast value on a medium gray background is 50% (either black or white). Bill Spitzak wrote on May 16, 2024 in fltk.general in thread "FLTK 1.4 Menu Bar Style": *"I would certainly aim for a function that does not alter color combinations where it is physically possible to read the text, even if squinting is needed."* See <https://groups.google.com/g/fltkgeneral/c/EkWI4HTHSLA/m/rsZunZ1vAwAJ>

Parameters

in	mode	if invalid, FL_CONTRAST_CIELAB will be selected

Since

1.4.0

See also

[fl_contrast_function\(Fl_Contrast_Function *\)](#)

[fl_contrast_level\(int\)](#)

10.6.2.11 fl_font() [1/2]

```
Fl_Font fl_font () [inline]
```

Return the `face` set by the most recent call to `fl_font()`.

This can be used to save/restore the font.

10.6.2.12 fl_font() [2/2]

```
void fl_font (
    Fl_Font face,
    Fl_Fontsize fsize)
```

Sets the current font, which is then used in various drawing routines.

You may call this outside a draw context if necessary to measure text, for instance by calling `fl_width()`, `fl_measure()`, or `fl_text_extents()`, but on X this will open the display.

The font is identified by a `face` and a `size`. The size of the font is measured in pixels and not "points". Lines should be spaced `size` pixels apart or more.

10.6.2.13 fl_height() [1/2]

```
int fl_height () [inline]
```

Return the recommended minimum line spacing for the current font.

You can also use the value of `size` passed to `fl_font()`.

10.6.2.14 fl_height() [2/2]

```
int fl_height (
    int font,
    int size)
```

This function returns the actual height of the specified `font` and `size`.

Normally the font height should always be 'size', but with the advent of XFT, there are (currently) complexities that seem to only be solved by asking the font what its actual font height is. (See STR#2115)

This function was originally undocumented in 1.1.x, and was used only by [Fl_Text_Display](#). We're now documenting it in 1.3.x so that apps that need precise height info can get it with this function.

Returns

the height of the font in pixels.

Todo In the future, when the XFT issues are resolved, this function should simply return the 'size' value.

10.6.2.15 fl_latin1_to_local()

```
const char * fl_latin1_to_local (
    const char * t,
    int n = -1)
```

Convert text from Windows/X11 latin1 character set to local encoding.

Parameters

in	<i>t</i>	character string (latin1 encoding)
in	<i>n</i>	optional number of characters (bytes) to convert (default is all)

Returns

pointer to internal buffer containing converted characters

10.6.2.16 fl_lightness()

```
double fl_lightness (
    Fl_Color color)
```

Return the perceived lightness of a color.

This function calculates the perceived lightness of `Fl_Color color`.

The returned lightness value `Lstar` according to the CIELAB ($L^*a^*b^*$) color model is almost linear with respect to human perception. It is in the range 0 (black) to 100 (white).

The result values of two colors can be compared directly and the difference is their perceived contrast.

Parameters

in	<i>color</i>	Fl_Color value

Returns

perceived lightness (0 .. 100)

Since

1.4.0

10.6.2.17 fl_local_to_latin1()

```
const char * fl_local_to_latin1 (
    const char * t,
    int n = -1)
```

Convert text from local encoding to Windows/X11 latin1 character set.

Parameters

in	<i>t</i>	character string (local encoding)
in	<i>n</i>	optional number of characters (bytes) to convert (default is all)

Returns

pointer to internal buffer containing converted characters

10.6.2.18 fl_local_to_mac_roman()

```
const char * fl_local_to_mac_roman (
    const char * t,
    int n = -1)
```

Convert text from local encoding to Mac Roman character set.

Parameters

in	<i>t</i>	character string (local encoding)
in	<i>n</i>	optional number of characters to convert (default is all)

Returns

pointer to internal buffer containing converted characters

10.6.2.19 fl_luminance()

```
double fl_luminance (
    Fl_Color color)
```

Return the raw / physical luminance of a color.

This function calculates the physical luminance of `Fl_Color color`.

The returned luminance value (aka Y) is the physical luminance of the `Fl_Color color`.

The result is in the range 0.0 (black) to 1.0 (white).

Note

This is probably not what you want if you are interested in perceived contrast or lightness calculation because the luminance Y is **not** linear with respect to human perception.

See [fl_lightness\(Fl_Color\)](#) for a function that returns the perceived lightness of a color which can be used directly for contrast calculation.

Parameters

in	<i>color</i>	Fl_Color value

Returns

Raw (physical) luminance (0.0 .. 1.0)

Since

1.4.0

See also

[fl_lightness\(Fl_Color\)](#)

10.6.2.20 fl_mac_roman_to_local()

```
const char * fl_mac_roman_to_local (
    const char * t,
    int n = -1)
```

Convert text from Mac Roman character set to local encoding.

Parameters

in	<i>t</i>	character string (Mac Roman encoding)
in	<i>n</i>	optional number of characters to convert (default is all)

Returns

pointer to internal buffer containing converted characters

10.6.2.21 fl_show_colormap()

```
Fl_Color fl_show_colormap (
    Fl_Color oldcol)
```

Pops up a window to let the user pick a colormap entry.



Figure 10.1 fl_show_colormap

Parameters

in	<i>oldcol</i>	color to be highlighted when grid is shown.

Return values

<i>Fl_Color</i>	value of the chosen colormap entry.

See also

[Fl_Color_Chooser](#)

10.6.2.22 fl_size()

```
Fl_Fontsize fl_size () [inline]
```

Return the size set by the most recent call to [fl_font\(\)](#).

This can be used to save/restore the font.

10.6.2.23 fl_text_extents() [1/2]

```
void fl_text_extents (
    const char * c,
    int & dx,
    int & dy,
    int & w,
    int & h)
```

Determine the minimum pixel dimensions of a nul-terminated string using the current [fl_font\(\)](#).

Usage: given a string "txt" drawn using `fl_draw(txt, x, y)` you would determine its pixel extents on the display using `fl_text_extents(txt, dx, dy, wo, ho)` such that a bounding box that exactly fits around the text could be drawn with `fl_rect(x+dx, y+dy, wo, ho)`. Note the dx, dy values hold the offset of the first "colored in" pixel of the string, from the draw origin.

Note the desired font and font size must be set with [fl_font\(\)](#) before calling this function.

This differs slightly from [fl_measure\(\)](#) in that the dx/dy values are also returned.

No FLTK symbol expansion will be performed.

Example use:

```
int dx,dy,W,H;
fl_font(FL_HELVETICA, 12); // set font face+size first
fl_text_extents("Some text", dx, dy, W, H); // get width and height of string
printf("text's width=%d, height=%d\n", W, H);
```

10.6.2.24 fl_text_extents() [2/2]

```
void fl_text_extents (
    const char * t,
    int n,
    int & dx,
    int & dy,
    int & w,
    int & h) [inline]
```

Determine the minimum pixel dimensions of a sequence of n characters (bytes) using the current [fl_font\(\)](#).

Note

The string length is measured in bytes, not (UTF-8) characters.

See also

[fl_text_extents\(const char*, int& dx, int& dy, int& w, int& h\)](#)

10.6.2.25 fl_width()

```
double fl_width (
    unsigned int c) [inline]
```

Return the typographical width of a single character using the current font face and size.

Note

If a valid `fl_gc` is NOT found then it uses the first window `gc`, or the screen `gc` if no fltk window is available when called.

10.6.2.26 free_color()

```
void Fl::free_color (
    Fl_Color i,
    int overlay = 0) [static]
```

Frees the specified color from the colormap, if applicable.

If overlay is non-zero then the color is freed from the overlay colormap.

10.6.2.27 get_color() [1/3]

```
unsigned Fl::get_color (
    Fl_Color i) [static]
```

Returns the RGB value(s) for the given FLTK color index.

This form returns the RGB values packed in a 32-bit unsigned integer with the red value in the upper 8 bits, the green value in the next 8 bits, and the blue value in bits 8-15. The lower 8 bits will always be 0.

10.6.2.28 get_color() [2/3]

```
void Fl::get_color (
    Fl_Color i,
    uchar & red,
    uchar & green,
    uchar & blue) [static]
```

Returns the RGB value(s) for the given FLTK color index.

This form returns the red, green, and blue values separately in referenced variables.

See also

unsigned [get_color\(Fl_Color c\)](#)

10.6.2.29 get_color() [3/3]

```
void Fl::get_color (
    Fl_Color i,
    uchar & red,
    uchar & green,
    uchar & blue,
    uchar & alpha) [static]
```

Returns the RGBA value(s) for the given FLTK color index.

This form returns the red, green, blue, and alpha values separately in referenced variables.

See also

unsigned [get_color\(Fl_Color c\)](#)

10.6.2.30 get_font()

```
const char * Fl::get_font (
    Fl_Font fnum) [static]
```

Gets the string for this face.

This string is different for each face. Under X this value is passed to XListFonts to get all the sizes of this face.

10.6.2.31 get_font_name()

```
const char * Fl::get_font_name (
    Fl_Font fnum,
    int * attributes = 0) [static]
```

Get a human-readable string describing the family of this face.

This is useful if you are presenting a choice to the user. There is no guarantee that each face has a different name. The return value points to a static buffer that is overwritten each call.

The integer pointed to by `attributes` (if the pointer is not zero) is set to zero, `FL_BOLD` or `FL_ITALIC` or `FL_↵_BOLD | FL_ITALIC`. To locate a "family" of fonts, search forward and back for a set with non-zero attributes, these faces along with the face with a zero attribute before them constitute a family.

10.6.2.32 `get_font_sizes()`

```
int Fl::get_font_sizes (
    Fl_Font fnum,
    int *& sizep) [static]
```

Return an array of sizes in `sizep`.

The return value is the length of this array. The sizes are sorted from smallest to largest and indicate what sizes can be given to `fl_font()` that will be matched exactly (`fl_font()` will pick the closest size for other sizes). A zero in the first location of the array indicates a scalable font, where any size works, although the array may list sizes that work "better" than others. Warning: the returned array points at a static buffer that is overwritten each call. Under X this will open the display.

10.6.2.33 `set_color()` [1/3]

```
void Fl::set_color (
    Fl_Color i,
    unsigned c) [static]
```

Sets an entry in the `fl_color` index table.

You can set it to any 8-bit RGB color. The color is not allocated until `fl_color(i)` is used.

10.6.2.34 `set_color()` [2/3]

```
void Fl::set_color (
    Fl_Color i,
    uchar red,
    uchar green,
    uchar blue) [static]
```

Sets an entry in the `fl_color` index table.

You can set it to any 8-bit RGB color. The color is not allocated until `fl_color(i)` is used.

10.6.2.35 `set_color()` [3/3]

```
void Fl::set_color (
    Fl_Color i,
    uchar red,
    uchar green,
    uchar blue,
    uchar alpha) [static]
```

Sets an entry in the `fl_color` index table.

You can set it to any 8-bit RGBA color.

Note

The color transparency is effective under the Wayland, hybrid Wayland/X11 and macOS platforms, whereas it has no effect under the X11 and Windows platforms. It's also effective for widgets added to an [FL_Gl_Window](#).

Version

1.4

10.6.2.36 `set_font()`

```
void Fl::set_font (
    Fl_Font fnum,
    const char * name) [static]
```

Changes a face.

Parameters

<i>fnum</i>	The font number to be assigned a new face
<i>name</i>	Name of the font to assign. The string pointer is simply stored, the string is not copied, so the string must be in static memory. The exact name to be used depends on the platform :

- Windows, X11, Xft: use the family name prefixed by one character to indicate the desired font variant. Characters ' ', 'I', 'B', 'P' denote plain, italic, bold, and bold-italic variants, respectively. For example, string "IGabriola" is to be used to denote the "Gabriola italic" font. The "Oblique" suffix, in whatever case, is to be treated as "italic", that is, prefix the family name with 'I'.
- Other platforms, i.e., X11 + Pango, Wayland, macOS: use the full font name as returned by function [Fl::get_font_name\(\)](#) or as listed by applications test/fonts or test/utf8. No prefix is to be added.

10.6.2.37 set_fonts()

```
Fl_Font Fl::set_fonts (
    const char * xstarname = 0) [static]
```

FLTK will open the display, and add every fonts on the server to the face table.

It will attempt to put "families" of faces together, so that the normal one is first, followed by bold, italic, and bold italic. The only argument to this function is somewhat obsolete since FLTK and most underlying operating systems move to support Unicode. For completeness, following is the original documentation and a few updates:

On X11, the optional argument is a string to describe the set of fonts to add. Passing NULL will select only fonts that have the ISO8859-1 character set (and are thus usable by normal text). Passing "-*" will select all fonts with any encoding as long as they have normal X font names with dashes in them. Passing "*" will list every font that exists (on X this may produce some strange output). Other values may be useful but are system dependent.

With the Xft option on Linux, this parameter is ignored.

With Windows, NULL selects fonts with ANSI_CHARSET encoding and non-NULL selects all fonts.

On macOS, this parameter is ignored.

The return value is how many faces are in the table after this is done.

10.7 Drawing functions

FLTK global graphics and GUI drawing functions.

Enumerations

- enum {
[FL_SOLID](#) = 0 , [FL_DASH](#) = 1 , [FL_DOT](#) = 2 , [FL_DASHDOT](#) = 3 ,
[FL_DASHDOTDOT](#) = 4 , [FL_CAP_FLAT](#) = 0x100 , [FL_CAP_ROUND](#) = 0x200 , [FL_CAP_SQUARE](#) = 0x300 ,
[FL_JOIN_MITER](#) = 0x1000 , [FL_JOIN_ROUND](#) = 0x2000 , [FL_JOIN_BEVEL](#) = 0x3000 }

Functions

- int [fl_add_symbol](#) (const char *name, void(*drawit)([Fl_Color](#)), int scalable)
Adds a symbol to the system.
- int [fl_antialias](#) ()
Return whether line drawings are currently antialiased.
- void [fl_antialias](#) (int state)
Turn antialiased line drawings ON or OFF, if supported by platform.
- void [fl_arc](#) (double x, double y, double r, double start, double end)
Add a series of points to the current path on the arc of a circle.
- void [fl_arc](#) (int x, int y, int w, int h, double a1, double a2)
Draw ellipse sections using integer coordinates.

- void [fl_begin_complex_polygon](#) ()
Start drawing a complex filled polygon.
- void [fl_begin_line](#) ()
Start drawing a list of lines.
- void [fl_begin_loop](#) ()
Start drawing a closed sequence of lines.
- void [fl_begin_offscreen](#) ([Fl_Offscreen](#) ctx)
Send all subsequent drawing commands to this offscreen buffer.
- void [fl_begin_points](#) ()
Start drawing a list of points.
- void [fl_begin_polygon](#) ()
Start drawing a convex filled polygon.
- char [fl_can_do_alpha_blending](#) ()
Check whether platform supports true alpha blending for RGBA images.
- [Fl_RGB_Image](#) * [fl_capture_window](#) ([Fl_Window](#) *win, int x, int y, int w, int h)
Captures the content of a rectangular zone of a mapped window.
- void [fl_chord](#) (int x, int y, int w, int h, double a1, double a2)
fl_chord declaration is a place holder - the function does not yet exist
- void [fl_circle](#) (double x, double y, double r)
fl_circle(x,y,r) is equivalent to fl_arc(x,y,r,0,360), but may be faster.
- void [fl_clip](#) (int x, int y, int w, int h)
Intersect the current clip region with a rectangle and push this new region onto the stack (deprecated).
- int [fl_clip_box](#) (int x, int y, int w, int h, int &X, int &Y, int &W, int &H)
Intersect a rectangle with the current clip region and return the bounding box of the result.
- [Fl_Region](#) [fl_clip_region](#) ()
Return the current clipping region.
- void [fl_clip_region](#) ([Fl_Region](#) r)
Replace the top of the clipping stack with a clipping region of any shape.
- void [fl_copy_offscreen](#) (int x, int y, int w, int h, [Fl_Offscreen](#) pixmap, int srcx, int srcy)
Copy a rectangular area of the given offscreen buffer into the current drawing destination.
- [Fl_Offscreen](#) [fl_create_offscreen](#) (int w, int h)
Creation of an offscreen graphics buffer.
- void [fl_cursor](#) ([Fl_Cursor](#))
Sets the cursor for the current window to the specified shape and colors.
- void [fl_cursor](#) ([Fl_Cursor](#), [Fl_Color](#) fg, [Fl_Color](#) bg=[FL_WHITE](#))
- void [fl_curve](#) (double X0, double Y0, double X1, double Y1, double X2, double Y2, double X3, double Y3)
Add a series of points on a Bézier curve to the path.
- void [fl_delete_offscreen](#) ([Fl_Offscreen](#) ctx)
Deletion of an offscreen graphics buffer.
- void [fl_draw](#) (const char *str, int n, int x, int y)
Draws starting at the given x, y location a UTF-8 string of length n bytes.
- void [fl_draw](#) (const char *str, int x, int y)
Draw a nul-terminated UTF-8 string starting at the given x, y location.
- void [fl_draw](#) (const char *str, int x, int y, int w, int h, [Fl_Align](#) align, [Fl_Image](#) *img=0, int draw_symbols=1, int spacing=0)
Fancy string drawing function which is used to draw all the labels.
- void [fl_draw](#) (const char *str, int x, int y, int w, int h, [Fl_Align](#) align, void(*callthis)(const char *, int, int, int), [Fl_Image](#) *img=0, int draw_symbols=1, int spacing=0)
The same as fl_draw(const char,int,int,int,int,Fl_Align,Fl_Image*,int) with the addition of the callthis parameter, which is a pointer to a text drawing function such as fl_draw(const char*, int, int, int) to do the real work.*
- void [fl_draw](#) (int angle, const char *str, int n, int x, int y)

- Draw at the given x , y location a UTF-8 string of length n bytes rotating $angle$ degrees counter-clockwise.*

 - void `fl_draw` (int angle, const char *str, int x, int y)
- Draw a nul-terminated UTF-8 string starting at the given x , y location and rotating $angle$ degrees counter-clockwise.*

 - void `fl_draw_arrow` (FI_Rect bb, FI_Arrow_Type t, FI_Orientation o, FI_Color color)
- Draw an "arrow like" GUI element for the selected scheme.*

 - void `fl_draw_box` (FI_Boxtype, int x, int y, int w, int h, FI_Color)
- Draws a box using given type, position, size and color.*

 - void `fl_draw_box_focus` (FI_Boxtype, int x, int y, int w, int h, FI_Color, FI_Color)
- Draws the focus rectangle inside a box using given type, position, size and color.*

 - void `fl_draw_check` (FI_Rect bb, FI_Color col)
- Draw a check mark inside the given bounding box.*

 - void `fl_draw_circle` (int x, int y, int d, FI_Color color)
- Draw a potentially small, filled circle using a given color.*

 - void `fl_draw_image` (const uchar *buf, int X, int Y, int W, int H, int D=3, int L=0)
- Draw an 8-bit per color RGB or luminance image.*

 - void `fl_draw_image` (FI_Draw_Image_Cb cb, void *data, int X, int Y, int W, int H, int D=3)
- Draw an image using a callback function to generate image data.*

 - void `fl_draw_image_mono` (const uchar *buf, int X, int Y, int W, int H, int D=1, int L=0)
- Draw a gray-scale (1 channel) image.*

 - void `fl_draw_image_mono` (FI_Draw_Image_Cb cb, void *data, int X, int Y, int W, int H, int D=1)
- Draw a gray-scale image using a callback function to generate image data.*

 - int `fl_draw_pixmap` (char *const *data, int x, int y, FI_Color bg=FL_GRAY)
- Draw XPM image data, with the top-left corner at the given position.*

 - int `fl_draw_pixmap` (const char *const *data, int x, int y, FI_Color bg=FL_GRAY)
- Draw XPM image data, with the top-left corner at the given position.*

 - void `fl_draw_radio` (int x, int y, int d, FI_Color color)
- Draw a round check mark (circle) of a radio button.*

 - int `fl_draw_symbol` (const char *label, int x, int y, int w, int h, FI_Color)
- Draw the named symbol in the given rectangle using the given color.*

 - void `fl_end_complex_polygon` ()
- End complex filled polygon, and draw.*

 - void `fl_end_line` ()
- End list of lines, and draw.*

 - void `fl_end_loop` ()
- End closed sequence of lines, and draw.*

 - void `fl_end_offscreen` ()
- Quit sending drawing commands to the current offscreen buffer.*

 - void `fl_end_points` ()
- End list of points, and draw.*

 - void `fl_end_polygon` ()
- End convex filled polygon, and draw.*

 - const char * `fl_expand_text` (const char *from, char *buf, int maxbuf, double maxw, int &n, double &width, int wrap, int draw_symbols=0)
- Copy from to buf, replacing control characters with ^X.*

 - void `fl_focus_rect` (int x, int y, int w, int h)
- Draw a dotted rectangle, used to indicate keyboard focus on a widget.*

 - void `fl_frame` (const char *s, int x, int y, int w, int h)
- Draws a series of line segments around the given box.*

 - void `fl_frame2` (const char *s, int x, int y, int w, int h)
- Draws a series of line segments around the given box.*

 - void `fl_gap` ()

- Separate loops of the path.*

 - void **fl_line** (int x, int y, int x1, int y1)

Draw a line from (x,y) to (x1,y1)
- void **fl_line** (int x, int y, int x1, int y1, int x2, int y2)

Draw a line from (x,y) to (x1,y1) and another from (x1,y1) to (x2,y2)
- void **fl_line_style** (int style, int width=0, char *dashes=0)

Set how to draw lines (the "pen").
- void **fl_load_identity** ()

Set the transformation matrix to identity.
- void **fl_load_matrix** (double a, double b, double c, double d, double x, double y)

Set the current transformation matrix.
- void **fl_loop** (int x, int y, int x1, int y1, int x2, int y2)

Outline a 3-sided polygon with lines.
- void **fl_loop** (int x, int y, int x1, int y1, int x2, int y2, int x3, int y3)

Outline a 4-sided polygon with lines.
- void **fl_measure** (const char *str, int &x, int &y, int draw_symbols=1)

Measure how wide and tall the string will be when printed by the fl_draw() function with align parameter.
- int **fl_measure_pixmap** (char *const *data, int &w, int &h)

Get the dimensions of a pixmap.
- int **fl_measure_pixmap** (const char *const *cdata, int &w, int &h)

Get the dimensions of a pixmap.
- void **fl_mult_matrix** (double a, double b, double c, double d, double x, double y)

Concatenate another transformation onto the current one.
- int **fl_not_clipped** (int x, int y, int w, int h)

Does the rectangle intersect the current clip region?
- unsigned int **fl_old_shortcut** (const char *s)

Emulation of XForms named shortcuts.
- void **fl_overlay_clear** ()

Erase a selection rectangle without drawing a new one.
- void **fl_overlay_rect** (int x, int y, int w, int h)

Draw a transient dotted selection rectangle.
- float **fl_override_scale** ()

Removes any GUI scaling factor in subsequent drawing operations.
- void **fl_pie** (int x, int y, int w, int h, double a1, double a2)

Draw filled ellipse sections using integer coordinates.
- void **fl_point** (int x, int y)

Draw a single pixel at the given coordinates.
- void **fl_polygon** (int x, int y, int x1, int y1, int x2, int y2)

Fill a 3-sided polygon.
- void **fl_polygon** (int x, int y, int x1, int y1, int x2, int y2, int x3, int y3)

Fill a 4-sided polygon.
- void **fl_pop_clip** ()

Restore the previous clip region.
- void **fl_pop_matrix** ()

Restore the current transformation matrix from the stack.
- void **fl_push_clip** (int x, int y, int w, int h)

Intersect the current clip region with a rectangle and push this new region onto the stack.
- void **fl_push_matrix** ()

Save the current transformation matrix on the stack.
- void **fl_push_no_clip** ()

Push an empty clip region onto the stack so nothing will be clipped.

- `uchar * fl_read_image (uchar *p, int X, int Y, int W, int H, int alpha=0)`
Reads an RGB(A) image from the current window or off-screen buffer.
- `void fl_rect (FI_Rect r)`
Draw a border inside the given bounding box.
- `void fl_rect (int x, int y, int w, int h)`
Draw a border inside the given bounding box.
- `void fl_rect (int x, int y, int w, int h, FI_Color c)`
Draw with passed color a border inside the given bounding box.
- `void fl_rectf (FI_Rect bb, uchar r, uchar g, uchar b)`
Color a rectangle with "exactly" the passed r, g, b color.
- `void fl_rectf (FI_Rect r)`
Color with current color a rectangle that exactly fills the given bounding box.
- `void fl_rectf (FI_Rect r, FI_Color c)`
Color with passed color a rectangle that exactly fills the given bounding box.
- `void fl_rectf (int x, int y, int w, int h)`
Color with current color a rectangle that exactly fills the given bounding box.
- `void fl_rectf (int x, int y, int w, int h, FI_Color c)`
Color with passed color a rectangle that exactly fills the given bounding box.
- `void fl_rectf (int x, int y, int w, int h, uchar r, uchar g, uchar b)`
Color a rectangle with "exactly" the passed r, g, b color.
- `void fl_rescale_offscreen (FI_Offscreen &ctx)`
Adapts an offscreen buffer to a changed value of the scale factor.
- `void fl_reset_spot (void)`
Resets marked text.
- `void fl_restore_clip ()`
Undo any clobbering of the clip region done by your program.
- `void fl_restore_scale (float s)`
Restores the GUI scaling factor in subsequent drawing operations.
- `void fl_rotate (double d)`
Concatenate rotation transformation onto the current one.
- `void fl_rounded_rect (int x, int y, int w, int h, int r)`
Draw a rounded border inside the given bounding box.
- `void fl_rounded_rectf (int x, int y, int w, int h, int r)`
Color with current color a rounded rectangle that exactly fills the given bounding box.
- `void fl_rtl_draw (const char *str, int n, int x, int y)`
Draw a UTF-8 string of length n bytes right to left starting at the given x, y location.
- `void fl_scale (double x)`
Concatenate scaling transformation onto the current one.
- `void fl_scale (double x, double y)`
Concatenate scaling transformation onto the current one.
- `void fl_scroll (int X, int Y, int W, int H, int dx, int dy, void(*draw_area)(void *, int, int, int, int), void *data)`
Scroll a rectangle and draw the newly exposed portions.
- `void fl_set_spot (int font, int size, int X, int Y, int W, int H, FI_Window *win=0)`
Inform text input methods about the current text insertion cursor.
- `void fl_set_status (int X, int Y, int W, int H)`
Related to text input methods under X11.
- `const char * fl_shortcut_label (unsigned int shortcut)`
Get a human-readable string from a shortcut value.
- `const char * fl_shortcut_label (unsigned int shortcut, const char **eom)`
Get a human-readable string from a shortcut value.
- `double fl_transform_dx (double x, double y)`

- Transform distance using current transformation matrix.*
- double [fl_transform_dy](#) (double x, double y)
- Transform distance using current transformation matrix.*
- double [fl_transform_x](#) (double x, double y)
- Transform coordinate using the current transformation matrix.*
- double [fl_transform_y](#) (double x, double y)
- Transform coordinate using the current transformation matrix.*
- void [fl_transformed_vertex](#) (double xf, double yf)
- Add coordinate pair to the vertex list without further transformations.*
- void [fl_translate](#) (double x, double y)
- Concatenate translation transformation onto the current one.*
- void [fl_vertex](#) (double x, double y)
- Add a single vertex to the current path.*
- void [fl_xylene](#) (int x, int y, int x1)
- Draw a horizontal line from (x,y) to (x1,y).*
- void [fl_xylene](#) (int x, int y, int x1, int y2)
- Draw a horizontal line from (x,y) to (x1,y), then vertical from (x1,y) to (x1,y2).*
- void [fl_xylene](#) (int x, int y, int x1, int y2, int x3)
- Draw a horizontal line from (x,y) to (x1,y), then a vertical from (x1,y) to (x1,y2) and then another horizontal from (x1,y2) to (x3,y2).*
- void [fl_yxline](#) (int x, int y, int y1)
- Draw a vertical line from (x,y) to (x,y1)*
- void [fl_yxline](#) (int x, int y, int y1, int x2)
- Draw a vertical line from (x,y) to (x,y1), then a horizontal from (x,y1) to (x2,y1).*
- void [fl_yxline](#) (int x, int y, int y1, int x2, int y3)
- Draw a vertical line from (x,y) to (x,y1), then a horizontal from (x,y1) to (x2,y1), then another vertical from (x2,y1) to (x2,y3).*

10.7.1 Detailed Description

FLTK global graphics and GUI drawing functions.

These functions are declared in [<FL/fl_draw.H>](#), and in [<FL/platform.H>](#) for offscreen buffer-related ones.

10.7.2 Enumeration Type Documentation

10.7.2.1 anonymous enum

anonymous enum

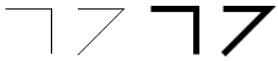
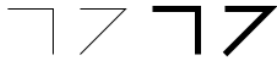
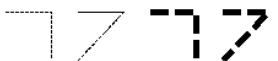
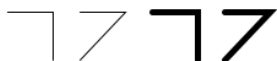
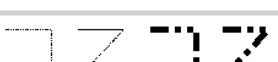
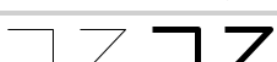
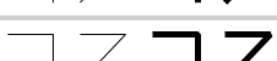
FL_SOLID		FL_CAP_FLAT	
FL_DASH		FL_CAP_ROUND	
FL_DOT		FL_CAP_SQUARE	
FL_DASHDOT		FL_JOIN_MITER	
FL_DASHDOTDOT		FL_JOIN_ROUND	
		FL_JOIN_BEVEL	

Figure 10.2 [fl_line_style\(\)](#) styles

Enumerator

FL_SOLID	line style: solid line
FL_DASH	line style: 75% dashed line
FL_DOT	line style: 50% pixel dotted
FL_DASHDOT	line style: dash / dot pattern
FL_DASHDOTDOT	line style: dash / two dot pattern
FL_CAP_FLAT	cap style: end is flat
FL_CAP_ROUND	cap style: end is round
FL_CAP_SQUARE	cap style: end wraps end point
FL_JOIN_MITER	join style: line join extends to a point
FL_JOIN_ROUND	join style: line join is rounded
FL_JOIN_BEVEL	join style: line join is tidied

10.7.3 Function Documentation

10.7.3.1 fl_add_symbol()

```
int fl_add_symbol (
    const char * name,
    void(* drawit ) (Fl_Color),
    int scalable)
```

Adds a symbol to the system.

Parameters

in	<i>name</i>	name of symbol (without the "@")
in	<i>drawit</i>	function to draw symbol
in	<i>scalable</i>	set to 1 if <i>drawit</i> uses scalable vector drawing

Returns

1 on success, 0 on failure

10.7.3.2 fl_antialias()

```
void fl_antialias (
    int state) [inline]
```

Turn antialiased line drawings ON or OFF, if supported by platform.

Currently, only the Windows platform allows to change whether line drawings are antialiased. Turning it OFF may accelerate heavy drawing operations.

10.7.3.3 fl_arc() [1/2]

```
void fl_arc (
    double x,
    double y,
    double r,
    double start,
    double end) [inline]
```

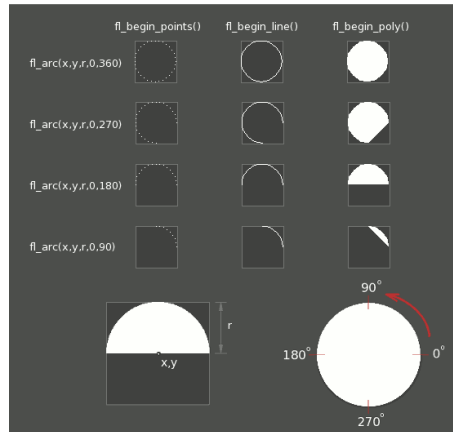
Add a series of points to the current path on the arc of a circle.

The arc is drawn counter-clockwise from 3 o'clock. If *end* is less than *start* then it draws the arc in a clockwise direction. To draw an arc across the 3 o'clock line, *start* and *end* can be greater than 360 or less than 0. For example, to draw a counter-clockwise arc from 6 to 12 o'clock, *start* would be -90 deg, and *end* would be at +90 deg.

You can get elliptical paths by using `scale` and `rotate` before calling `fl_arc()`.

Parameters

in	<i>x,y,r</i>	center and radius of circular arc
in	<i>start,end</i>	angles of start and end of arc measured in degrees

Figure 10.3 `fl_arc(x,y,r,a1,a2)`

Examples:

```
// Draw an arc of points
fl_begin_points();
fl_arc(100.0, 100.0, 50.0, 0.0, 180.0);
fl_end_points();

// Draw arc with a line
fl_begin_line();
fl_arc(200.0, 100.0, 50.0, 0.0, 180.0);
fl_end_line();

// Draw filled arc
fl_begin_polygon();
fl_arc(300.0, 100.0, 50.0, 0.0, 180.0);
fl_end_polygon();
```

10.7.3.4 `fl_arc()` [2/2]

```
void fl_arc (
    int x,
    int y,
    int w,
    int h,
    double a1,
    double a2) [inline]
```

Draw ellipse sections using integer coordinates.

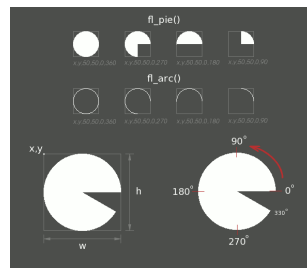
These functions match the rather limited circle drawing code provided by X and Windows. The advantage over using `fl_arc` with floating point coordinates is that they are faster because they often use the hardware, and they draw much nicer small circles, since the small sizes are often hard-coded bitmaps.

If a complete circle is drawn it will fit inside the passed bounding box. The two angles are measured in degrees counter-clockwise from 3 o'clock and are the starting and ending angle of the arc, `a2` must be greater or equal to `a1`.

`fl_arc()` draws a series of lines to approximate the arc. Notice that the integer version of `fl_arc()` has a different number of arguments than the double version `fl_arc(double x, double y, double r, double start, double end)`

Parameters

in	<i>x,y,w,h</i>	bounding box of complete circle
in	<i>a1,a2</i>	start and end angles of arc measured in degrees counter-clockwise from 3 o'clock. <code>a2</code> must be greater than or equal to <code>a1</code> .

Figure 10.4 `fl_pie()` and `fl_arc()`

10.7.3.5 `fl_begin_complex_polygon()`

```
void fl_begin_complex_polygon () [inline]
```

Start drawing a complex filled polygon.

The polygon may be concave, may have holes in it, or may be several disconnected pieces. Call `fl_gap()` to separate loops of the path.

To outline the polygon, use `fl_begin_loop()` and replace each `fl_gap()` with `fl_end_loop();fl_begin_loop()` pairs.

Note

For portability, you should only draw polygons that appear the same whether "even/odd" or "non-zero" winding rules are used to fill them. Holes should be drawn in the opposite direction to the outside loop.

10.7.3.6 `fl_begin_offscreen()`

```
void fl_begin_offscreen (
    Fl_Offscreen ctx)
```

Send all subsequent drawing commands to this offscreen buffer.

Parameters

<code>ctx</code>	the offscreen buffer.

Note

The `ctx` argument must have been created by `fl_create_offscreen()`.

10.7.3.7 `fl_begin_points()`

```
void fl_begin_points () [inline]
```

Start drawing a list of points.

Points are added to the list with `fl_vertex()`.

10.7.3.8 `fl_can_do_alpha_blending()`

```
char fl_can_do_alpha_blending () [inline]
```

Check whether platform supports true alpha blending for RGBA images.

Returns

1 if true alpha blending supported by platform

0 not supported so FLTK will use screen door transparency

10.7.3.9 fl_capture_window()

```
Fl_RGB_Image * fl_capture_window (
    Fl_Window * win,
    int x,
    int y,
    int w,
    int h)
```

Captures the content of a rectangular zone of a mapped window.

Parameters

<i>win</i>	a mapped Fl_Window (derived types including Fl_Gl_Window are also possible)
<i>x,y,w,h</i>	window area to be captured. Intersecting sub-windows are captured too.

Returns

The captured pixels as an [Fl_RGB_Image](#). The raw and drawing sizes of the image can differ. Returns NULL when capture was not successful. The image depth may differ between platforms.

Version

1.4

10.7.3.10 fl_circle()

```
void fl_circle (
    double x,
    double y,
    double r) [inline]
```

`fl_circle(x,y,r)` is equivalent to `fl_arc(x,y,r,0,360)`, but may be faster.

Parameters

in	<i>x,y,r</i>	center and radius of circle

Note

`fl_circle()` is best used as part of the [Drawing Complex Shapes](#) API, that is, flanked by `fl_begin_XXX()` and `fl_end_XXX()` calls where XXX can be 'loop' or 'polygon' to draw, respectively a circle or a disk. Transformation functions (e.g., [fl_scale\(double, double\)](#)) can be also used for `fl_circle()` to draw empty or filled ellipses. It must be the *only* thing in the path: if you want a circle as part of a complex polygon you must use `fl_arc()`. Nevertheless, `fl_circle()` can also be used by itself to draw circles.

10.7.3.11 fl_clip()

```
void fl_clip (
    int x,
    int y,
    int w,
    int h) [inline]
```

Intersect the current clip region with a rectangle and push this new region onto the stack (deprecated).

Parameters

in	<i>x,y,w,h</i>	position and size

Deprecated Please use `fl_push_clip(int x, int y, int w, int h)` instead. `fl_clip(int, int, int, int)` will be removed in FLTK 1.5.

10.7.3.12 fl_clip_box()

```
int fl_clip_box (
    int x,
    int y,
    int w,
    int h,
    int & X,
    int & Y,
    int & W,
    int & H) [inline]
```

Intersect a rectangle with the current clip region and return the bounding box of the result.

Returns non-zero if the resulting rectangle is different to the original. The given rectangle (*x*, *y*, *w*, *h*) *should* be entirely inside its window, otherwise the result may be unexpected, i.e. this function *may* not clip the rectangle to the window coordinates and size. In particular *x* and *y* *should* not be negative.

The resulting bounding box can be used to limit the necessary drawing to this rectangle.

Example:

```
void MyGroup::draw() {
    int X = 0, Y = 0, W = 0, H = 0;
    int ret = fl_clip_box(x(), y(), w(), h(), X, Y, W, H);
    if (ret == 0) { // entire group is visible (not clipped)
        // full drawing code here
    } else { // parts of this group are clipped
        // partial drawing code here (uses X, Y, W, and H to test)
    }
}
```

W and *H* are set to zero if the rectangle is completely outside the clipping region. In this case *X* and *Y* are undefined and should not be used. Possible values are (0, 0), (*x*, *y*), or anything else (platform dependent).

Note

This function is platform-dependent. If the given rectangle is not entirely inside the window, the results are not guaranteed to be the same on all platforms.

Parameters

in	<i>x,y,w,h</i>	position and size of rectangle
out	<i>X,Y,W,H</i>	position and size of resulting bounding box.

Returns

Non-zero if the resulting rectangle is different to the original.

See also

[fl_not_clipped\(\)](#)

10.7.3.13 fl_clip_region() [1/2]

```
Fl_Region fl_clip_region () [inline]
```

Return the current clipping region.

Note

This function is mostly intended for internal use by the FLTK library when drawing to the display. Its return value can be always NULL if the current drawing surface is not the display.

10.7.3.14 fl_clip_region() [2/2]

```
void fl_clip_region (
    Fl_Region r) [inline]
```

Replace the top of the clipping stack with a clipping region of any shape.

Fl_Region is an operating system specific type.

Note

This function is mostly intended for internal use by the FLTK library when drawing to the display. Its effect can be null if the current drawing surface is not the display.

Parameters

<code>in</code>	<code>r</code>	clipping region

10.7.3.15 fl_copy_offscreen()

```
void fl_copy_offscreen (
    int x,
    int y,
    int w,
    int h,
    Fl_Offscreen pixmap,
    int srcx,
    int srcy) [inline]
```

Copy a rectangular area of the given offscreen buffer into the current drawing destination.

Parameters

<code>x,y</code>	position where to draw the copied rectangle
<code>w,h</code>	size of the copied rectangle
<code>pixmap</code>	offscreen buffer containing the rectangle to copy
<code>srcx,srcy</code>	origin in offscreen buffer of rectangle to copy

10.7.3.16 fl_create_offscreen()

```
Fl_Offscreen fl_create_offscreen (
    int w,
    int h)
```

Creation of an offscreen graphics buffer.

Parameters

<code>w,h</code>	width and height in FLTK units of the buffer.

Returns

the created graphics buffer.

The pixel size of the created graphics buffer is equal to the number of pixels in an area of the screen containing the current window sized at `w,h` FLTK units. This pixel size varies with the value of the scale factor of this screen.

Note

Work with the `fl_XXX_offscreen()` functions is equivalent to work with an [Fl_Image_Surface](#) object, as follows :

Fl_Offscreen-based approach	Fl_Image_Surface-based approach
<code>Fl_Offscreen off = fl_create_offscreen(w, h)</code>	<code>Fl_Image_Surface *surface = new Fl_Image_Surface(w, h, 1)</code>
<code>fl_begin_offscreen(off)</code>	<code>Fl_Surface_Device::push_current(surface)</code>
<code>fl_end_offscreen()</code>	<code>Fl_Surface_Device::pop_current()</code>
<code>fl_copy_offscreen(x,y,w,h, off, sx,sy)</code>	<code>fl_copy_offscreen(x,y,w,h, surface->offscreen(), sx,sy)</code>
<code>fl_rescale_offscreen(off)</code>	<code>surface->rescale()</code>
<code>fl_delete_offscreen(off)</code>	delete surface

10.7.3.17 fl_cursor()

```
void fl_cursor (
    Fl\_Cursor c)
```

Sets the cursor for the current window to the specified shape and colors. The cursors are defined in the [<FL/Enumerations.H>](#) header file.

10.7.3.18 fl_curve()

```
void fl_curve (
    double X0,
    double Y0,
    double X1,
    double Y1,
    double X2,
    double Y2,
    double X3,
    double Y3) [inline]
```

Add a series of points on a Bézier curve to the path. The curve ends (and two of the points) are at X0,Y0 and X3,Y3.

Parameters

in	X0,Y0	curve start point
in	X1,Y1	curve control point
in	X2,Y2	curve control point
in	X3,Y3	curve end point

10.7.3.19 fl_delete_offscreen()

```
void fl_delete_offscreen (
    Fl\_Offscreen ctx)
```

Deletion of an offscreen graphics buffer.

Parameters

ctx	the buffer to be deleted.

Note

The `ctx` argument must have been created by [fl_create_offscreen\(\)](#).

10.7.3.20 fl_draw() [1/5]

```
void fl_draw (
    const char * str,
    int x,
    int y)
```

Draw a nul-terminated UTF-8 string starting at the given `x`, `y` location.

Text is aligned to the left and to the baseline of the font. To align to the bottom, subtract [fl_descent\(\)](#) from `y`. To align to the top, subtract [fl_descent\(\)](#) and add [fl_height\(\)](#). This version of `fl_draw` provides direct access to the text drawing function of the underlying OS. It does not apply any special handling to control characters.

10.7.3.21 fl_draw() [2/5]

```
void fl_draw (
    const char * str,
    int x,
    int y,
    int w,
    int h,
    Fl_Align align,
    Fl_Image * img = 0,
    int draw_symbols = 1,
    int spacing = 0)
```

Fancy string drawing function which is used to draw all the labels.

The string is formatted and aligned inside the passed box. Handles '\t' and '\n', expands all other control characters to '^X', and aligns inside or against the edges of the box. See [Fl_Widget::align\(\)](#) for values of `align`. The value `FL_ALIGN_INSIDE` is ignored, as this function always prints inside the box. If `img` is provided and is not `NULL`, the image is drawn above or below the text as specified by the `align` value. The `draw_symbols` argument specifies whether or not to look for symbol names starting with the '@' character

Parameters

in	<i>str</i>	UTF-8 string, can start and end with an '@sym' symbol, can contain '\n'
in	<i>x,y,w,h</i>	bounding box
in	<i>align</i>	label and image alignment in bounding box
in	<i>img</i>	pointer to image
in	<i>draw_symbols</i>	if true, interpret leading and trailing '@sym' as graphical symbols
in	<i>spacing</i>	spacing between text and image

10.7.3.22 fl_draw() [3/5]

```
void fl_draw (
    const char * str,
    int x,
    int y,
    int w,
    int h,
    Fl_Align align,
    void(* callthis )(const char *, int, int, int),
    Fl_Image * img = 0,
    int draw_symbols = 1,
    int spacing = 0)
```

The same as `fl_draw(const char*,int,int,int,int,Fl_Align,Fl_Image*,int)` with the addition of the `callthis` parameter, which is a pointer to a text drawing function such as `fl_draw(const char*, int, int, int)` to do the real work.

Parameters

in	<i>str</i>	UTF-8 string, can start and end with an '@sym' symbol, can contain '\n'
in	<i>x,y,w,h</i>	bounding box
in	<i>align</i>	label and image alignment in bounding box
in	<i>callthis</i>	pointer to text drawing function
in	<i>img</i>	pointer to image
in	<i>draw_symbols</i>	if true, interpret leading and trailing '@sym' as graphical symbols
in	<i>spacing</i>	spacing between text and image

10.7.3.23 fl_draw() [4/5]

```
void fl_draw (
    int angle,
    const char * str,
    int n,
    int x,
    int y) [inline]
```

Draw at the given *x*, *y* location a UTF-8 string of length *n* bytes rotating *angle* degrees counter-clockwise.

Note

When using X11 (Unix, Linux, Cygwin et al.) this needs Xft to work. Under plain X11 (w/o Xft) rotated text is not supported by FLTK. A warning will be issued to *stderr* at runtime (only once) if you use this method with an angle other than 0.

10.7.3.24 fl_draw() [5/5]

```
void fl_draw (
    int angle,
    const char * str,
    int x,
    int y)
```

Draw a nul-terminated UTF-8 string starting at the given *x*, *y* location and rotating *angle* degrees counter-clockwise.

This version of *fl_draw* provides direct access to the text drawing function of the underlying OS and is supported by all fltk platforms except X11 without Xft.

10.7.3.25 fl_draw_arrow()

```
void fl_draw_arrow (
    Fl_Rect r,
    Fl_Arrow_Type t,
    Fl_Orientation o,
    Fl_Color col)
```

Draw an "arrow like" GUI element for the selected scheme.

In the future this function should be integrated in [Fl_Scheme](#) as a virtual method, i.e. it would call a method like ...
`Fl_Scheme::current()->draw_arrow(r, t, o, col);`

Parameters

in	<i>r</i>	bounding box
in	<i>t</i>	arrow type
in	<i>o</i>	orientation
in	<i>col</i>	arrow color

Since

1.4.0

10.7.3.26 fl_draw_box()

```
void fl_draw_box (
    Fl_Boxtype t,
    int x,
    int y,
    int w,
    int h,
    Fl_Color c)
```

Draws a box using given type, position, size and color.

Parameters

in	<i>t</i>	box type
in	<i>x,y,w,h</i>	position and size
in	<i>c</i>	color

10.7.3.27 fl_draw_box_focus()

```
void fl_draw_box_focus (
    Fl_Boxtype bt,
    int x,
    int y,
    int w,
    int h,
    Fl_Color fg,
    Fl_Color bg)
```

Draws the focus rectangle inside a box using given type, position, size and color.

Boxes can set their own focus drawing callback. The focus frame does not need to be a rectangle at all, but should fit inside the shape of the box.

Parameters

in	<i>bt</i>	box type
in	<i>x,y,w,h</i>	position and size
in	<i>fg,bg</i>	foreground and background color

10.7.3.28 fl_draw_check()

```
void fl_draw_check (
    Fl_Rect bb,
    Fl_Color col)
```

Draw a check mark inside the given bounding box.

The check mark is allowed to fill the entire box but the algorithm used makes sure that a 1-pixel border is kept free if the box is large enough. You need to calculate margins for box borders etc. yourself.

The check mark size is limited (minimum and maximum size) and the check mark is always centered in the given box.

Note

If the box is too small (bad GUI design) the check mark will be drawn over the box borders. This is intentional for better user experience. Otherwise users might not be able to recognize if a box is checked.

The size limits are implementation details and may be changed at any time.

Parameters

in	<i>bb</i>	rectangle that defines the bounding box
in	<i>col</i>	Fl_Color to draw the check mark

Since

1.4.0

10.7.3.29 fl_draw_circle()

```
void fl_draw_circle (
    int x,
    int y,
    int d,
    Fl_Color color)
```

Draw a potentially small, filled circle using a given color.

This function draws a filled circle bounded by rectangle (x, y, d, d) using color `color`

This function is the same as `fl_pie(x, y, d, d, 0, 360)` except with some systems that don't draw small circles well. In that situation, the circle diameter `d` is converted from FLTK units to pixels and this function approximates a filled circle by drawing several filled rectangles if the converted diameter is 6 pixels.

The current drawing color `fl_color()` is preserved across the call.

Parameters

in	<code>x,y</code>	coordinates of top left of the bounding box
in	<code>d</code>	diameter == width and height of the bounding box in FLTK units
in	<code>color</code>	the color used to draw the circle

Since

1.4.0

10.7.3.30 fl_draw_image() [1/2]

```
void fl_draw_image (
    const uchar * buf,
    int X,
    int Y,
    int W,
    int H,
    int D = 3,
    int L = 0) [inline]
```

Draw an 8-bit per color RGB or luminance image.

Parameters

in	<code>buf</code>	points at the "r" data of the top-left pixel. Color data must be in <code>r, g, b</code> order. Luminance data is only one <code>gray</code> byte.
in	<code>X,Y</code>	position where to put top-left corner of image
in	<code>W,H</code>	size of the image
in	<code>D</code>	delta to add to the pointer between pixels. It may be any value greater than or equal to 1, or it can be negative to flip the image horizontally
in	<code>L</code>	delta to add to the pointer between lines (if 0 is passed it uses $W * D$), and may be larger than $W * D$ to crop data, or negative to flip the image vertically

It is highly recommended that you put the following code before the first `show()` of *any* window in your program to get rid of the dithering if possible:

```
Fl::visual(FL_RGB);
```

Gray scale (1-channel) images may be drawn. This is done if `abs(D)` is less than 3, or by calling `fl_draw_image_mono()`. Only one 8-bit sample is used for each pixel, and on screens with different numbers of bits for red, green, and blue only gray colors are used. Setting `D` greater than 1 will let you display one channel of a color image.

Note:

The X version does not support all possible visuals. If FLTK cannot draw the image in the current visual it will abort. FLTK supports any visual of 8 bits or less, and all common TrueColor visuals up to 32 bits.

10.7.3.31 fl_draw_image() [2/2]

```
void fl_draw_image (
    Fl_Draw_Image_Cb cb,
    void * data,
    int X,
    int Y,
    int W,
    int H,
    int D = 3) [inline]
```

Draw an image using a callback function to generate image data.

You can generate the image as it is being drawn, or do arbitrary decompression of stored data, provided it can be decompressed to individual scan lines.

Parameters

in	<i>cb</i>	callback function to generate scan line data
in	<i>data</i>	user data passed to callback function
in	<i>X,Y</i>	screen position of top left pixel
in	<i>W,H</i>	image width and height
in	<i>D</i>	data size per pixel in bytes (must be greater than 0)

See also

[fl_draw_image\(const uchar* buf, int X, int Y, int W, int H, int D, int L\)](#)

The callback function *cb* is called with the `void* data` user data pointer to allow access to a structure of information about the image, and the *x*, *y*, and *w* of the scan line desired from the image. 0,0 is the upper-left corner of the image, not *x*, *y*. A pointer to a buffer to put the data into is passed. You must copy *w* pixels from scanline *y*, starting at pixel *x*, to this buffer.

Due to cropping, less than the whole image may be requested. So *x* may be greater than zero, the first *y* may be greater than zero, and *w* may be less than *W*. The buffer is long enough to store the entire *W* * *D* pixels, this is for convenience with some decompression schemes where you must decompress the entire line at once: decompress it into the buffer, and then if *x* is not zero, copy the data over so the *x*'th pixel is at the start of the buffer.

You can assume the *y*'s will be consecutive, except the first one may be greater than zero.

If *D* is 4 or more, you must fill in the unused bytes with zero.

10.7.3.32 fl_draw_image_mono() [1/2]

```
void fl_draw_image_mono (
    const uchar * buf,
    int X,
    int Y,
    int W,
    int H,
    int D = 1,
    int L = 0) [inline]
```

Draw a gray-scale (1 channel) image.

See also

[fl_draw_image\(const uchar* buf, int X,int Y,int W,int H, int D, int L\)](#)

10.7.3.33 fl_draw_image_mono() [2/2]

```
void fl_draw_image_mono (
    Fl_Draw_Image_Cb cb,
    void * data,
```

```

    int X,
    int Y,
    int W,
    int H,
    int D = 1) [inline]

```

Draw a gray-scale image using a callback function to generate image data.

See also

[fl_draw_image\(Fl_Draw_Image_Cb cb, void* data, int X,int Y,int W,int H, int D\)](#)

10.7.3.34 fl_draw_pixmap() [1/2]

```

int fl_draw_pixmap (
    char *const * data,
    int x,
    int y,
    Fl_Color bg = FL_GRAY) [inline]

```

Draw XPM image data, with the top-left corner at the given position.

See also

[fl_draw_pixmap\(const char* const* data, int x, int y, Fl_Color bg\)](#)

10.7.3.35 fl_draw_pixmap() [2/2]

```

int fl_draw_pixmap (
    const char *const * data,
    int x,
    int y,
    Fl_Color bg = FL_GRAY)

```

Draw XPM image data, with the top-left corner at the given position.

The image is dithered on 8-bit displays so you won't lose color space for programs displaying both images and pixmaps.

Parameters

in	<i>data</i>	pointer to XPM image data
in	<i>x,y</i>	position of top-left corner
in	<i>bg</i>	background color

Returns

0 if there was any error decoding the XPM data.

10.7.3.36 fl_draw_radio()

```

void fl_draw_radio (
    int x,
    int y,
    int d,
    Fl_Color color)

```

Draw a round check mark (circle) of a radio button.

This draws only the round "radio button mark", it does not draw the (also typically round) box of the radio button.

Call this only if the radio button is ON.

This method draws a scheme specific "circle" with a particular light effect if the scheme is gtk+. For all other schemes this function draws a simple, small circle.

The `color` must be chosen by the caller so it has enough contrast with the background.

The bounding box of the circle is the rectangle (`x`, `y`, `d`, `d`).

The current drawing color [fl_color\(\)](#) is preserved across the call.

Parameters

in	<i>x,y</i>	coordinates of top left of the bounding box
in	<i>d</i>	diameter == width and height of the bounding box in FLTK units
in	<i>color</i>	the base color used to draw the circle

Since

1.4.0

10.7.3.37 fl_draw_symbol()

```
int fl_draw_symbol (
    const char * label,
    int x,
    int y,
    int w,
    int h,
    Fl_Color col)
```

Draw the named symbol in the given rectangle using the given color.

Parameters

in	<i>label</i>	name of symbol
in	<i>x,y</i>	position of symbol
in	<i>w,h</i>	size of symbol
in	<i>col</i>	color of symbox

Returns

1 on success, 0 on failure

10.7.3.38 fl_expand_text()

```
const char * fl_expand_text (
    const char * from,
    char * buf,
    int maxbuf,
    double maxw,
    int & n,
    double & width,
    int wrap,
    int draw_symbols)
```

Copy *from* to *buf*, replacing control characters with ^X.

Stop at a newline or if *maxbuf* characters written to buffer. Also word-wrap if width exceeds *maxw*. Returns a pointer to the start of the next line of characters. Sets *n* to the number of characters put into the buffer. Sets width to the width of the string in the [current font](#).

Parameters

in	<i>from</i>	input text, can contain ' ' and single or double '@' characters
out	<i>buf</i>	buffer for the output text segment
in	<i>maxbuf</i>	size of the buffer, or 0 to use the internal buffer allocated in this function
in	<i>maxw</i>	maximum width in pixels of the output text

Parameters

out	<i>n</i>	number of characters in the output text segment
out	<i>width</i>	actual width in pixels of the output text
in	<i>wrap</i>	if true, wrap at maxw, else wrap at newlines
in	<i>draw_symbols</i>	if true, double '@' characters are escaped into a single '@'

Returns

pointer to the next character in the input text

10.7.3.39 fl_focus_rect()

```
void fl_focus_rect (
    int x,
    int y,
    int w,
    int h) [inline]
```

Draw a dotted rectangle, used to indicate keyboard focus on a widget.

This method draws the rectangle in the current color and independent of the [Fl::visible_focus\(\)](#) option. You may need to set the current color with [fl_color\(\)](#) before you call this.

10.7.3.40 fl_frame()

```
void fl_frame (
    const char * s,
    int x,
    int y,
    int w,
    int h)
```

Draws a series of line segments around the given box.

The string *s* must contain groups of 4 letters which specify one of 24 standard grayscale values, where 'A' is black and 'X' is white. The order of each set of 4 characters is: top, left, bottom, right. The result of calling [fl_frame\(\)](#) with a string that is not a multiple of 4 characters in length is undefined. The only difference between this function and [fl_frame2\(\)](#) is the order of the line segments.

Parameters

in	<i>s</i>	sets of 4 grayscale values in top, left, bottom, right order
in	<i>x,y,w,h</i>	position and size

10.7.3.41 fl_frame2()

```
void fl_frame2 (
    const char * s,
    int x,
    int y,
    int w,
    int h)
```

Draws a series of line segments around the given box.

The string *s* must contain groups of 4 letters which specify one of 24 standard grayscale values, where 'A' is black and 'X' is white. The order of each set of 4 characters is: bottom, right, top, left. The result of calling [fl_frame2\(\)](#) with a string that is not a multiple of 4 characters in length is undefined. The only difference between this function and [fl_frame\(\)](#) is the order of the line segments.

Parameters

in	<i>s</i>	sets of 4 grayscale values in bottom, right, top, left order
in	<i>x,y,w,h</i>	position and size

10.7.3.42 fl_gap()

```
void fl_gap () [inline]
```

Separate loops of the path.

It is unnecessary but harmless to call `fl_gap()` before the first vertex, after the last vertex, or several times in a row.

10.7.3.43 fl_line_style()

```
void fl_line_style (
    int style,
    int width = 0,
    char * dashes = 0) [inline]
```

Set how to draw lines (the "pen").

If you change this it is your responsibility to set it back to the default using `fl_line_style(0)`.

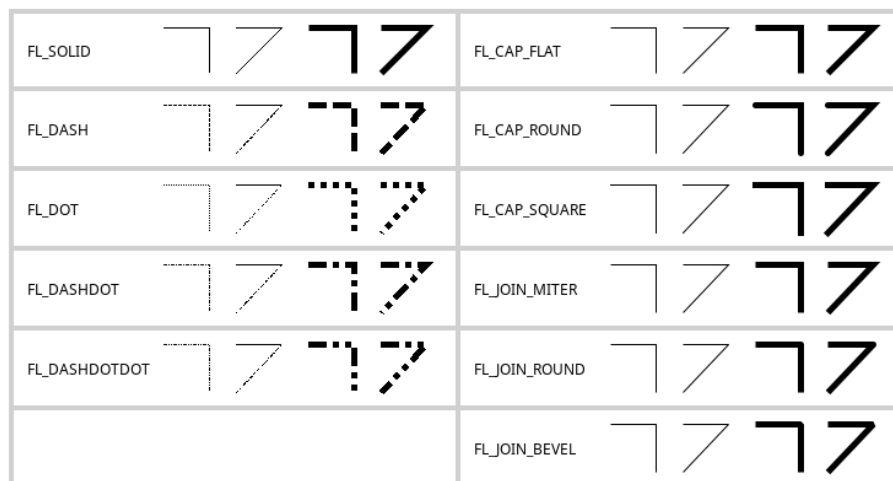


Figure 10.5 fl_line_style() styles

Parameters

in	<i>style</i>	A bitmask which is a bitwise-OR of Line Styles , a cap style, and a join style. If you don't specify a dash type you will get a solid line. If you don't specify a cap or join type you will get a system-defined default of whatever value is fastest.
in	<i>width</i>	The thickness of the lines in pixels. Zero results in the system defined default, which on both X and Windows is somewhat different and nicer than 1.
in	<i>dashes</i>	A pointer to an array of dash lengths, measured in pixels. The first location is how long to draw a solid portion, the next is how long to draw the gap, then the solid, etc. It is terminated with a zero-length entry. A <code>NULL</code> pointer or a zero-length array results in a solid line. Odd array sizes are not supported and result in undefined behavior.

Note

Because of how line styles are implemented on Win32 systems, you *must* set the line style *after* setting the drawing color. If you set the color after the line style you will lose the line style settings.

The `dashes` array does not work under the (unsupported!) operating systems Windows 95, 98 or Me, since those operating systems do not support complex line styles.

10.7.3.44 fl_load_matrix()

```
void fl_load_matrix (
    double a,
    double b,
    double c,
    double d,
    double x,
    double y) [inline]
```

Set the current transformation matrix.

Parameters

in	<i>a,b,c,d,x,y</i>	transformation matrix elements

10.7.3.45 fl_measure()

```
void fl_measure (
    const char * str,
    int & w,
    int & h,
    int draw_symbols)
```

Measure how wide and tall the string will be when printed by the `fl_draw()` function with `align` parameter.

If the incoming `w` is non-zero it will wrap to that width.

The [current font](#) is used to do the width/height calculations, so unless its value is known at the time `fl_measure()` is called, it is advised to first set the current font with `fl_font()`. With event-driven GUI programming you can never be sure which widget was exposed and redrawn last, nor which font it used. If you have not called `fl_font()` explicitly in your own code, the width and height may be set to unexpected values, even zero!

Note: In the general use case, it's a common error to forget to set `w` to 0 before calling `fl_measure()` when wrap behavior isn't needed.

Parameters

in	<i>str</i>	nul-terminated string
in, out	<i>w</i>	call with <code>w=0</code> , or with the preferred width for word wrapping, returns with the width of the string in current font
out	<i>h</i>	height of string in current font
in	<i>draw_symbols</i>	non-zero to enable @symbol handling [default=1]

```
// Example: Common use case for fl_measure()
const char *s = "This is a test";
int wi=0, hi=0;           // initialize to zero before calling fl_measure()
fl_font(FL_HELVETICA, 14); // set current font face/size to be used for measuring
fl_measure(s, wi, hi);     // returns pixel width/height of string in current font
```

10.7.3.46 fl_measure_pixmap() [1/2]

```
int fl_measure_pixmap (
    char *const * data,
    int & w,
    int & h)
```

Get the dimensions of a pixmap.

An XPM image contains the dimensions in its data. This function returns the width and height.

Parameters

in	<i>data</i>	pointer to XPM image data.
out	<i>w,h</i>	width and height of image

Returns

non-zero if the dimensions were parsed OK
 0 if there were any problems

10.7.3.47 fl_measure_pixmap() [2/2]

```
int fl_measure_pixmap (
    const char *const * cdata,
    int & w,
    int & h)
```

Get the dimensions of a pixmap.

See also

[fl_measure_pixmap\(char* const* data, int &w, int &h\)](#)

10.7.3.48 fl_mult_matrix()

```
void fl_mult_matrix (
    double a,
    double b,
    double c,
    double d,
    double x,
    double y) [inline]
```

Concatenate another transformation onto the current one.

Parameters

in	a, b, c, d, x, y	transformation matrix elements such that $X' = aX + cY + x$ and $Y' = bX + dY + y$

10.7.3.49 fl_not_clipped()

```
int fl_not_clipped (
    int x,
    int y,
    int w,
    int h) [inline]
```

Does the rectangle intersect the current clip region?

Parameters

in	x, y, w, h	position and size of rectangle

Returns

non-zero if any of the rectangle intersects the current clip region. If this returns 0 you don't have to draw the object.

Note

Under X this returns 2 if the rectangle is partially clipped and 1 if it is entirely inside the clip region.

See also

[fl_clip_box\(\)](#)

10.7.3.50 fl_old_shortcut()

```
unsigned int fl_old_shortcut (
    const char * s)
```

Emulation of XForms named shortcuts.

Converts ASCII shortcut specifications (eg. "^c") into the FLTK integer equivalent (eg. FL_CTRL+'c')

These ASCII characters are used to specify the various keyboard modifier keys:

```
# - Alt
+ - Shift
^ - Control
! - Meta
@ - Command (Ctrl on linux/win, Meta on OSX)
```

These special characters can be combined to form chords of modifier keys. (See 'Remarks' below)

After the optional modifier key prefixes listed above, one can either specify a single keyboard character to use as the shortcut, or a numeric sequence in hex, decimal or octal.

Examples:

```
"c"      -- Uses 'c' as the shortcut
"#^c"    -- Same as FL_ALT|FL_CTRL|'c'
"#^!c"   -- Same as FL_ALT|FL_CTRL|FL_META|'c'
"@c"     -- Same as FL_COMMAND|'c' (see FL_COMMAND for platform specific behavior)
"0x63"   -- Same as "c" (hex 63=='c')
"99"     -- Same as "c" (dec 99=='c')
"0143"   -- Same as "c" (octal 0143=='c')
"^0x63"  -- Same as (FL_CTRL|'c'), or (FL_CTRL|0x63)
"^99"    -- Same as (FL_CTRL|'c'), or (FL_CTRL|99)
"^0143"  -- Same as (FL_CTRL|'c'), or (FL_CTRL|0143)
```

Remarks

Due to XForms legacy, there are some odd things to consider when using the modifier characters.

(1) You can use the special modifier keys for chords *only* if the modifiers are provided in this order: #, +, ^, !, @. Other ordering can yield undefined results.

So for instance, Ctrl-Alt-c must be specified as "#^c" (and not "^#c"), due to the above ordering rule.

(2) If you want to make a shortcut that uses one of the special modifier characters (as the character being modified), then to avoid confusion, specify the numeric equivalent, e.g.

If you want..	Then use..
'#' as the shortcut..	"0x23" (instead of just "#").
'+' as the shortcut..	"0x2b" (instead of just "+").
'^' as the shortcut..	"0x5e" (instead of just "^").
Alt-+ as the shortcut..	"#0x2b" (instead of "#+").
Alt-^ as the shortcut..	"#0x5e" (instead of "#^").
..etc..	

As a general rule that's easy to remember, unless the shortcut key to be modified is a single alpha-numeric character [A-Z,a-z,0-9], it's probably best to use the numeric equivalents.

Don't fix these silly legacy issues in a future release. Nobody is using this anymore.

10.7.3.51 fl_overlay_clear()

```
void fl_overlay_clear ()
```

Erase a selection rectangle without drawing a new one.

See also

[fl_overlay_rect\(int x, int y, int w, int h\)](#)

10.7.3.52 fl_overlay_rect()

```
void fl_overlay_rect (
    int x,
    int y,
    int w,
    int h)
```

Draw a transient dotted selection rectangle.

This function saves the current screen content and then draws a dotted selection rectangle into the front screen buffer. If another selection rectangle was drawn earlier, the previous screen graphics are restored first.

To clear the selection rectangle, call [fl_overlay_clear\(\)](#).

The typical (and only) use for this function is to draw a selection rectangle during a mouse drag event sequence without having to redraw the entire content of the widget.

Your event handle should look similar to this (also see `test/mandelbrot.cxx`):

```
int MyWidget::handle(int event) {
    static int ix, iy;
    switch (event) {
        case FL_PUSH:
            ix = Fl::event_x(); iy = Fl::event_y();
            return 1;
        case FL_DRAG:
            this->make_current();
            fl_overlay_rect(ix, iy, ix-Fl::event_x(), iy-Fl::event_y());
            return 1;
        case FL_RELEASE:
            this->make_current();
            fl_overlay_clear();
            // select the element under the rectangle
            return 1;
    }
    return MySuperWidget::handle(event);
}
```

Note

Between drawing an overlay rect and clearing it, the content of the widget must not change.

[fl_overlay_rect\(\)](#) and [fl_overlay_clear\(\)](#) should be called when the actual event occurs, and *not* within `MyWidget::draw()`.

[fl_overlay_rect\(\)](#) and [fl_overlay_clear\(\)](#) should not be mixed with [Fl_Overlay_Window](#). [Fl_Overlay_Window](#) provides an entirely different way of drawing selection outlines and is not limited to rectangles.

Parameters

<i>x,y,w,h</i>	position and size of the overlay rectangle.

See also

[fl_overlay_clear\(\)](#)

10.7.3.53 fl_override_scale()

```
float fl_override_scale ()
```

Removes any GUI scaling factor in subsequent drawing operations.

This must be matched by a later call to [fl_restore_scale\(\)](#). This function can be used to transiently perform drawing operations that are not rescaled by the current value of the GUI scaling factor. It's not supported to call [fl_push_clip\(\)](#) while transiently removing scaling.

Returns

The GUI scaling factor value that was in place when the function started.

10.7.3.54 fl_pie()

```
void fl_pie (
    int x,
    int y,
    int w,
    int h,
    double a1,
    double a2) [inline]
```

Draw filled ellipse sections using integer coordinates.

Like [fl_arc\(\)](#), but [fl_pie\(\)](#) draws a filled-in pie slice. This slice may extend outside the line drawn by [fl_arc\(\)](#); to avoid this use $w - 1$ and $h - 1$.

Parameters

in	x, y, w, h	bounding box of complete circle
in	$a1, a2$	start and end angles of arc measured in degrees counter-clockwise from 3 o'clock. $a2$ must be greater than or equal to $a1$.

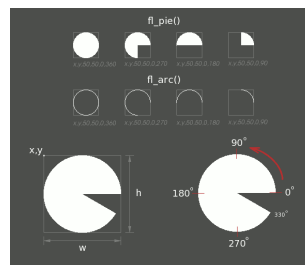


Figure 10.6 [fl_pie\(\)](#) and [fl_arc\(\)](#)

10.7.3.55 fl_polygon()

```
void fl_polygon (
    int x,
    int y,
    int x1,
    int y1,
    int x2,
    int y2,
    int x3,
    int y3) [inline]
```

Fill a 4-sided polygon.

The polygon must be convex.

10.7.3.56 fl_pop_clip()

```
void fl_pop_clip () [inline]
```

Restore the previous clip region.

You must call [fl_pop_clip\(\)](#) once for every time you call [fl_push_clip\(\)](#). Unpredictable results may occur if the clip stack is not empty when you return to FLTK.

10.7.3.57 fl_push_clip()

```
void fl_push_clip (
    int x,
    int y,
```

```
int w,
int h) [inline]
```

Intersect the current clip region with a rectangle and push this new region onto the stack.

Parameters

in	<i>x,y,w,h</i>	position and size

10.7.3.58 fl_push_matrix()

```
void fl_push_matrix () [inline]
```

Save the current transformation matrix on the stack.

The maximum depth of the stack is 32.

10.7.3.59 fl_read_image()

```
uchar * fl_read_image (
    uchar * p,
    int X,
    int Y,
    int w,
    int h,
    int alpha)
```

Reads an RGB(A) image from the current window or off-screen buffer.

Parameters

in	<i>p</i>	pixel buffer, or NULL to allocate one
in	<i>X,Y</i>	position of top-left of image to read
in	<i>w,h</i>	width and height of image to read
in	<i>alpha</i>	alpha value for image (0 for none)

Returns

pointer to pixel buffer, or NULL if allocation failed.

The *p* argument points to a buffer that can hold the image and must be at least *w***h**3 bytes when reading RGB images, or *w***h**4 bytes when reading RGBA images. If NULL, [fl_read_image\(\)](#) will create an array of the proper size which can be freed using `delete[]`.

The *alpha* parameter controls whether an alpha channel is created and the value that is placed in the alpha channel. If 0, no alpha channel is generated.

See also

[fl_capture_window\(\)](#)

10.7.3.60 fl_rect() [1/3]

```
void fl_rect (
    Fl_Rect r) [inline]
```

Draw a border *inside* the given bounding box.

This is the same as [fl_rect\(int x, int y, int w, int h\)](#) but with [Fl_Rect](#) *r* as input argument.

10.7.3.61 fl_rect() [2/3]

```
void fl_rect (
    int x,
    int y,
    int w,
    int h) [inline]
```

Draw a border *inside* the given bounding box.

This function is meant for quick drawing of simple boxes. The behavior is undefined for line widths that are not 1.

10.7.3.62 fl_rect() [3/3]

```
void fl_rect (
    int x,
    int y,
    int w,
    int h,
    Fl_Color c) [inline]
```

Draw with passed color a border *inside* the given bounding box.

Warning

The current color is changed to `c` upon return.

10.7.3.63 fl_rectf() [1/4]

```
void fl_rectf (
    Fl_Rect bb,
    uchar r,
    uchar g,
    uchar b) [inline]
```

Color a rectangle with "exactly" the passed `r`, `g`, `b` color.

This is the same as `fl_rectf(int x, int y, int w, int h, uchar r, uchar g, uchar b)` but with `Fl_Rect bb` (bounding box) as argument instead of `(x, y, w, h)`.

See also

`fl_rectf(int x, int y, int w, int h, uchar r, uchar g, uchar b)`

10.7.3.64 fl_rectf() [2/4]

```
void fl_rectf (
    Fl_Rect r,
    Fl_Color c) [inline]
```

Color with passed color a rectangle that exactly fills the given bounding box.

Warning

The current color is changed to `c` upon return.

10.7.3.65 fl_rectf() [3/4]

```
void fl_rectf (
    int x,
    int y,
    int w,
    int h,
    Fl_Color c) [inline]
```

Color with passed color a rectangle that exactly fills the given bounding box.

Warning

The current color is changed to `c` upon return.

10.7.3.66 fl_rectf() [4/4]

```
void fl_rectf (
    int x,
    int y,
    int w,
    int h,
    uchar r,
    uchar g,
    uchar b) [inline]
```

Color a rectangle with "exactly" the passed `r, g, b` color.

On screens with less than 24 bits of color this is done by drawing a solid-colored block using [fl_draw_image\(\)](#) so that the correct color shade is produced. On other screens, the current color is changed to `fl_color(r, g, b)` upon return.

10.7.3.67 fl_rescale_offscreen()

```
void fl_rescale_offscreen (
    Fl_Offscreen & ctx)
```

Adapts an offscreen buffer to a changed value of the scale factor.

The `ctx` argument must have been created by [fl_create_offscreen\(\)](#) and the calling context must not be between [fl_begin_offscreen\(\)](#) and [fl_end_offscreen\(\)](#). The graphical content of the offscreen is preserved. The current scale factor value is given by `Fl_Graphics_Driver::default_driver().scale()`.

Version

1.4

10.7.3.68 fl_reset_spot()

```
void fl_reset_spot (
    void )
```

Resets marked text.

In many languages, typing a character can involve multiple keystrokes. For example, the Å can be composed of two dots (¨) on top of the character, followed by the letter A (on a Mac with U.S. keyboard, you'd type Alt-U, Shift-A. To inform the user that the dots may be followed by another character, the ¨ is underlined).

Call this function if character composition needs to be aborted for some reason. One such example would be the text input widget losing focus.

10.7.3.69 fl_restore_scale()

```
void fl_restore_scale (
    float s)
```

Restores the GUI scaling factor in subsequent drawing operations.

Parameters

s	Value returned by a previous call to fl_override_scale() .

10.7.3.70 fl_rotate()

```
void fl_rotate (
    double d) [inline]
```

Concatenate rotation transformation onto the current one.

Parameters

in	<i>d</i>	- rotation angle, counter-clockwise in degrees (not radians)

10.7.3.71 fl_rounded_rect()

```
void fl_rounded_rect (
    int x,
    int y,
    int w,
    int h,
    int r) [inline]
```

Draw a rounded border *inside* the given bounding box.

The radius code is optimized for speed and works best for values between 5 and 15 units.

10.7.3.72 fl_rounded_rectf()

```
void fl_rounded_rectf (
    int x,
    int y,
    int w,
    int h,
    int r) [inline]
```

Color with current color a rounded rectangle that exactly fills the given bounding box.

The radius code is optimized for speed and works best for values between 5 and 15 units.

10.7.3.73 fl_scale() [1/2]

```
void fl_scale (
    double x) [inline]
```

Concatenate scaling transformation onto the current one.

Parameters

in	<i>x</i>	scale factor in both x-direction and y-direction

10.7.3.74 fl_scale() [2/2]

```
void fl_scale (
    double x,
    double y) [inline]
```

Concatenate scaling transformation onto the current one.

Parameters

in	<i>x,y</i>	scale factors in x-direction and y-direction

10.7.3.75 fl_scroll()

```
void fl_scroll (
    int X,
    int Y,
    int W,
    int H,
    int dx,
```

```

    int dy,
    void(* draw_area )(void *, int, int, int, int),
    void * data)

```

Scroll a rectangle and draw the newly exposed portions.

Parameters

in	<i>X,Y</i>	position of top-left of rectangle
in	<i>W,H</i>	size of rectangle
in	<i>dx,dy</i>	pixel offsets for shifting rectangle
in	<i>draw_area</i>	callback function to draw rectangular areas
in	<i>data</i>	pointer to user data for callback The contents of the rectangular area is first shifted by <i>dx</i> and <i>dy</i> pixels. The <i>draw_area</i> callback is then called for every newly exposed rectangular area.

Warning

With FLTK 1.4 and above, it's recommended to put graphical elements in an [Fl_Scroll](#) widget rather than use function [fl_scroll\(\)](#) to update those elements. That's because [fl_scroll\(\)](#) may not produce a pixel-accurate result when the GUI scaling factor value is not a multiple of 100%. Alternatively, use [fl_scroll\(\)](#) when the scaling factor value is a multiple of 100% and perform a full redraw of the graphical elements otherwise. Statement `float s = Fl::screen\_scale(win->screen_num());` gives the current scaling factor value given [Fl_Window](#) *win.

10.7.3.76 fl_set_spot()

```

void fl_set_spot (
    int font,
    int size,
    int X,
    int Y,
    int W,
    int H,
    Fl\_Window * win = 0)

```

Inform text input methods about the current text insertion cursor.

Parameters

<i>font</i>	Font currently in use in text input.
<i>size</i>	Size of the current font.
<i>X,Y</i>	Position of the bottom of the current text insertion cursor.
<i>W,H</i>	Width and height of the current text insertion cursor.
<i>win</i>	Points to the Fl_Window object containing the current text widget, or NULL.

10.7.3.77 fl_set_status()

```

void fl_set_status (
    int X,
    int Y,
    int W,
    int H)

```

Related to text input methods under X11.

This function is presently used only by the `utf8` test application and only for the X11 platform. This function is apparently not indispensable for text input to work correctly as suggested by other apps that don't use it (e.g., editor).

10.7.3.78 fl_shortcut_label() [1/2]

```
const char * fl_shortcut_label (
    unsigned int shortcut)
```

Get a human-readable string from a shortcut value.

Unparse a shortcut value as used by [Fl_Button](#) or [Fl_Menu_Item](#) into a human-readable string like "Alt+N". This only works if the shortcut is a character key or a numbered function key. If the shortcut is zero then an empty string is returned. The return value points at a static buffer that is overwritten with each call.

Since

FLTK 1.3.4 modifier key names can be localized, but key names can not yet be localized. This may be added to a future FLTK version.

Modifier key names (human-readable shortcut names) can be defined with the following global const char * pointer variables:

- `fl_local_ctrl` => name of FL_CTRL
- `fl_local_alt` => name of FL_ALT
- `fl_local_shift` => name of FL_SHIFT
- `fl_local_meta` => name of FL_META

```
fl_local_ctrl = "Strg";      // German for "Ctrl"
fl_local_shift = "Umschalt"; // German for "Shift"
```

Note

Due to **random** static initialization order this should always be done from code in `main()` or called by `main()` as opposed to static initialization since the default strings in the FLTK library are set by static initializers. Otherwise this **might** result in the wrong order so FLTK's internal initialization overwrites your strings.

The shortcut name will be constructed by adding all modifier names in the order defined above plus the name of the key. A '+' character is added to each modifier name unless it has a trailing '\' or a trailing '+'.
Example:

Ctrl+Alt+Shift+Meta+F12

The default values for modifier key names are as given above for all platforms except macOS. macOS uses graphical characters that represent the typical macOS modifier names in menus, e.g. cloverleaf, saucepan, etc. You may, however, redefine macOS modifier names as well.

Parameters

in	<i>shortcut</i>	the integer value containing the ASCII character or extended keystroke plus modifiers

Returns

a pointer to a static buffer containing human readable text for the shortcut

10.7.3.79 fl_shortcut_label() [2/2]

```
const char * fl_shortcut_label (
    unsigned int shortcut,
    const char ** eom)
```

Get a human-readable string from a shortcut value.

Parameters

in	<i>shortcut</i>	the integer value containing the ASCII character or extended keystroke plus modifiers
in	<i>eom</i>	if this pointer is set, it will receive a pointer to the end of the modifier text

Returns

a pointer to a static buffer containing human readable text for the shortcut

See also

[fl_shortcut_label\(unsigned int shortcut\)](#)

10.7.3.80 fl_transform_dx()

```
double fl_transform_dx (  
    double x,  
    double y) [inline]
```

Transform distance using current transformation matrix.

Parameters

in	x,y	coordinate

10.7.3.81 fl_transform_dy()

```
double fl_transform_dy (  
    double x,  
    double y) [inline]
```

Transform distance using current transformation matrix.

Parameters

in	x,y	coordinate

10.7.3.82 fl_transform_x()

```
double fl_transform_x (  
    double x,  
    double y) [inline]
```

Transform coordinate using the current transformation matrix.

Parameters

in	x,y	coordinate

10.7.3.83 fl_transform_y()

```
double fl_transform_y (  
    double x,  
    double y) [inline]
```

Transform coordinate using the current transformation matrix.

Parameters

in	x,y	coordinate

10.7.3.84 fl_transformed_vertex()

```
void fl_transformed_vertex (  
    double xf,  
    double yf) [inline]
```

Add coordinate pair to the vertex list without further transformations.

Parameters

in	<i>xf,yf</i>	transformed coordinate

10.7.3.85 fl_translate()

```
void fl_translate (
    double x,
    double y) [inline]
```

Concatenate translation transformation onto the current one.

Parameters

in	<i>x,y</i>	translation factor in x-direction and y-direction

10.7.3.86 fl_vertex()

```
void fl_vertex (
    double x,
    double y) [inline]
```

Add a single vertex to the current path.

Parameters

in	<i>x,y</i>	coordinate

10.8 Multithreading support functions

fl multithreading support functions declared in <FL/FL.H>

Functions

- static int [Fl::awake](#) ([Fl_Awake_Handler](#) cb, void *message=0)
See void awake(void message=0).*
- static void [Fl::awake](#) (void *message=0)
Sends a message pointer to the main thread, causing any pending [Fl::wait\(\)](#) call to terminate so that the main thread can retrieve the message and any pending redraws can be processed.
- static int [Fl::lock](#) ()
The [lock\(\)](#) method blocks the current thread until it can safely access FLTK widgets and data.
- static void * [Fl::thread_message](#) ()
The [thread_message\(\)](#) method returns the last message that was sent from a child by the [awake\(\)](#) method.
- static void [Fl::unlock](#) ()
The [unlock\(\)](#) method releases the lock that was set using the [lock\(\)](#) method.

10.8.1 Detailed Description

fl multithreading support functions declared in <FL/FL.H>

10.8.2 Function Documentation

10.8.2.1 `awake()` [1/2]

```
int Fl::awake (
    Fl_Awake_Handler func,
    void * data = 0) [static]
```

See `void awake(void* message=0)`.

Let the main thread know an update is pending and have it call a specific function.

Registers a function that will be called by the main thread during the next message handling cycle. Returns 0 if the callback function was registered, and -1 if registration failed. Over a thousand awake callbacks can be registered simultaneously.

See also

[Fl::awake\(void* message=0\)](#)

10.8.2.2 `awake()` [2/2]

```
void Fl::awake (
    void * msg = 0) [static]
```

Sends a message pointer to the main thread, causing any pending [Fl::wait\(\)](#) call to terminate so that the main thread can retrieve the message and any pending redraws can be processed.

Multiple calls to [Fl::awake\(\)](#) will queue multiple pointers for the main thread to process, up to a system-defined (typically several thousand) depth. The default message handler saves the last message which can be accessed using the [Fl::thread_message\(\)](#) function.

In the context of a threaded application, a call to [Fl::awake\(\)](#) with no argument will trigger event loop handling in the main thread. Since it is not possible to call [Fl::flush\(\)](#) from a subsidiary thread, [Fl::awake\(\)](#) is the best (and only, really) substitute.

It's *not* necessary to wrap calls to any form of [Fl::awake\(\)](#) by [Fl::lock\(\)](#) and [Fl::unlock\(\)](#). Nevertheless, the early, single call to [Fl::lock\(\)](#) used to initialize threading support is necessary.

Function [Fl::awake\(\)](#) in all its forms is typically called by worker threads, but it can be used safely by the main thread too, as a means to break the event loop.

See also

[Multithreading](#)

10.8.2.3 `lock()`

```
int Fl::lock () [static]
```

The [lock\(\)](#) method blocks the current thread until it can safely access FLTK widgets and data.

Child threads should call this method prior to updating any widgets or accessing data. The main thread must call [lock\(\)](#) to initialize the threading support in FLTK. [lock\(\)](#) will return non-zero if threading is not available on the platform.

Child threads must call [unlock\(\)](#) when they are done accessing FLTK.

When the [wait\(\)](#) method is waiting for input or timeouts, child threads are given access to FLTK. Similarly, when the main thread needs to do processing, it will wait until all child threads have called [unlock\(\)](#) before processing additional data.

Returns

0 if threading is available on the platform; non-zero otherwise.

See also: [Multithreading](#)

10.8.2.4 `thread_message()`

```
void * Fl::thread_message () [static]
```

The [thread_message\(\)](#) method returns the last message that was sent from a child by the [awake\(\)](#) method.

See also: [Multithreading](#)

10.8.2.5 unlock()

```
void Fl::unlock () [static]
```

The `unlock()` method releases the lock that was set using the `lock()` method.

Child threads should call this method as soon as they are finished accessing FLTK.

See also: [Multithreading](#)

10.9 Safe widget deletion support functions

Functions

- static void `Fl::clear_widget_pointer (Fl_Widget const *w)`
Clears a widget pointer in the watch list.
- static void `Fl::delete_widget (Fl_Widget *w)`
Schedules a widget for deletion at the next call to the event loop.
- static void `Fl::do_widget_deletion ()`
Deletes widgets previously scheduled for deletion.
- static void `Fl::release_widget_pointer (Fl_Widget *&w)`
Releases a widget pointer from the watch list.
- static void `Fl::watch_widget_pointer (Fl_Widget *&w)`
Adds a widget pointer to the widget watch list.

10.9.1 Detailed Description

These functions, declared in `<FL/Fl.H>`, support deletion of widgets inside callbacks.

`Fl::delete_widget()` should be called when deleting widgets or complete widget trees (`Fl_Group`, `Fl_Window`, ...) inside callbacks.

The other functions are intended for internal use. The preferred way to use them is by using the helper class `Fl_Widget_Tracker`.

The following is to show how it works ...

There are three groups of related methods:

1. scheduled widget deletion

- `Fl::delete_widget()` schedules widgets for deletion
- `Fl::do_widget_deletion()` deletes all scheduled widgets

2. widget watch list ("smart pointers")

- `Fl::watch_widget_pointer()` adds a widget pointer to the watch list
- `Fl::release_widget_pointer()` removes a widget pointer from the watch list
- `Fl::clear_widget_pointer()` clears a widget pointer *in* the watch list

3. the class `Fl_Widget_Tracker`:

- the constructor calls `Fl::watch_widget_pointer()`
- the destructor calls `Fl::release_widget_pointer()`
- the access methods can be used to test, if a widget has been deleted

See also

[Fl_Widget_Tracker](#).

10.9.2 Function Documentation

10.9.2.1 `clear_widget_pointer()`

```
void Fl::clear_widget_pointer (
    Fl_Widget const * w) [static]
```

Clears a widget pointer *in* the watch list.

This is called when a widget is destroyed (by its destructor). You should never call this directly.

Note

Internal use only !

This method searches the widget watch list for pointers to the widget and clears each pointer that points to it. Widget pointers can be added to the widget watch list by calling `Fl::watch_widget_pointer()` or by using the helper class `Fl_Widget_Tracker` (recommended).

See also

`Fl::watch_widget_pointer()`
 class `Fl_Widget_Tracker`

10.9.2.2 `delete_widget()`

```
void Fl::delete_widget (
    Fl_Widget * wi) [static]
```

Schedules a widget for deletion at the next call to the event loop.

Use this method to delete a widget inside a callback function.

To avoid early deletion of widgets, this function should be called toward the end of a callback and only after any call to the event loop (`Fl::wait()`, `Fl::flush()`, `Fl::check()`, `fl_ask()`, etc.).

When deleting groups or windows, you must only delete the group or window widget and not the individual child widgets.

Since

FLTK 1.3.4 the widget will be hidden immediately, but the actual destruction will be delayed until the event loop is finished. Up to FLTK 1.3.3 windows wouldn't be hidden before the event loop was done, hence you had to `hide()` a window in your window close callback if you called `Fl::delete_widget()` to destroy (and hide) the window.

FLTK 1.3.0 it is not necessary to remove widgets from their parent groups or windows before calling this, because it will be done in the widget's destructor, but it is not a failure to do this nevertheless.

Note

In FLTK 1.1 you **must** remove widgets from their parent group (or window) before deleting them.

See also

`Fl_Widget::~~Fl_Widget()`

10.9.2.3 `do_widget_deletion()`

```
void Fl::do_widget_deletion () [static]
```

Deletes widgets previously scheduled for deletion.

This is for internal use only. You should never call this directly.

`Fl::do_widget_deletion()` is called from the FLTK event loop or whenever you call `Fl::wait()`. The previously scheduled widgets are deleted in the same order they were scheduled by calling `Fl::delete_widget()`.

See also

`Fl::delete_widget(Fl_Widget *wi)`

10.9.2.4 `release_widget_pointer()`

```
void Fl::release_widget_pointer (
    Fl_Widget *& w) [static]
```

Releases a widget pointer from the watch list.

This is used to remove a widget pointer that has been added to the watch list with [Fl::watch_widget_pointer\(\)](#), when it is not needed anymore.

Note

Internal use only, please use class [Fl_Widget_Tracker](#) instead.

See also

[Fl::watch_widget_pointer\(\)](#)

10.9.2.5 `watch_widget_pointer()`

```
void Fl::watch_widget_pointer (
    Fl_Widget *& w) [static]
```

Adds a widget pointer to the widget watch list.

Note

Internal use only, please use class [Fl_Widget_Tracker](#) instead.

This can be used, if it is possible that a widget might be deleted during a callback or similar function. The widget pointer must be added to the watch list before calling the callback. After the callback the widget pointer can be queried, if it is NULL. If it is NULL, then the widget has been deleted during the callback and must not be accessed anymore. If the widget pointer is *not* NULL, then the widget has not been deleted and can be accessed safely.

After accessing the widget, the widget pointer must be released from the watch list by calling [Fl::release_widget_pointer\(\)](#).

Example for a button that is clicked (from its [handle\(\)](#) method):

```
Fl_Widget *wp = this;           // save 'this' in a pointer variable
Fl::watch_widget_pointer(wp);    // add the pointer to the watch list
set_changed();                  // set the changed flag
do_callback();                  // call the callback
if (!wp) {                      // the widget has been deleted

    // DO NOT ACCESS THE DELETED WIDGET !

} else {                        // the widget still exists
    clear_changed();            // reset the changed flag
}
```

```
Fl::release_widget_pointer(wp); // remove the pointer from the watch list
```

This works, because all widgets call [Fl::clear_widget_pointer\(\)](#) in their destructors.

See also

[Fl::release_widget_pointer\(\)](#)

[Fl::clear_widget_pointer\(\)](#)

An easier and more convenient method to control widget deletion during callbacks is to use the class [Fl_Widget_Tracker](#) with a local (automatic) variable.

See also

class [Fl_Widget_Tracker](#)

10.10 Cairo Support Functions and Classes

Classes

- class [Fl_Cairo_State](#)
Contains all the necessary info on the current cairo context.
- class [Fl_Cairo_Window](#)
This defines an FLTK window with Cairo support.

Functions

- static bool [Fl::cairo_autolink_context](#) ()
Gets the current autolink mode for Cairo support.
- static void [Fl::cairo_autolink_context](#) (bool alink)
When FLTK_HAVE_CAIRO is defined and [cairo_autolink_context\(\)](#) is true, any current window dc is linked to a current Cairo context.
- static cairo_t * [Fl::cairo_cc](#) ()
Gets the current Cairo context linked with a fltk window.
- static void [Fl::cairo_cc](#) (cairo_t *c, bool own=false)
Sets the current Cairo context to c.
- static void [Fl::cairo_flush](#) (cairo_t *c)
Flush Cairo drawings on Cairo context c.
- static cairo_t * [Fl::cairo_make_current](#) (Fl_Window *w)
Provides a Cairo context for window wi.

10.10.1 Detailed Description

10.10.2 Function Documentation

10.10.2.1 [cairo_autolink_context\(\)](#) [1/2]

```
static bool Fl::cairo_autolink_context () [inline], [static]
```

Gets the current autolink mode for Cairo support.

Return values

<i>false</i>	if no Cairo context autolink is made for each window.
<i>true</i>	if any fltk window is attached a Cairo context when it is current.

See also

void [cairo_autolink_context](#)(bool alink)

Note

Only available when configure has the `--enable-cairo` option

10.10.2.2 [cairo_autolink_context\(\)](#) [2/2]

```
static void Fl::cairo_autolink_context (
    bool alink) [inline], [static]
```

When FLTK_HAVE_CAIRO is defined and [cairo_autolink_context\(\)](#) is true, any current window dc is linked to a current Cairo context.

This is not the default, because it may not be necessary to add Cairo support to all fltk supported windows. When you wish to associate a Cairo context in this mode, you need to call explicitly in your draw() overridden method, [Fl::cairo_make_current\(Fl_Window*\)](#). This will create a Cairo context only for this Window. Still in custom Cairo application it is possible to handle completely this process automatically by setting `alink` to true. In this last case, you don't need anymore to call [Fl::cairo_make_current\(\)](#). You can use [Fl::cairo_cc\(\)](#) to get the current Cairo context anytime.

Note

Only available when configure has the `--enable-cairo` option

10.10.2.3 `cairo_cc()`

```
static void Fl::cairo_cc (
    cairo_t * c,
    bool own = false) [inline], [static]
```

Sets the current Cairo context to `c`.

Set `own` to true if you want fltk to handle this cc deletion.

Note

Only available when configure has the `--enable-Cairo` option

10.10.2.4 `cairo_flush()`

```
static void Fl::cairo_flush (
    cairo_t * c) [inline], [static]
```

Flush Cairo drawings on Cairo context `c`.

This is **required** on Windows if you use the Cairo context provided by the "Cairo autolink" option. Call this when all your drawings on the Cairo context are finished. This is maybe not necessary on other platforms than Windows but it does no harm if you call it always.

You don't need to use this if you use an [Fl_Cairo_Window](#) which does this automatically after the draw callback returns.

Code example for "Cairo autolink" mode:

In the overridden `draw()` method of your subclass of [Fl_Window](#) or any widget:

```
cairo_t *cc = Fl::cairo_cc(); // get the "autolink" Cairo context
// ... your Cairo drawings are here ...
Fl::cairo_flush(cc); // flush Cairo drawings to the device
```

If you configure FLTK with `'--enable-cairo'` or CMake option `'FLTK_OPTION_CAIRO_WINDOW'` (i.e. without `'--enable-cairoext'` or CMake option `'FLTK_OPTION_CAIRO_EXT'`) or if you don't enable the 'autolink' Cairo context you may do the equivalent to use Cairo drawings in an overridden `draw()` method of derived classes by using

```
// get the Cairo context for the \c window
cairo_t *cc = Fl::cairo_make_current(window);
// ... your Cairo drawings are here ...
Fl::cairo_flush(cc); // flush Cairo drawings to the device
```

See also

[Fl::cairo_autolink_context\(bool\)](#)

[Fl::cairo_make_current\(Fl_Window*\)](#);

10.10.2.5 `cairo_make_current()`

```
cairo_t * Fl::cairo_make_current (
    Fl_Window * wi) [static]
```

Provides a Cairo context for window `wi`.

This is needed in a `draw()` override if [Fl::cairo_autolink_context\(\)](#) returns false, which is the default. The `cairo_↵context()` does not need to be freed as it is freed every time a new Cairo context is created. When the program terminates, a call to `Fl::cairo_make_current(0)` will destroy any residual context.

Note

A new Cairo context is not always re-created when this method is used. In particular, if the current graphical context and the current window didn't change between two calls, the previous gc is internally kept, thus optimizing the drawing performances. Also, after this call, [Fl::cairo_cc\(\)](#) is adequately updated with this Cairo context.

Only available when CMake option `'-D FLTK_OPTION_CAIRO_WINDOW'` is set.

Returns

The valid `cairo_t *cairo` context associated to this window.

Return values

<code>NULL</code>	if <code>wi</code> is <code>NULL</code> or maybe with GL windows under Wayland

10.11 Unicode and UTF-8 functions

fl global Unicode and UTF-8 handling functions declared in `<FL/fl_utf8.h>`

Macros

- `#define ERRORS_TO_CP1252 1`
Set to 1 to turn bad UTF-8 bytes in the 0x80-0x9f range into the Unicode index for Microsoft's CP1252 character set.
- `#define ERRORS_TO_ISO8859_1 1`
Set to 1 to turn bad UTF-8 bytes into ISO-8859-1.
- `#define NBC 0xFFFF + 1`
- `#define STRICT_RFC3629 0`
A number of Unicode code points are in fact illegal and should not be produced by a UTF-8 converter.

Functions

- `int fl_access (const char *f, int mode)`
Cross-platform function to test a files access() with a UTF-8 encoded name or value.
- `int fl_chdir (const char *path)`
Cross-platform function to change the current working directory, given as a UTF-8 encoded string.
- `int fl_chmod (const char *f, int mode)`
Cross-platform function to set a files mode() with a UTF-8 encoded name or value.
- `int fl_close_fd (int fd)`
Cross-platform function to close a file descriptor.
- `int fl_execvp (const char *file, char *const *argv)`
- `FILE * fl_fopen (const char *f, const char *mode)`
Cross-platform function to open files with a UTF-8 encoded name.
- `char * fl_getcwd (char *buf, int len)`
Cross-platform function to get the current working directory as a UTF-8 encoded value.
- `char * fl_getenv (const char *v)`
Cross-platform function to get environment variables with a UTF-8 encoded name or value.
- `char fl_make_path (const char *path)`
Cross-platform function to recursively create a path in the file system.
- `void fl_make_path_for_file (const char *path)`
Cross-platform function to create a path for the file in the file system.
- `int fl_mkdir (const char *f, int mode)`
Cross-platform function to create a directory with a UTF-8 encoded name.
- `unsigned int fl_nonspacing (unsigned int ucs)`
Returns true if the Unicode character `ucs` is non-spacing.
- `int fl_open (const char *fname, int oflags,...)`
Cross-platform function to open files with a UTF-8 encoded name.
- `int fl_open_ext (const char *fname, int binary, int oflags,...)`
Cross-platform function to open files with a UTF-8 encoded name.
- `int fl_putenv (const char *var)`
Cross-platform function to write environment variables with a UTF-8 encoded name or value.
- `int fl_rename (const char *f, const char *n)`
Cross-platform function to rename a filesystem object using UTF-8 encoded names.

- `int fl_rmdir (const char *f)`
Cross-platform function to remove a directory with a UTF-8 encoded name.
- `int fl_stat (const char *f, struct stat *b)`
Cross-platform function to stat() a file using a UTF-8 encoded name or value.
- `int fl_system (const char *cmd)`
Cross-platform function to run a system command with a UTF-8 encoded string.
- `int fl_tolower (unsigned int ucs)`
Returns the Unicode lower case value of ucs.
- `int fl_toupper (unsigned int ucs)`
Returns the Unicode upper case value of ucs.
- `unsigned fl_ucs_to_Utf16 (const unsigned ucs, unsigned short *dst, const unsigned dstlen)`
Convert a single 32-bit Unicode codepoint into an array of 16-bit characters.
- `int fl_unlink (const char *fname)`
Cross-platform function to unlink() (that is, delete) a file using a UTF-8 encoded filename.
- `char * fl_utf2mbcs (const char *s)`
Converts UTF-8 string s to a local multi-byte character string.
- `const char * fl_utf8back (const char *p, const char *start, const char *end)`
Move p backward until it points to the start of a UTF-8 character.
- `int fl_utf8bytes (unsigned ucs)`
Return the number of bytes needed to encode the given UCS4 character in UTF-8.
- `unsigned fl_utf8decode (const char *p, const char *end, int *len)`
Decode a single UTF-8 encoded character starting at p.
- `int fl_utf8encode (unsigned ucs, char *buf)`
Write the UTF-8 encoding of ucs into buf and return the number of bytes written.
- `unsigned fl_utf8from_mb (char *dst, unsigned dstlen, const char *src, unsigned srclen)`
Convert a filename from the locale-specific multibyte encoding used by Windows to UTF-8 as used by FLTK.
- `unsigned fl_utf8froma (char *dst, unsigned dstlen, const char *src, unsigned srclen)`
Convert an ISO-8859-1 (ie normal c-string) byte stream to UTF-8.
- `unsigned fl_utf8fromwc (char *dst, unsigned dstlen, const wchar_t *src, unsigned srclen)`
Turn "wide characters" as returned by some system calls (especially on Windows) into UTF-8.
- `const char * fl_utf8fwd (const char *p, const char *start, const char *end)`
Move p forward until it points to the start of a UTF-8 character.
- `int fl_utf8len (char c)`
Returns the byte length of the UTF-8 sequence with first byte c, or -1 if c is not valid.
- `int fl_utf8len1 (char c)`
Returns the byte length of the UTF-8 sequence with first byte c, or 1 if c is not valid.
- `int fl_utf8locale ()`
Return true if the "locale" seems to indicate that UTF-8 encoding is used.
- `int fl_utf8strlen (const char *text, int len)`
Return the length in bytes of a UTF-8 string.
- `int fl_utf8test (const char *src, unsigned srclen)`
Examines the first srclen bytes in src and returns a verdict on whether it is UTF-8 or not.
- `unsigned fl_utf8to_mb (const char *src, unsigned srclen, char *dst, unsigned dstlen)`
Convert the UTF-8 used by FLTK to the locale-specific encoding used for filenames (and sometimes used for data in files).
- `unsigned fl_utf8toa (const char *src, unsigned srclen, char *dst, unsigned dstlen)`
Convert a UTF-8 sequence into an array of 1-byte characters.
- `unsigned fl_utf8toUtf16 (const char *src, unsigned srclen, unsigned short *dst, unsigned dstlen)`
Convert a UTF-8 sequence into an array of 16-bit characters.
- `unsigned fl_utf8towc (const char *src, unsigned srclen, wchar_t *dst, unsigned dstlen)`
Converts a UTF-8 string into a wide character string.

- `int fl_utf_nb_char` (const unsigned char *buf, int len)
Returns the number of Unicode chars in the UTF-8 string.
- `int fl_utf_strcasecmp` (const char *s1, const char *s2)
UTF-8 aware strcasecmp - converts to Unicode and tests.
- `int fl_utf_strncasecmp` (const char *s1, const char *s2, int n)
UTF-8 aware strncasecmp - converts to lower case Unicode and tests.
- `int fl_utf_tolower` (const unsigned char *str, int len, char *buf)
Converts the string `str` to its lower case equivalent into `buf`.
- `int fl_utf_toupper` (const unsigned char *str, int len, char *buf)
Converts the string `str` to its upper case equivalent into `buf`.
- `int fl_wcwidth` (const char *src)
extended wrapper around `fl_wcwidth_(unsigned int ucs)` function.
- `int fl_wcwidth_` (unsigned int ucs)
Wrapper to adapt Markus Kuhn's implementation of `wcwidth()` for FLTK.

10.11.1 Detailed Description

fl global Unicode and UTF-8 handling functions declared in `<FL/fl_utf8.h>`

10.11.2 Macro Definition Documentation

10.11.2.1 ERRORS_TO_CP1252

```
#define ERRORS_TO_CP1252 1
```

Set to 1 to turn bad UTF-8 bytes in the 0x80-0x9f range into the Unicode index for Microsoft's CP1252 character set.

You should also set `ERRORS_TO_ISO8859_1`. With this a huge amount of more available text (such as all web pages) are correctly converted to Unicode.

10.11.2.2 ERRORS_TO_ISO8859_1

```
#define ERRORS_TO_ISO8859_1 1
```

Set to 1 to turn bad UTF-8 bytes into ISO-8859-1.

If this is zero they are instead turned into the Unicode REPLACEMENT CHARACTER, of value 0xfffd. If this is on `fl_utf8decode()` will correctly map most (perhaps all) human-readable text that is in ISO-8859-1. This may allow you to completely ignore character sets in your code because virtually everything is either ISO-8859-1 or UTF-8.

10.11.2.3 STRICT_RFC3629

```
#define STRICT_RFC3629 0
```

A number of Unicode code points are in fact illegal and should not be produced by a UTF-8 converter.

Turn this on will replace the bytes in those encodings with errors. If you do this then converting arbitrary 16-bit data to UTF-8 and then back is not an identity, which will probably break a lot of software.

10.11.3 Function Documentation

10.11.3.1 fl_access()

```
int fl_access (
    const char * f,
    int mode)
```

Cross-platform function to test a files `access()` with a UTF-8 encoded name or value.

This function is especially useful on the Windows platform where the standard `access()` function fails with UTF-8 encoded non-ASCII filenames.

Windows defines the mode values 0 for existence, 2 for writable, 4 for readable, and 6 of readable and writable. On other systems, the modes `X_OK`, `W_OK`, and `R_OK` are usually defined as 1, 2, and 4.

Upon successful completion, the value 0 is returned on all platforms.

Parameters

in	<i>f</i>	the UTF-8 encoded filename
in	<i>mode</i>	the mode to test

Returns

the return value of `_waccess()` on Windows or `access()` on other platforms.

10.11.3.2 `fl_chdir()`

```
int fl_chdir (
    const char * path)
```

Cross-platform function to change the current working directory, given as a UTF-8 encoded string.

This function is especially useful on the Windows platform where the standard `_wchdir()` function needs a `path` in UTF-16 encoding.

The `path` is converted to a system specific encoding if necessary and the system specific `chdir(converted↵_path)` function is called.

The function returns 0 on success and -1 on error. Depending on the platform, `errno` **may** be set if an error occurs.

Note

The possible `errno` values are platform specific. Refer to the documentation of the platform specific `chdir()` function.

If the function is not implemented on a particular platform the default implementation returns -1 and `errno` is **not** set.

If the `path` is `NULL` the function returns -1, but `errno` is **not** changed. This is a convenience feature of `fl_chdir()` as opposed to `chdir()`.

Parameters

in	<i>path</i>	the target directory for <code>chdir</code> (may be <code>NULL</code>)

Returns

0 if successful, -1 on error (`errno` may be set)

10.11.3.3 `fl_chmod()`

```
int fl_chmod (
    const char * f,
    int mode)
```

Cross-platform function to set a files `mode()` with a UTF-8 encoded name or value.

This function is especially useful on the Windows platform where the standard `chmod()` function fails with UTF-8 encoded non-ASCII filenames.

Parameters

in	<i>f</i>	the UTF-8 encoded filename
in	<i>mode</i>	the mode to set

Returns

the return value of `_wchmod()` on Windows or `chmod()` on other platforms.

10.11.3.4 fl_close_fd()

```
int fl_close_fd (
    int fd)
```

Cross-platform function to close a file descriptor.

Returns

0 in case of success, or -1 in case of error.

10.11.3.5 fl_fopen()

```
FILE * fl_fopen (
    const char * f,
    const char * mode)
```

Cross-platform function to open files with a UTF-8 encoded name.

This function is especially useful on the Windows platform where the standard fopen() function fails with UTF-8 encoded non-ASCII filenames.

Parameters

<i>f</i>	the UTF-8 encoded filename
<i>mode</i>	same as the second argument of the standard fopen() function

Returns

a FILE pointer upon successful completion, or NULL in case of error.

See also

[fl_open\(\)](#).

10.11.3.6 fl_getcwd()

```
char * fl_getcwd (
    char * buf,
    int len)
```

Cross-platform function to get the current working directory as a UTF-8 encoded value.

This function is especially useful on the Windows platform where the standard _wgetcwd() function returns UTF-16 encoded non-ASCII filenames.

If buf is NULL a buffer of size (len+1) is allocated, filled with the current working directory, and returned. In this case the buffer must be released by the caller with free() to prevent memory leaks.

Parameters

in	<i>buf</i>	the buffer to populate (may be NULL)
in	<i>len</i>	the length of the buffer

Returns

the CWD encoded as UTF-8

10.11.3.7 fl_getenv()

```
char * fl_getenv (
    const char * v)
```

Cross-platform function to get environment variables with a UTF-8 encoded name or value.

This function is especially useful on the Windows platform where non-ASCII environment variables are encoded as wide characters. The returned value of the variable is encoded in UTF-8 as well.

On platforms other than Windows this function calls `getenv` directly. The return value is returned as-is.

The return value is a pointer to an implementation defined buffer:

- an internal buffer that is (re)allocated as needed (Windows) or
- the string in the environment itself (Unix, Linux, MacOS) or
- any other implementation (other platforms). This string must be considered read-only and must not be freed by the caller.

If the resultant string is to be used later it must be copied to a safe place. The next call to `fl_getenv()` or any other environment changes may overwrite the string.

Note

This function is not thread-safe.

Parameters

in	v	the UTF-8 encoded environment variable

Returns

the environment variable in UTF-8 encoding, or NULL in case of error.

10.11.3.8 fl_make_path()

```
char fl_make_path (
    const char * path)
```

Cross-platform function to recursively create a path in the file system.

This function creates a `path` in the file system by recursively creating all directories.

Parameters

in	path	a Unix style ('/' forward slashes) absolute or relative pathname

Returns

1 if the path was created, 0 if creating the path failed at some point

10.11.3.9 fl_make_path_for_file()

```
void fl_make_path_for_file (
    const char * path)
```

Cross-platform function to create a path for the file in the file system.

This function strips the filename from the given `path` and creates a path in the file system by recursively creating all directories.

10.11.3.10 fl_mkdir()

```
int fl_mkdir (
    const char * f,
    int mode)
```

Cross-platform function to create a directory with a UTF-8 encoded name.

This function is especially useful on the Windows platform where the standard `_wmkdir()` function expects UTF-16 encoded non-ASCII filenames.

Parameters

in	<i>f</i>	the UTF-8 encoded filename
in	<i>mode</i>	the mode of the directory

Returns

the return value of `_wmkdir()` on Windows or `mkdir()` on other platforms.

10.11.3.11 `fl_nonspacing()`

```
unsigned int fl_nonspacing (  
    unsigned int ucs)
```

Returns true if the Unicode character `ucs` is non-spacing.

Non-spacing characters in Unicode are typically combining marks like tilde (~), diaeresis (¨), or other marks that are added to a base character, for instance 'a' (base character) + '¨' (combining mark) = 'ä' (German Umlaut).

- http://unicode.org/glossary/#base_character
- http://unicode.org/glossary/#nonspacing_mark
- http://unicode.org/glossary/#combining_character

10.11.3.12 `fl_open()`

```
int fl_open (  
    const char * fname,  
    int oflags,  
    ...)
```

Cross-platform function to open files with a UTF-8 encoded name.

This function is especially useful on the Windows platform where the standard `open()` function fails with UTF-8 encoded non-ASCII filenames.

Parameters

in	<i>fname</i>	the UTF-8 encoded filename
in	<i>oflags</i>	other arguments are as in the standard <code>open()</code> function

Returns

a file descriptor upon successful completion, or -1 in case of error.

See also

[fl_fopen\(\)](#), [fl_open_ext\(\)](#), [fl_close_fd\(int fd\)](#).

10.11.3.13 `fl_open_ext()`

```
int fl_open_ext (  
    const char * fname,  
    int binary,  
    int oflags,  
    ...)
```

Cross-platform function to open files with a UTF-8 encoded name.

In comparison with [fl_open\(\)](#), this function allows to control whether the file is opened in binary (a.k.a. untranslated) mode. This is especially useful on the Windows platform where files are by default opened in text (translated) mode.

Parameters

in	<i>fname</i>	the UTF-8 encoded filename
in	<i>binary</i>	if non-zero, the file is to be accessed in binary (a.k.a. untranslated) mode.
in	<i>oflags,...</i>	these arguments are as in the standard <code>open()</code> function. Setting <code>oflags</code> to zero opens the file for reading.

Returns

a file descriptor upon successful completion, or -1 in case of error.

10.11.3.14 fl_putenv()

```
int fl_putenv (
    const char * var)
```

Cross-platform function to write environment variables with a UTF-8 encoded name or value.

This function is especially useful on the Windows platform where non-ASCII environment variables are encoded as wide characters.

The given argument `var` must be encoded in UTF-8 in the form "name=value". The 'name' part must conform to platform dependent restrictions on environment variable names.

The string given in `var` is copied and optionally converted to the required encoding for the platform. On platforms other than Windows this function calls `putenv` directly.

The return value is zero on success and non-zero in case of error. The value in case of error is platform specific and returned as-is.

Note

The copied string is allocated on the heap and "lost" on some platforms, i.e. calling `fl_putenv()` to change environment variables frequently may cause memory leaks. There may be an option to avoid this in a future implementation.

This function is not thread-safe.

Parameters

in	<i>var</i>	the UTF-8 encoded environment variable 'name=value'

Returns

0 on success, non-zero in case of error.

10.11.3.15 fl_rename()

```
int fl_rename (
    const char * f,
    const char * n)
```

Cross-platform function to rename a filesystem object using UTF-8 encoded names.

This function is especially useful on the Windows platform where the standard `_wrename()` function expects UTF-16 encoded non-ASCII filenames.

Parameters

in	<i>f</i>	the UTF-8 encoded filename to change
in	<i>n</i>	the new UTF-8 encoded filename to set

Returns

the return value of `_wrename()` on Windows or `rename()` on other platforms.

10.11.3.16 fl_rmdir()

```
int fl_rmdir (  
    const char * f)
```

Cross-platform function to remove a directory with a UTF-8 encoded name.

This function is especially useful on the Windows platform where the standard `_wrmdir()` function expects UTF-16 encoded non-ASCII filenames.

Parameters

in	<i>f</i>	the UTF-8 encoded filename to remove

Returns

the return value of `_wrmdir()` on Windows or `rmdir()` on other platforms.

10.11.3.17 fl_stat()

```
int fl_stat (  
    const char * f,  
    struct stat * b)
```

Cross-platform function to `stat()` a file using a UTF-8 encoded name or value.

This function is especially useful on the Windows platform where the standard `stat()` function fails with UTF-8 encoded non-ASCII filenames.

Parameters

in	<i>f</i>	the UTF-8 encoded filename
	<i>b</i>	the stat struct to populate

Returns

the return value of `_wstat()` on Windows or `stat()` on other platforms.

10.11.3.18 fl_system()

```
int fl_system (  
    const char * cmd)
```

Cross-platform function to run a system command with a UTF-8 encoded string.

This function is especially useful on the Windows platform where non-ASCII program (file) names must be encoded as wide characters.

On platforms other than Windows this function calls `system()` directly.

Parameters

in	<i>cmd</i>	the UTF-8 encoded command string

Returns

the return value of `_wsystem()` on Windows or `system()` on other platforms.

10.11.3.19 fl_ucs_to_Utf16()

```
unsigned fl_ucs_to_Utf16 (
    const unsigned ucs,
    unsigned short * dst,
    const unsigned dstlen)
```

Convert a single 32-bit Unicode codepoint into an array of 16-bit characters.

These are used by some system calls, especially on Windows.

`ucs` is the value to convert.

`dst` points at an array to write, and `dstlen` is the number of locations in this array. At most `dstlen` words will be written, and a 0 terminating word will be added if `dstlen` is large enough. Thus this function will never overwrite the buffer and will attempt return a zero-terminated string if space permits. If `dstlen` is zero then `dst` can be set to NULL and no data is written, but the length is returned.

The return value is the number of 16-bit words that *would* be written to `dst` if it is large enough, not counting any terminating zero.

If the return value is greater than `dstlen` it indicates truncation, you should then allocate a new array of size `return+1` and call this again.

Unicode characters in the range 0x10000 to 0x10ffff are converted to "surrogate pairs" which take two words each (in UTF-16 encoding). Typically, setting `dstlen` to 2 will ensure that any valid Unicode value can be converted, and setting `dstlen` to 3 or more will allow a NULL terminated sequence to be returned.

10.11.3.20 fl_unlink()

```
int fl_unlink (
    const char * fname)
```

Cross-platform function to `unlink()` (that is, delete) a file using a UTF-8 encoded filename.

This function is especially useful on the Windows platform where the standard function expects UTF-16 encoded non-ASCII filenames.

Parameters

<i>fname</i>	the filename to unlink

Returns

the return value of `_wunlink()` on Windows or `unlink()` on other platforms.

10.11.3.21 fl_utf8back()

```
const char * fl_utf8back (
    const char * p,
    const char * start,
    const char * end)
```

Move `p` backward until it points to the start of a UTF-8 character.

If it already points at the start of one then it is returned unchanged. Any UTF-8 errors are treated as though each byte of the error is an individual character.

`start` is the start of the string and is used to limit the backwards search for the start of a UTF-8 character.

`end` is the end of the string and is assumed to be a break between characters. It is assumed to be greater than `p`.

If you wish to decrement a UTF-8 pointer, pass `p-1` to this.

10.11.3.22 fl_utf8bytes()

```
int fl_utf8bytes (
    unsigned ucs)
```

Return the number of bytes needed to encode the given UCS4 character in UTF-8.

Returns number of bytes that `utf8encode()` will use to encode the character `ucs`.

Parameters

in	<i>ucs</i>	UCS4 encoded character

Returns

number of bytes required

10.11.3.23 fl_utf8decode()

```
unsigned fl_utf8decode (
    const char * p,
    const char * end,
    int * len)
```

Decode a single UTF-8 encoded character starting at *p*.

The resulting Unicode value (in the range 0-0x10ffff) is returned, and *len* is set to the number of bytes in the UTF-8 encoding (adding *len* to *p* will point at the next character).

If *p* points at an illegal UTF-8 encoding, including one that would go past *end*, or where a code uses more bytes than necessary, then (*unsigned char*)*p* is translated as though it is in the Microsoft CP1252 character set and *len* is set to 1. Treating errors this way allows this to decode almost any ISO-8859-1 or CP1252 text that has been mistakenly placed where UTF-8 is expected, and has proven very useful.

If you want errors to be converted to error characters (as the standards recommend), adding a test to see if the length is unexpectedly 1 will work:

```
if (*p & 0x80) { // what should be a multibyte encoding
    code = fl_utf8decode(p, end, &len);
    if (len < 2) code = 0xFFFD; // Turn errors into REPLACEMENT CHARACTER
} else { // handle the 1-byte UTF-8 encoding:
    code = *p;
    len = 1;
}
```

Direct testing for the 1-byte case (as shown above) will also speed up the scanning of strings where the majority of characters are ASCII.

10.11.3.24 fl_utf8encode()

```
int fl_utf8encode (
    unsigned ucs,
    char * buf)
```

Write the UTF-8 encoding of *ucs* into *buf* and return the number of bytes written.

Up to 4 bytes may be written. If you know that *ucs* is less than 0x10000 then at most 3 bytes will be written. If you wish to speed this up, remember that anything less than 0x80 is written as a single byte.

If *ucs* is greater than 0x10ffff this is an illegal character according to RFC 3629. These are converted as though they are 0xFFFD (REPLACEMENT CHARACTER).

RFC 3629 also says many other values for *ucs* are illegal (in the range 0xd800 to 0xdfff, or ending with 0xfffe or 0xffff). However I encode these as though they are legal, so that *utf8encode/fl_utf8decode* will be the identity for all codes between 0 and 0x10ffff.

10.11.3.25 fl_utf8from_mb()

```
unsigned fl_utf8from_mb (
    char * dst,
    unsigned dstlen,
    const char * src,
    unsigned srclen)
```

Convert a filename from the locale-specific multibyte encoding used by Windows to UTF-8 as used by FLTK.

Up to *dstlen* bytes are written to *dst*, including a null terminator. The return value is the number of bytes that would be written, not counting the null terminator. If greater or equal to *dstlen* then if you malloc a new array of size *n*+1 you will have the space needed for the entire string. If *dstlen* is zero then nothing is written and this call just measures the storage space needed.

On Unix or on Windows when a UTF-8 locale is in effect, this does not change the data. You may also want to check if `fl_utf8test()` returns non-zero, so that the filesystem can store filenames in UTF-8 encoding regardless of the locale.

10.11.3.26 `fl_utf8froma()`

```
unsigned fl_utf8froma (
    char * dst,
    unsigned dstlen,
    const char * src,
    unsigned srclen)
```

Convert an ISO-8859-1 (ie normal c-string) byte stream to UTF-8.

It is possible this should convert Microsoft's CP1252 to UTF-8 instead. This would translate the codes in the range 0x80-0x9f to different characters. Currently it does not do this.

Up to `dstlen` bytes are written to `dst`, including a null terminator. The return value is the number of bytes that would be written, not counting the null terminator. If greater or equal to `dstlen` then if you malloc a new array of size `n+1` you will have the space needed for the entire string. If `dstlen` is zero then nothing is written and this call just measures the storage space needed.

`srclen` is the number of bytes in `src` to convert.

If the return value equals `srclen` then this indicates that no conversion is necessary, as only ASCII characters are in the string.

10.11.3.27 `fl_utf8fromwc()`

```
unsigned fl_utf8fromwc (
    char * dst,
    unsigned dstlen,
    const wchar_t * src,
    unsigned srclen)
```

Turn "wide characters" as returned by some system calls (especially on Windows) into UTF-8.

Up to `dstlen` bytes are written to `dst`, including a null terminator. The return value is the number of bytes that would be written, not counting the null terminator. If greater or equal to `dstlen` then if you malloc a new array of size `n+1` you will have the space needed for the entire string. If `dstlen` is zero then nothing is written and this call just measures the storage space needed.

`srclen` is the number of words in `src` to convert. On Windows this is not necessarily the number of characters, due to there possibly being "surrogate pairs" in the UTF-16 encoding used. On Unix `wchar_t` is 32 bits and each location is a character.

On Unix if a `src` word is greater than 0x10ffff then this is an illegal character according to RFC 3629. These are converted as though they are 0xFFFFD (REPLACEMENT CHARACTER). Characters in the range 0xd800 to 0xdfff, or ending with 0xfffe or 0xffff are also illegal according to RFC 3629. However I encode these as though they are legal, so that `fl_utf8towc` will return the original data.

On Windows "surrogate pairs" are converted to a single character and UTF-8 encoded (as 4 bytes). Mismatched halves of surrogate pairs are converted as though they are individual characters.

10.11.3.28 `fl_utf8fwd()`

```
const char * fl_utf8fwd (
    const char * p,
    const char * start,
    const char * end)
```

Move `p` forward until it points to the start of a UTF-8 character.

If it already points at the start of one then it is returned unchanged. Any UTF-8 errors are treated as though each byte of the error is an individual character.

`start` is the start of the string and is used to limit the backwards search for the start of a UTF-8 character.

`end` is the end of the string and is assumed to be a break between characters. It is assumed to be greater than `p`.

This function is for moving a pointer that was jumped to the middle of a string, such as when doing a binary search for a position. You should use either this or `fl_utf8back()` depending on which direction your algorithm can handle the pointer moving. Do not use this to scan strings, use `fl_utf8decode()` instead.

10.11.3.29 fl_utf8len()

```
int fl_utf8len (
    char c)
```

Returns the byte length of the UTF-8 sequence with first byte `c`, or -1 if `c` is not valid. This function is helpful for finding faulty UTF-8 sequences.

See also

[fl_utf8len1](#)

10.11.3.30 fl_utf8len1()

```
int fl_utf8len1 (
    char c)
```

Returns the byte length of the UTF-8 sequence with first byte `c`, or 1 if `c` is not valid. This function can be used to scan faulty UTF-8 sequences, albeit ignoring invalid codes.

See also

[fl_utf8len](#)

10.11.3.31 fl_utf8locale()

```
int fl_utf8locale (
    void )
```

Return true if the "locale" seems to indicate that UTF-8 encoding is used.

If true the `fl_utf8to_mb` and `fl_utf8from_mb` don't do anything useful.

It is highly recommended that you change your system so this does return true. On Windows this is done by setting the "codepage" to CP_UTF8. On Unix this is done by setting `$LC_CTYPE` to a string containing the letters "utf" or "UTF" in it, or by deleting all `$LC*` and `$LANG` environment variables. In the future it is likely that all non-Asian Unix systems will return true, due to the compatibility of UTF-8 with ISO-8859-1.

10.11.3.32 fl_utf8strlen()

```
int fl_utf8strlen (
    const char * text,
    int len)
```

Return the length in bytes of a UTF-8 string.

Parameters

in	<i>text</i>	encoded in UTF-8
in	<i>len</i>	number of Unicode characters, -1 to test until the end of text

Returns

number of bytes that make up the Unicode string

See also

[fl_utf_nb_char\(const unsigned char *buf, int len\)](#)

10.11.3.33 fl_utf8test()

```
int fl_utf8test (
    const char * src,
    unsigned srclen)
```

Examines the first `srclen` bytes in `src` and returns a verdict on whether it is UTF-8 or not.

- Returns 0 if there is any illegal UTF-8 sequences, using the same rules as [fl_utf8decode\(\)](#). Note that some UCS values considered illegal by RFC 3629, such as 0xffff, are considered legal by this.
- Returns 1 if there are only single-byte characters (ie no bytes have the high bit set). This is legal UTF-8, but also indicates plain ASCII. It also returns 1 if `srclen` is zero.
- Returns 2 if there are only characters less than 0x800.
- Returns 3 if there are only characters less than 0x10000.
- Returns 4 if there are characters in the 0x10000 to 0x10ffff range.

Because there are many illegal sequences in UTF-8, it is almost impossible for a string in another encoding to be confused with UTF-8. This is very useful for transitioning Unix to UTF-8 filenames, you can simply test each filename with this to decide if it is UTF-8 or in the locale encoding. My hope is that if this is done we will be able to cleanly transition to a locale-less encoding.

10.11.3.34 fl_utf8to_mb()

```
unsigned fl_utf8to_mb (
    const char * src,
    unsigned srclen,
    char * dst,
    unsigned dstlen)
```

Convert the UTF-8 used by FLTK to the locale-specific encoding used for filenames (and sometimes used for data in files).

Unfortunately due to stupid design you will have to do this as needed for filenames. This is a bug on both Unix and Windows.

Up to `dstlen` bytes are written to `dst`, including a null terminator. The return value is the number of bytes that would be written, not counting the null terminator. If greater or equal to `dstlen` then if you malloc a new array of size `n+1` you will have the space needed for the entire string. If `dstlen` is zero then nothing is written and this call just measures the storage space needed.

If [fl_utf8locale\(\)](#) returns true then this does not change the data.

10.11.3.35 fl_utf8toa()

```
unsigned fl_utf8toa (
    const char * src,
    unsigned srclen,
    char * dst,
    unsigned dstlen)
```

Convert a UTF-8 sequence into an array of 1-byte characters.

If the UTF-8 decodes to a character greater than 0xff then it is replaced with '?'.

Errors in the UTF-8 sequence are converted as individual bytes, same as [fl_utf8decode\(\)](#) does. This allows ISO-8859-1 text mistakenly identified as UTF-8 to be printed correctly (and possibly CP1252 on Windows).

`src` points at the UTF-8 sequence, and `srclen` is the number of bytes to convert.

Up to `dstlen` bytes are written to `dst`, including a null terminator. The return value is the number of bytes that would be written, not counting the null terminator. If greater or equal to `dstlen` then if you malloc a new array of size `n+1` you will have the space needed for the entire string. If `dstlen` is zero then nothing is written and this call just measures the storage space needed.

10.11.3.36 fl_utf8toUtf16()

```
unsigned fl_utf8toUtf16 (
    const char * src,
    unsigned srclen,
    unsigned short * dst,
    unsigned dstlen)
```

Convert a UTF-8 sequence into an array of 16-bit characters.

These are used by some system calls, especially on Windows.

`src` points at the UTF-8, and `srclen` is the number of bytes to convert.

`dst` points at an array to write, and `dstlen` is the number of locations in this array. At most `dstlen-1` words will be written there, plus a 0 terminating word. Thus this function will never overwrite the buffer and will always return a zero-terminated string. If `dstlen` is zero then `dst` can be null and no data is written, but the length is returned. The return value is the number of 16-bit words that *would* be written to `dst` if it were long enough, not counting the terminating zero. If the return value is greater or equal to `dstlen` it indicates truncation, you can then allocate a new array of size `return+1` and call this again.

Errors in the UTF-8 are converted as though each byte in the erroneous string is in the Microsoft CP1252 encoding. This allows ISO-8859-1 text mistakenly identified as UTF-8 to be printed correctly.

Unicode characters in the range 0x10000 to 0x10ffff are converted to "surrogate pairs" which take two words each (this is called UTF-16 encoding).

10.11.3.37 `fl_utf8towc()`

```
unsigned fl_utf8towc (
    const char * src,
    unsigned srclen,
    wchar_t * dst,
    unsigned dstlen)
```

Converts a UTF-8 string into a wide character string.

This function generates 32-bit `wchar_t` (e.g. "ucs4" as it were) except on Windows where it is equivalent to `fl_utf8toUtf16` and returns UTF-16.

`src` points at the UTF-8, and `srclen` is the number of bytes to convert.

`dst` points at an array to write, and `dstlen` is the number of locations in this array. At most `dstlen-1` `wchar_t` will be written there, plus a 0 terminating `wchar_t`.

The return value is the number of `wchar_t` that *would* be written to `dst` if it were long enough, not counting the terminating zero. If the return value is greater or equal to `dstlen` it indicates truncation, you can then allocate a new array of size `return+1` and call this again.

Notice that `sizeof(wchar_t)` is 2 on Windows and is 4 on Linux and most other systems. Where `wchar_t` is 16 bits, Unicode characters in the range 0x10000 to 0x10ffff are converted to "surrogate pairs" which take two words each (this is called UTF-16 encoding). If `wchar_t` is 32 bits this rather nasty problem is avoided.

Note that Windows includes Cygwin, i.e. compiled with Cygwin's POSIX layer (`cygwin1.dll`, `-enable-cygwin`), either native (GDI) or X11.

10.11.3.38 `fl_utf_nb_char()`

```
int fl_utf_nb_char (
    const unsigned char * buf,
    int len)
```

Returns the number of Unicode chars in the UTF-8 string.

See also

[fl_utf8strlen\(const char *text, int len\)](#)

10.11.3.39 `fl_utf_strcasecmp()`

```
int fl_utf_strcasecmp (
    const char * s1,
    const char * s2)
```

UTF-8 aware `strcasecmp` - converts to Unicode and tests.

Returns

result of comparison

Return values

0	if the strings are equal

1	if s1 is greater than s2
-1	if s1 is less than s2

10.11.3.40 fl_utf_strncasecmp()

```
int fl_utf_strncasecmp (
    const char * s1,
    const char * s2,
    int n)
```

UTF-8 aware strncasecmp - converts to lower case Unicode and tests.

Parameters

<i>s1,s2</i>	the UTF-8 strings to compare
<i>n</i>	the maximum number of UTF-8 characters to compare

Returns

result of comparison

Return values

0	if the strings are equal
>0	if s1 is greater than s2
<0	if s1 is less than s2

10.11.3.41 fl_utf_tolower()

```
int fl_utf_tolower (
    const unsigned char * str,
    int len,
    char * buf)
```

Converts the string `str` to its lower case equivalent into `buf`.

Warning: to be safe `buf` length must be at least `3 * len` [for 16-bit Unicode]

10.11.3.42 fl_utf_toupper()

```
int fl_utf_toupper (
    const unsigned char * str,
    int len,
    char * buf)
```

Converts the string `str` to its upper case equivalent into `buf`.

Warning: to be safe `buf` length must be at least `3 * len` [for 16-bit Unicode]

10.11.3.43 fl_wcwidth()

```
int fl_wcwidth (
    const char * src)
```

extended wrapper around [fl_wcwidth_\(unsigned int ucs\)](#) function.

Parameters

<i>in</i>	<i>src</i>	pointer to start of UTF-8 byte sequence

Returns

width of character in columns

Depending on build options, this function may map C1 control characters (0x80 to 0x9f) to CP1252, and return the width of that character instead. This is not the same behaviour as `fl_wcwidth_(unsigned int ucs)`.

Note that other control characters and DEL will still return -1, so if you want different behaviour, you need to test for those characters before calling `fl_wcwidth()`, and handle them separately.

10.11.3.44 fl_wcwidth_()

```
int fl_wcwidth_ (
    unsigned int ucs)
```

Wrapper to adapt Markus Kuhn's implementation of `wcwidth()` for FLTK.

Parameters

in	ucs	Unicode character value

Returns

width of character in columns

See <http://www.cl.cam.ac.uk/~mgk25/ucs/wcwidth.c> for Markus Kuhn's original implementation of `wcwidth()` and `wcswidth()` (defined in IEEE Std 1002.1-2001) for Unicode.

WARNING: this function returns widths for "raw" Unicode characters. It does not even try to map C1 control characters (0x80 to 0x9F) to CP1252, and C0/C1 control characters and DEL will return -1. You are advised to use `fl_width(const char* src)` instead.

10.12 String handling functions

String handling functions declared in `<FL/fl_string_functions.h>`

Functions

- `char * fl_strdup (const char *s)`
Cross platform interface to POSIX function `strdup()`.
- `size_t fl_strlcpy (char *, const char *, size_t)`

10.12.1 Detailed Description

String handling functions declared in `<FL/fl_string_functions.h>`

10.12.2 Function Documentation**10.12.2.1 fl_strdup()**

```
char * fl_strdup (
    const char * s)
```

Cross platform interface to POSIX function `strdup()`.

The `fl_strdup()` function returns a pointer to a new string which is a duplicate of the string 's'. Memory for the new string is obtained with `malloc(3)`, and can be freed with `free(3)`.

Implementation:

- POSIX: `strdup()`
- WinAPI: `_strdup()`

10.13 Mac OS X-specific symbols

Mac OS X-specific symbols declared in `<FL/platform.H>`

Classes

- class `Fl_Mac_App_Menu`

Functions

- `Fl_Window * fl_mac_find (FLWindow *)`
Returns the `Fl_Window` corresponding to the given macOS-specific window reference.
- `CGContextRef fl_mac_gc ()`
Returns the macOS-specific graphics context for the current window.
- `void fl_mac_set_about (Fl_Callback *cb, void *user_data, int shortcut=0)`
Attaches a callback to the "About myprog" item of the system application menu.
- `FLWindow * fl_mac_xid (const Fl_Window *win)`
Returns the macOS-specific window reference corresponding to the given `Fl_Window` object.
- `void fl_open_callback (void(*cb)(const char *))`
Register a function called for each file dropped onto an application icon.

Variables

- `int fl_mac_os_version`
The version number of the running Mac OS X (e.g., 100604 for 10.6.4, 101300 for 10.13, 140102 for 14.1.2).

10.13.1 Detailed Description

Mac OS X-specific symbols declared in `<FL/platform.H>`

See also

[The Apple OS X Interface](#)

10.13.2 Function Documentation

10.13.2.1 fl_mac_set_about()

```
void fl_mac_set_about (
    Fl_Callback * cb,
    void * user_data,
    int shortcut = 0)
```

Attaches a callback to the "About myprog" item of the system application menu.

For back-compatibility. Equivalent to `Fl_Sys_Menu_Bar::about(Fl_Callback *cb, void *user_data)`.

10.13.2.2 fl_open_callback()

```
void fl_open_callback (
    void(* cb )(const char *))
```

Register a function called for each file dropped onto an application icon.

This function is effective only on the Mac OS X platform. `cb` will be called with a single Unix-style file name and path. If multiple files were dropped, `cb` will be called multiple times.

This function should be called before `fl_open_display()` is called, either directly or indirectly (this happens at the first `show()` of a window), to be effective for files dropped on the application icon at launch time. It can also be called at any point to change the function used to open dropped files. A call with a NULL argument, after a previous call, makes the app ignore files dropped later.

10.13.3 Variable Documentation

10.13.3.1 fl_mac_os_version

```
int fl_mac_os_version [extern]
```

The version number of the running Mac OS X (e.g., 100604 for 10.6.4, 101300 for 10.13, 140102 for 14.1.2).
FLTK initializes this global variable before main() begins running. If the value is needed in a static initializer, a previous call to [Fl::system_driver\(\)](#) makes sure [fl_mac_os_version](#) has been initialized.

10.14 Common Dialog Classes and Functions

Files

- file [fl_ask.cxx](#)
Utility functions for common dialogs.

Classes

- class [Fl_Color_Chooser](#)
The [Fl_Color_Chooser](#) widget provides a standard RGB color chooser.
- class [Fl_File_Chooser](#)
The [Fl_File_Chooser](#) widget displays a standard file selection dialog that supports various selection modes.

Functions

- void [fl_alert](#) (const char *fmt,...)
Shows an alert message dialog box.
- int [fl_ask](#) (const char *fmt,...)
*Shows a dialog displaying the *fmt* message, this dialog features 2 yes/no buttons.*
- void [fl_beep](#) (int type)
Emits a system beep.
- int [fl_choice](#) (const char *fmt, const char *b0, const char *b1, const char *b2,...)
*Shows a dialog displaying the printf style *fmt* message.*
- int [fl_choice_n](#) (const char *fmt, const char *b0, const char *b1, const char *b2,...)
*Shows a dialog displaying the printf style *fmt* message.*
- int [fl_color_chooser](#) (const char *name, double &r, double &g, double &b, int cmode)
Pops up a window to let the user pick an arbitrary RGB color.
- int [fl_color_chooser](#) (const char *name, uchar &r, uchar &g, uchar &b, int cmode)
Pops up a window to let the user pick an arbitrary RGB color.
- char * [fl_dir_chooser](#) (const char *message, const char *fname, int relative)
Shows a file chooser dialog and gets a directory.
- char * [fl_file_chooser](#) (const char *message, const char *pat, const char *fname, int relative)
Shows a file chooser dialog and gets a filename.
- void [fl_file_chooser_callback](#) (void(*cb)(const char *))
Set the file chooser callback.
- void [fl_file_chooser_ok_label](#) (const char *l)
Set the "OK" button label.
- const char * [fl_input](#) (const char *fmt, const char *defstr,...)
*Shows an input dialog displaying the *fmt* message with variable arguments.*
- const char * [fl_input](#) (int maxchar, const char *fmt, const char *defstr,...)
*Shows an input dialog displaying the *fmt* message with variable arguments.*
- void [fl_message](#) (const char *fmt,...)
Shows an information message dialog box.
- int [fl_message_hotspot](#) ()

- Gets whether or not to move the message box used in many common dialogs like `fl_message()`, `fl_alert()`, `fl_ask()`, `fl_choice()`, `fl_input()`, `fl_password()` to follow the mouse pointer.*
- void `fl_message_hotspot` (int enable)
 - Sets whether or not to move the message box used in many common dialogs like `fl_message()`, `fl_alert()`, `fl_ask()`, `fl_choice()`, `fl_input()`, `fl_password()` to follow the mouse pointer.*
- `Fl_Widget * fl_message_icon` ()
 - Gets the `Fl_Box` icon container of the current default dialog used in many common dialogs like `fl_message()`, `fl_alert()`, `fl_ask()`, `fl_choice()`, `fl_input()`, `fl_password()`.*
- void `fl_message_icon_label` (const char *str)
 - Sets the icon label of the dialog window used in many common dialogs.*
- void `fl_message_position` (const int x, const int y, const int center)
 - Sets the preferred position for the message box used in many common dialogs like `fl_message()`, `fl_alert()`, `fl_ask()`, `fl_choice()`, `fl_input()`, `fl_password()`.*
- void `fl_message_position` (`Fl_Widget *widget`)
 - Sets the preferred position for the message box used in many common dialogs like `fl_message()`, `fl_alert()`, `fl_ask()`, `fl_choice()`, `fl_input()`, `fl_password()`.*
- int `fl_message_position` (int *x, int *y)
 - Gets the preferred position for the message box used in many common dialogs like `fl_message()`, `fl_alert()`, `fl_ask()`, `fl_choice()`, `fl_input()`, `fl_password()`.*
- void `fl_message_title` (const char *title)
 - Sets the title of the dialog window used in many common dialogs.*
- void `fl_message_title_default` (const char *title)
 - Sets the default title of the dialog window used in many common dialogs.*
- const char * `fl_password` (const char *fmt, const char *defstr,...)
 - Shows an input dialog displaying the `fmt` message with variable arguments.*
- const char * `fl_password` (int maxchar, const char *fmt, const char *defstr,...)
 - Shows an input dialog displaying the `fmt` message with variable arguments.*

Variables

- static void(* `Fl::error`)(const char *,...) = `Fl_System_Driver::error`
 - FLTK calls `Fl::error()` to output a normal error message.*
- static void(* `Fl::fatal`)(const char *,...) = `Fl_System_Driver::fatal`
 - FLTK calls `Fl::fatal()` to output a fatal error message.*
- const char * `fl_cancel` = "Cancel"
 - string pointer used in common dialogs, you can change it to another language*
- const char * `fl_close` = "Close"
 - string pointer used in common dialogs, you can change it to another language*
- `Fl_Font fl_message_font` = `FL_HELVETICA`
- `Fl_Fontsize fl_message_size` = -1
- const char * `fl_no` = "No"
 - string pointer used in common dialogs, you can change it to another language*
- const char * `fl_ok` = "OK"
 - string pointer used in common dialogs, you can change it to another language*
- const char * `fl_yes` = "Yes"
 - string pointer used in common dialogs, you can change it to another language*
- static void(* `Fl::warning`)(const char *,...) = `Fl_System_Driver::warning`
 - FLTK calls `Fl::warning()` to output a warning message.*

10.14.1 Detailed Description

Common dialog functions for file selection, message output, and more.

10.14.2 Function Documentation

10.14.2.1 fl_alert()

```
void fl_alert (
    const char * fmt,
    ...)
```

Shows an alert message dialog box.

```
#include <FL/fl_ask.H>
```

Parameters

in	<i>fmt</i>	can be used as an sprintf-like format and variables for the message text

10.14.2.2 fl_ask()

```
int fl_ask (
    const char * fmt,
    ...)
```

Shows a dialog displaying the *fmt* message, this dialog features 2 yes/no buttons.

```
#include <FL/fl_ask.H>
```

Parameters

in	<i>fmt</i>	can be used as an sprintf-like format and variables for the message text

Return values

0	if the no button is selected
1	if yes is selected

Deprecated `fl_ask()` is deprecated since it uses "Yes" and "No" for the buttons which does not conform to the current FLTK Human Interface Guidelines. Use `fl_choice()` with the appropriate verbs instead.

10.14.2.3 fl_beep()

```
void fl_beep (
    int type)
```

Emits a system beep.

This function is platform specific. Depending on the input *type* a different sound may be played or the system speaker may beep with a different volume.

On X the system speaker is used which may not work at all on newer systems that don't have a speaker. Since 1.4.0 `FL_BEEP_DEFAULT` and other types honor the system or user settings whereas `FL_BEEP_ERROR` uses 100% volume. This may be changed in a future version.

On Wayland an ASCII BEL (0x07) is output to stderr.

On Windows the `MessageBeep()` function is used to play different sounds depending on the *type* argument.

On macOS the system beep function `NSBeep()` is used for `FL_BEEP_DEFAULT` and `FL_BEEP_ERROR`. Other types are ignored.

On other platforms the behavior is undefined and may change in the future.

Parameters

in	<i>type</i>	The beep type from the <code>FL_Beep</code> enumeration (optional)

```
#include <FL/fl_ask.H>
```

10.14.2.4 fl_choice()

```
int fl_choice (
    const char * fmt,
    const char * b0,
    const char * b1,
    const char * b2,
    ...)
```

Shows a dialog displaying the printf style `fmt` message.

This dialog features up to 3 customizable choice buttons which are specified in order of *right-to-left* in the dialog, e.g.

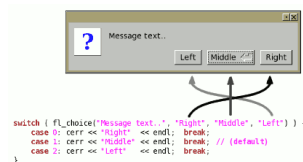


Figure 10.7 `fl_choice()` button ordering

```
#include <FL/fl_ask.H>
```

Three choices with `printf()` style formatting:

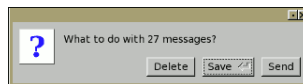


Figure 10.8 `fl_choice()` three choices with `printf` formatting

```
int num_msgs = GetNumberOfMessages();
switch ( fl_choice("What to do with %d messages?", "Send", "Save", "Delete", num_msgs) ) {
    case 0: .. // Send
    case 1: .. // Save (default)
    case 2: .. // Delete
    ..
}
```

Three choice example:

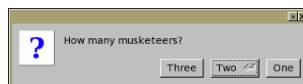


Figure 10.9 `fl_choice()` three choices

```
switch ( fl_choice("How many bedrooms?", "Zero", "One", "Two") ) {
    case 0: .. // "Zero"
    case 1: .. // "One" (default)
    case 2: .. // "Two"
}
```

Two choice example:

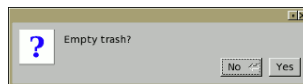


Figure 10.10 `fl_choice()` two choices

```
switch ( fl_choice("Empty trash?", "Yes", "No", 0) ) {
    case 0: .. // Yes
    case 1: .. // No (default)
}
```

One choice example:

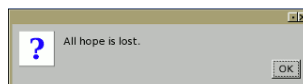


Figure 10.11 `fl_choice()` one choice

```
fl_choice("All hope is lost.", "OK", 0, 0); // "OK" default
```

Parameters

in	<i>fmt</i>	can be used as an sprintf-like format and variables for the message text
in	<i>b0</i>	text label for right button 0
in	<i>b1</i>	text label for middle button 1 (can be 0)
in	<i>b2</i>	text label for left button 2 (can be 0)

Return values

0	if the button with <i>b0</i> text is pushed or the user pressed the <code>Escape</code> key or clicked the window close button
1	if the button with <i>b1</i> text is pushed or the user pressed the <code>Return</code> key
2	if the button with <i>b2</i> text is pushed

10.14.2.5 fl_choice_n()

```
int fl_choice_n (
    const char * fmt,
    const char * b0,
    const char * b1,
    const char * b2,
    ...)
```

Shows a dialog displaying the printf style *fmt* message.

This function is like [fl_choice\(\)](#) but returns `-1` if the dialog window was closed by pressing the `Escape` key or the window close button rather than pushing one of the dialog buttons.

See also

[fl_choice\(\)](#)

Parameters

in	<i>fmt</i>	can be used as an sprintf-like format and variables for the message text
in	<i>b0</i>	text label for right button 0
in	<i>b1</i>	text label for middle button 1 (can be 0)
in	<i>b2</i>	text label for left button 2 (can be 0)

Return values

-3	reserved, FLTK 1.3 only: another dialog is still open (not possible in 1.4)
-2	if the dialog was closed by pushing the window close button
-1	if the dialog was closed by hitting <code>Escape</code>
0	if the button with <i>b0</i> text is pushed
1	if the button with <i>b1</i> text is pushed
2	if the button with <i>b2</i> text is pushed

10.14.2.6 fl_color_chooser() [1/2]

```
int fl_color_chooser (
    const char * name,
    double & r,
    double & g,
    double & b,
    int cmode) [related]
```

Pops up a window to let the user pick an arbitrary RGB color.

Note

```
#include <FL/Fl_Color_Chooser.H>
```

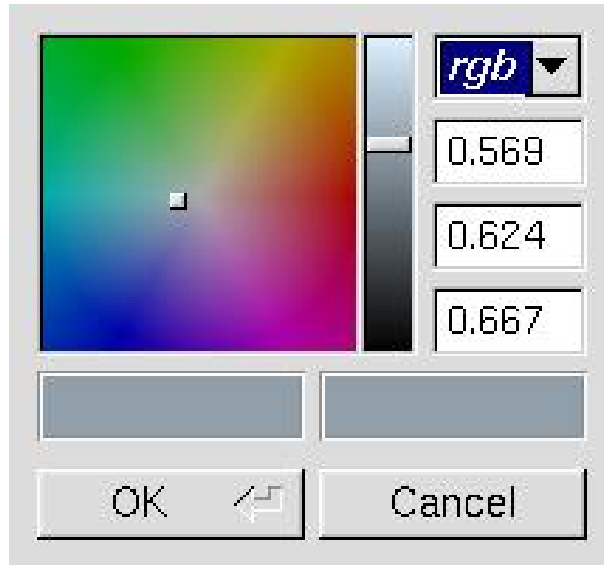


Figure 10.12 fl_color_chooser

Parameters

in	<i>name</i>	Title label for the window
in, out	<i>r,g,b</i>	Color components in the range 0.0 to 1.0.
in	<i>cmode</i>	Optional mode for color chooser. See mode(int) . Default -1 if none (rgb mode).

Return values

1	if user confirms the selection
0	if user cancels the dialog

10.14.2.7 fl_color_chooser() [2/2]

```
int fl_color_chooser (
    const char * name,
    uchar & r,
    uchar & g,
    uchar & b,
    int cmode) [related]
```

Pops up a window to let the user pick an arbitrary RGB color.

Note

```
#include <FL/Fl_Color_Chooser.H>
```

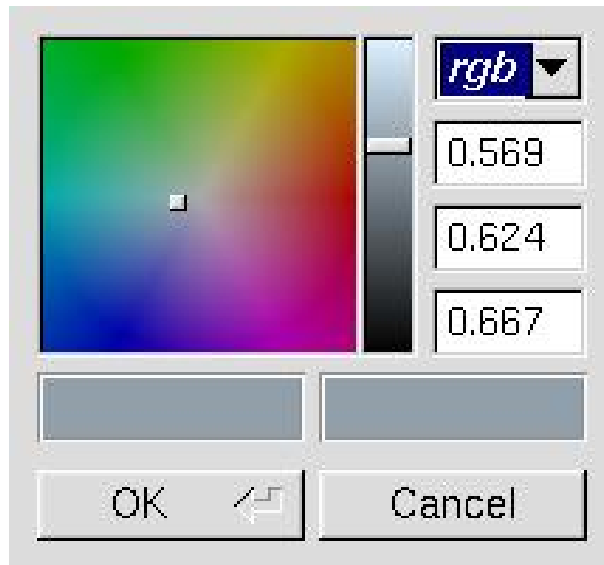


Figure 10.13 fl_color_chooser

Parameters

in	<i>name</i>	Title label for the window
in, out	<i>r,g,b</i>	Color components in the range 0 to 255.
in	<i>cmode</i>	Optional mode for color chooser. See mode(int) . Default -1 if none (rgb mode).

Return values

1	if user confirms the selection
0	if user cancels the dialog

10.14.2.8 fl_dir_chooser()

```
char * fl_dir_chooser (
    const char * message,
    const char * fname,
    int relative) [related]
```

Shows a file chooser dialog and gets a directory.

Note

```
#include <FL/Fl_File_Chooser.H>
```

Parameters

in	<i>message</i>	title bar text
in	<i>fname</i>	initial/default directory name
in	<i>relative</i>	0 for absolute path return, relative otherwise

Returns

the directory path string chosen by the user or NULL if user cancels

10.14.2.9 fl_file_chooser()

```
char * fl_file_chooser (
    const char * message,
    const char * pat,
    const char * fname,
    int relative) [related]
```

Shows a file chooser dialog and gets a filename.

Note

```
#include <FL/Fl_File_Chooser.H>
```

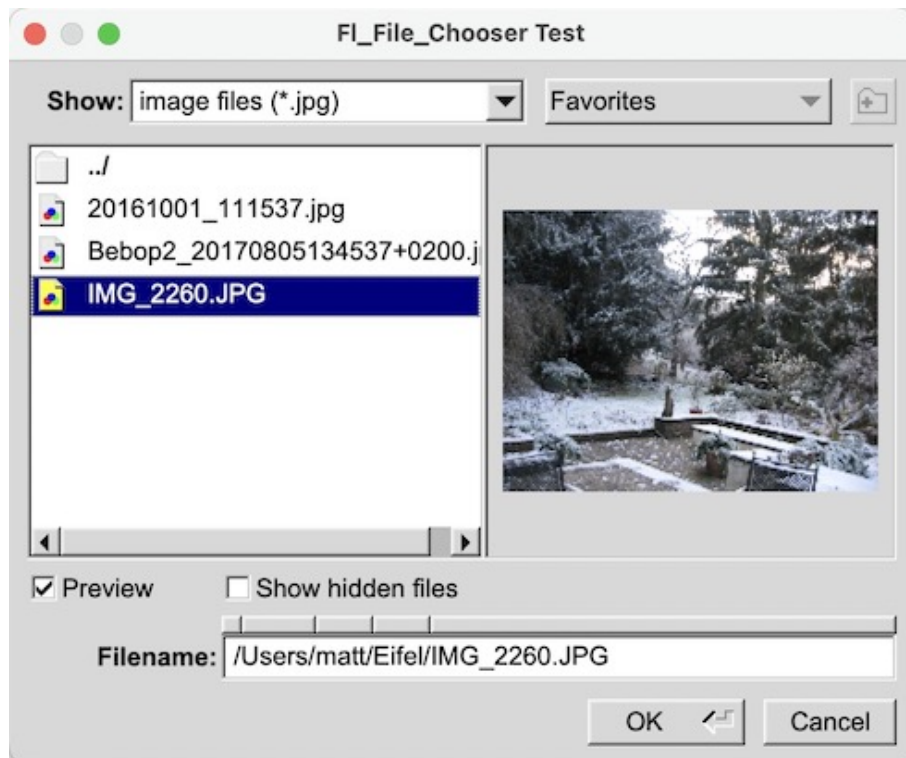


Figure 10.14 Fl_File_Chooser

Parameters

in	<i>message</i>	text in title bar
in	<i>pat</i>	filename pattern filter
in	<i>fname</i>	initial/default filename selection
in	<i>relative</i>	0 for absolute path name, relative path name otherwise

Returns

the user selected filename, in absolute or relative format or NULL if user cancels

10.14.2.10 fl_file_chooser_callback()

```
void fl_file_chooser_callback (
    void(* cb ) (const char *)) [related]
```

Set the file chooser callback.

Note

```
#include <FL/Fl_File_Chooser.H>
```

10.14.2.11 fl_file_chooser_ok_label()

```
void fl_file_chooser_ok_label (
    const char * l) [related]
```

Set the "OK" button label.

Note

```
#include <FL/Fl_File_Chooser.H>
```

10.14.2.12 fl_input() [1/2]

```
const char * fl_input (
    const char * fmt,
    const char * defstr,
    ...)
```

Shows an input dialog displaying the `fmt` message with variable arguments.

Returns the string in an internally allocated buffer that may be changed later. You **must** copy the string immediately after return from this method - at least before the next execution of the event loop.

```
#include <FL/fl_ask.H>
```

Parameters

in	<i>fmt</i>	can be used as an sprintf-like format and variables for the message text
in	<i>defstr</i>	defines the default returned string if no text is entered

Returns

the user string input if OK was pushed

Return values

<i>NULL</i>	if Cancel was pushed or the window was closed by the user

10.14.2.13 fl_input() [2/2]

```
const char * fl_input (
    int maxchar,
    const char * fmt,
    const char * defstr,
    ...)
```

Shows an input dialog displaying the `fmt` message with variable arguments.

This is the same as `const char *fl_input(const char *fmt, const char *defstr, ...)` except that it has an additional parameter to limit the number of characters the user can input.

Returns the string in an internally allocated buffer that may be changed later. You **must** copy the string immediately after return from this method - at least before the next execution of the event loop.

```
#include <FL/fl_ask.H>
```

Parameters

in	<i>maxchar</i>	maximum number of characters the user can input (UTF-8 aware)
in	<i>fmt</i>	can be used as an sprintf-like format and variables for the message text
in	<i>defstr</i>	defines the default returned string if no text is entered

Returns

the user string input if OK was pushed

Return values

<i>NULL</i>	if Cancel was pushed or the window was closed by the user

10.14.2.14 fl_message()

```
void fl_message (
    const char * fmt,
    ...)
```

Shows an information message dialog box.

```
#include <FL/fl_ask.H>
```

Parameters

<i>in</i>	<i>fmt</i>	can be used as an sprintf-like format and variables for the message text

10.14.2.15 fl_message_hotspot() [1/2]

```
int fl_message_hotspot (
    void )
```

Gets whether or not to move the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#) to follow the mouse pointer.

This is a permanent setting. It remains active and affects the window position unless overridden by an explicit positioning request by means of one of the [fl_message_position\(\)](#) variants.

```
#include <FL/fl_ask.H>
```

Returns

0 if disabled, non-zero otherwise

See also

```
void fl_message_hotspot(int)
int fl_message_position(int *x, int *y)
void fl_message_position(Fl_Widget *)
fl_message_position()
```

10.14.2.16 fl_message_hotspot() [2/2]

```
void fl_message_hotspot (
    int enable)
```

Sets whether or not to move the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#) to follow the mouse pointer.

The default is *enabled*, so that the default button is the hotspot and appears at the mouse position.

```
#include <FL/fl_ask.H>
```

Parameters

<i>in</i>	<i>enable</i>	non-zero enables hotspot behavior, 0 disables hotspot

10.14.2.17 fl_message_icon()

```
Fl_Widget * fl_message_icon ()
```

Gets the [Fl_Box](#) icon container of the current default dialog used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).

The return value cannot be Null. The object pointed to is an [Fl_Box](#) widget. The returned pointer ([Fl_Widget *](#)) can be safely cast to an [Fl_Box*](#) pointer.

Note

You can set some attributes of this **default** icon box. These attributes are sticky, i.e. they will be used in all subsequent common dialogs unless overridden by specific "one shot" variables. Setting any attribute except those mentioned below causes undefined behavior.

Supported icon attributes:

- [box\(\)](#)
- [labelfont\(\)](#)
- [labelsize\(\)](#)
- [color\(\)](#)
- [labelcolor\(\)](#)
- [image\(\)](#)
- [align\(\)](#)

The icon size can not be changed. If you set an [image\(\)](#) you should scale it to the available size, i.e. [w\(\)](#) and [h\(\)](#) of the icon box.

```
#include <FL/fl_ask.H>
```

10.14.2.18 fl_message_icon_label()

```
void fl_message_icon_label (
    const char * str)
```

Sets the icon label of the dialog window used in many common dialogs.

This icon label will be used in the next call of one of the common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).

The label `str` is stored internally as a reference, it must be in scope until the dialog function (e.g. [fl_choice\(\)](#)) is called.

It applies only to the **next** call of one of the common dialogs and will be reset after that call so the next dialog will use its default label unless set again.

Note

This label string must be short, usually only one character so it fits in the icon box. You can use any valid UTF-8 character, e.g. the Euro sign ("€") which is three bytes in UTF-8 encoding.

```
#include <FL/fl_ask.H>
```

Parameters

<code>in</code>	<code>str</code>	icon label

10.14.2.19 fl_message_position() [1/3]

```
void fl_message_position (
    const int x,
    const int y,
    const int center)
```

Sets the preferred position for the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).

The position set with this method overrides the hotspot setting, i.e. setting a position has higher priority than the hotspot mode set by [fl_message_hotspot\(int\)](#).

The preferred position set by any of the [fl_message_position\(\)](#) variants affects only the next call of one of the common dialogs. The preferred position is reset to 0 (unset) as soon as the dialog is shown.

If the optional argument `center` is non-zero (true) the message box will be centered at the given coordinates rather than using the X/Y position as the window position (top left corner).

```
#include <FL/fl_ask.H>
```

Parameters

in	<i>x</i>	Preferred X position
in	<i>y</i>	Preferred Y position
in	<i>center</i>	1 = centered, 0 = absolute

See also

```
int fl_message_position(int *x, int *y)
```

10.14.2.20 fl_message_position() [2/3]

```
void fl_message_position (
    Fl_Widget * widget)
```

Sets the preferred position for the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).

The message box will be centered over the given widget or window extensions.

Everything else is like [fl_message_position\(int, int, int\)](#) with argument 'center' set to 1.

```
#include <FL/fl_ask.H>
```

Parameters

in	<i>widget</i>	Widget or window to position the message box over.

See also

```
int fl_message_position(int x, int y, int center)
```

10.14.2.21 fl_message_position() [3/3]

```
int fl_message_position (
    int * x,
    int * y)
```

Gets the preferred position for the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).

```
#include <FL/fl_ask.H>
```

The position set with this method overrides the hotspot setting, i.e. setting a position has higher priority than the hotspot mode set by [fl_message_hotspot\(int\)](#).

The preferred position set by any of the [fl_message_position\(\)](#) variants affects only the next call of one of the common dialogs. The preferred position is reset to 0 (unset) as soon as the dialog is shown.

Parameters

out	<i>x</i>	Preferred X position, returns -1 if not set
out	<i>y</i>	Preferred Y position, returns -1 if not set

Returns

whether position is currently set or not

Return values

0	position is not set (hotspot may be enabled or not)
1	position is set (window position)
2	position is set (message box centered)

See also

[fl_message_hotspot\(\)](#)
[fl_message_hotspot\(int\)](#)
[fl_message_position\(int, int\)](#)
[fl_message_position\(const int x, const int y, const int center\)](#)
[fl_message_position\(Fl_Widget *\)](#)

10.14.2.22 fl_message_title()

```
void fl_message_title (
    const char * title)
```

Sets the title of the dialog window used in many common dialogs.

This window `title` will be used in the next call of one of the common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).

The `title` string is copied internally, so that you can use a local variable or free the string immediately after this call. It applies only to the **next** call of one of the common dialogs and will be reset to an empty title (the default for all dialogs) after that call.

```
#include <FL/fl_ask.H>
```

Parameters

in	<i>title</i>	window label, string copied internally

10.14.2.23 fl_message_title_default()

```
void fl_message_title_default (
    const char * title)
```

Sets the default title of the dialog window used in many common dialogs.

This window `title` will be used in all subsequent calls of one of the common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#), unless a specific title has been set with [fl_message_title\(const char *title\)](#).

The default is no title. You can override the default title for a single dialog with [fl_message_title\(const char *title\)](#).

The `title` string is copied internally, so that you can use a local variable or free the string immediately after this call.

```
#include <FL/fl_ask.H>
```

Parameters

in	<i>title</i>	default window label, string copied internally

10.14.2.24 fl_password() [1/2]

```
const char * fl_password (
    const char * fmt,
    const char * defstr,
    ...)
```

Shows an input dialog displaying the `fmt` message with variable arguments.

Like [fl_input\(\)](#) except the input text is not shown, '*' or similar replacement characters are displayed instead.

```
#include <FL/fl_ask.H>
```


Parameters

in	<i>fmt</i>	can be used as an sprintf-like format and variables for the message text
in	<i>defstr</i>	defines the default returned string if no text is entered

Returns

the user string input if OK was pushed

Return values

<i>NULL</i>	if Cancel was pushed or the window was closed by the user

10.14.2.25 `fl_password()` [2/2]

```
const char * fl_password (
    int maxchar,
    const char * fmt,
    const char * defstr,
    ...)
```

Shows an input dialog displaying the *fmt* message with variable arguments.

Like `fl_input()` except the input text is not shown, '*' or similar replacement characters are displayed instead.

```
#include <FL/fl_ask.H>
```

Parameters

in	<i>maxchar</i>	input lenght limit in chars, 0 = no limit
in	<i>fmt</i>	can be used as an sprintf-like format and variables for the message text
in	<i>defstr</i>	defines the default returned string if no text is entered

Returns

the user string input if OK was pushed

Return values

<i>NULL</i>	if Cancel was pushed or the window was closed by the user

10.14.3 Variable Documentation

10.14.3.1 `error`

```
void(* Fl::error)(const char *format,...) = Fl_System_Driver::error [static]
```

FLTK calls `Fl::error()` to output a normal error message.

The default version on Windows displays the error message in a MessageBox window.

The default version on all other platforms prints the error message to stderr.

You can override the behavior by setting the function pointer to your own routine.

`Fl::error()` means there is a recoverable error such as the inability to read an image file. The default implementation returns after displaying the message.

Note

```
#include <FL/Fl.H>
```

10.14.3.2 fatal

```
void(* Fl::fatal)(const char *format,...) = Fl_System_Driver::fatal [static]
```

FLTK calls [Fl::fatal\(\)](#) to output a fatal error message.

The default version on Windows displays the error message in a MessageBox window.

The default version on all other platforms prints the error message to stderr.

You can override the behavior by setting the function pointer to your own routine.

[Fl::fatal\(\)](#) must not return, as FLTK is in an unusable state, however your version may be able to use longjmp or an exception to continue, as long as it does not call FLTK again. The default implementation exits with status 1 after displaying the message.

Note

```
#include <FL/Fl.H>
```

10.14.3.3 warning

```
void(* Fl::warning)(const char *format,...) = Fl_System_Driver::warning [static]
```

FLTK calls [Fl::warning\(\)](#) to output a warning message.

The default version on Windows returns *without* printing a warning message, because Windows programs normally don't have stderr (a console window) enabled.

The default version on all other platforms prints the warning message to stderr.

You can override the behavior by setting the function pointer to your own routine.

[Fl::warning\(\)](#) means that there was a recoverable problem, the display may be messed up, but the user can probably keep working - all X protocol errors call this, for example. The default implementation returns after displaying the message.

Note

```
#include <FL/Fl.H>
```

10.15 File names and URI utility functions

File names and URI functions defined in [<FL/filename.H>](#)

Macros

- `#define FL_PATH_MAX 2048`
all path buffers should use this length

Typedefs

- typedef int [Fl_File_Sort_F](#)(struct dirent **, struct dirent **)
File sorting function.

Functions

- void [fl_decode_uri](#) (char *uri)
Decodes a URL-encoded string.
- int [fl_filename_absolute](#) (char *to, int tolen, const char *from)
Makes a filename absolute from a relative filename to the current working directory.
- int [fl_filename_absolute](#) (char *to, int tolen, const char *from, const char *cwd)
Concatenate the absolute path base with from to form the new absolute path in to.
- std::string [fl_filename_absolute_str](#) (const std::string &from)
Makes a filename absolute from a filename relative to the current working directory.
- std::string [fl_filename_absolute_str](#) (const std::string &from, const std::string &base)
Append the relative filename from to the absolute filename base to form the new absolute path.
- int [fl_filename_expand](#) (char *to, int tolen, const char *from)

- Expands a filename containing shell variables and tilde (~).*
- `std::string fl_filename_expand_str` (const std::string &from)
- Expands a filename containing shell variables and tilde (~).*
- `const char * fl_filename_ext` (const char *buf)
- Gets the extension of a filename.*
- `std::string fl_filename_ext_str` (const std::string &filename)
- Return a new string that contains the filename extension.*
- `void fl_filename_free_list` (struct dirent **l, int n)
- Free the list of filenames that is generated by `fl_filename_list()`.*
- `int fl_filename_isdir` (const char *name)
- Determines if a file exists and is a directory from its filename.*
- `int fl_filename_list` (const char *d, struct dirent **l, `FI_File_Sort_F` *s=`fl_numericssort`)
- Portable and const-correct wrapper for the `scandir()` function.*
- `int fl_filename_match` (const char *name, const char *pattern)
- Checks if a string `s` matches a pattern `p`.*
- `const char * fl_filename_name` (const char *filename)
- Gets the file name from a path.*
- `std::string fl_filename_name_str` (const std::string &filename)
- Return a new string that contains the name part of the filename.*
- `std::string fl_filename_path_str` (const std::string &filename)
- Return a new string that contains the path part of the filename.*
- `int fl_filename_relative` (char *to, int tolen, const char *from)
- Makes a filename relative to the current working directory.*
- `int fl_filename_relative` (char *to, int tolen, const char *from, const char *cwd)
- Makes a filename relative to any other directory.*
- `std::string fl_filename_relative_str` (const std::string &from)
- Makes a filename relative to the current working directory.*
- `std::string fl_filename_relative_str` (const std::string &from, const std::string &base)
- Makes a filename relative to any directory.*
- `char * fl_filename_setext` (char *to, int tolen, const char *ext)
- Replaces the extension in `buf` of `max`.*
- `std::string fl_filename_setext_str` (const std::string &filename, const std::string &new_extension)
- Return a copy of the old filename with the new extension.*
- `std::string fl_getcwd_str` ()
- Cross-platform function to get the current working directory as a UTF-8 encoded value in an `std::string`.*
- `int fl_open_uri` (const char *uri, char *msg, int msglen)
- Opens the specified Uniform Resource Identifier (URI).*

10.15.1 Detailed Description

File names and URI functions defined in [<FL/filename.H>](#)

10.15.2 Typedef Documentation

10.15.2.1 FI_File_Sort_F

```
typedef int FI_File_Sort_F(struct dirent **, struct dirent **)
File sorting function.
```

See also

[fl_filename_list\(\)](#)

10.15.3 Function Documentation

10.15.3.1 fl_decode_uri()

```
void fl_decode_uri (
    char * uri)
```

Decodes a URL-encoded string.

In a Uniform Resource Identifier (URI), all non-ASCII bytes and several others (e.g., '<', '"', ''') are URL-encoded using 3 bytes by "%XY" where XY is the hexadecimal value of the byte. This function decodes the URI restoring its original UTF-8 encoded content. Decoding is done in-place.

10.15.3.2 fl_filename_absolute() [1/2]

```
int fl_filename_absolute (
    char * to,
    int tolen,
    const char * from)
```

Makes a filename absolute from a relative filename to the current working directory.

```
#include <FL/filename.H>
[...]
```

```
fl_chdir("/var/tmp");
fl_filename_absolute(out, sizeof(out), "foo.txt");           // out="/var/tmp/foo.txt"
fl_filename_absolute(out, sizeof(out), "../foo.txt");       // out="/var/tmp/foo.txt"
fl_filename_absolute(out, sizeof(out), "../log/messages");  // out="/var/log/messages"
```

Parameters

out	<i>to</i>	resulting absolute filename
in	<i>tolen</i>	size of the absolute filename buffer
in	<i>from</i>	relative filename

Returns

0 if no change, non zero otherwise

10.15.3.3 fl_filename_absolute() [2/2]

```
int fl_filename_absolute (
    char * to,
    int tolen,
    const char * from,
    const char * base)
```

Concatenate the absolute path *base* with *from* to form the new absolute path in *to*.

```
#include <FL/filename.H>
char out[FL_PATH_MAX];
fl_filename_absolute(out, sizeof(out), "../foo.txt", "/var/tmp"); // out="/var/foo.txt"
fl_filename_absolute(out, sizeof(out), "../local/bin", "/usr/bin"); // out="/usr/local/bin"
```

Parameters

out	<i>to</i>	resulting absolute filename
in	<i>tolen</i>	size of the absolute filename buffer
in	<i>from</i>	relative filename
in	<i>base</i>	<i>from</i> is relative to this absolute file path

Returns

0 if no change, non zero otherwise

10.15.3.4 fl_filename_absolute_str() [1/2]

```
std::string fl_filename_absolute_str (
    const std::string & from)
```

Makes a filename absolute from a filename relative to the current working directory.

Parameters

in	<i>from</i>	relative filename

Returns

the new, absolute filename

See also

[fl_filename_absolute\(char *to, int tolen, const char *from\)](#)

10.15.3.5 fl_filename_absolute_str() [2/2]

```
std::string fl_filename_absolute_str (
    const std::string & from,
    const std::string & base)
```

Append the relative filename *from* to the absolute filename *base* to form the new absolute path.

Parameters

in	<i>from</i>	relative filename
in	<i>base</i>	<i>from</i> is relative to this absolute file path

Returns

the new, absolute filename

See also

[fl_filename_absolute\(char *to, int tolen, const char *from, const char *base\)](#)

10.15.3.6 fl_filename_expand()

```
int fl_filename_expand (
    char * to,
    int tolen,
    const char * from)
```

Expands a filename containing shell variables and tilde (~).

Currently handles these variants:

```
"~username"           // if 'username' does not exist, result will be unchanged
"~/file"              // does NOT handle ${VARIABLE}
"${VARIABLE}"
```

Examples:

```
#include <FL/filename.H>
[...
putenv("TMPDIR=/var/tmp");
fl_filename_expand(out, sizeof(out), "~fred/.cshrc");    // out="/usr/fred/.cshrc"
fl_filename_expand(out, sizeof(out), "~/cshrc");         // out="/usr/<yourname>/cshrc"
fl_filename_expand(out, sizeof(out), "$TMPDIR/foo.txt"); // out="/var/tmp/foo.txt"
```

Parameters

out	<i>to</i>	resulting expanded filename
in	<i>tolen</i>	size of the expanded filename buffer
in	<i>from</i>	filename containing shell variables

Returns

0 if no change, non zero otherwise

10.15.3.7 fl_filename_expand_str()

```
std::string fl_filename_expand_str (
    const std::string & from)
```

Expands a filename containing shell variables and tilde (~).

Parameters

in	from	file path and name

Returns

the new, expanded filename

See also

[fl_filename_expand\(char *to, int tolen, const char *from\)](#)

10.15.3.8 fl_filename_ext()

```
const char * fl_filename_ext (
    const char * buf)
```

Gets the extension of a filename.

```
#include <FL/filename.H>
[...]
```

```
const char *out;
out = fl_filename_ext("/some/path/foo.txt");           // result: ".txt"
out = fl_filename_ext("/some/path/foo");               // result: NULL
```

Parameters

in	buf	the filename to be parsed

Returns

a pointer to the extension (including '.') if any or NULL otherwise

10.15.3.9 fl_filename_ext_str()

```
std::string fl_filename_ext_str (
    const std::string & filename)
```

Return a new string that contains the filename extension.

Parameters

in	filename	file path and name

Returns

the filename extension including the prepending '.', or an empty string if the filename has no extension

See also

[fl_filename_ext\(const char *buf\)](#)

10.15.3.10 fl_filename_free_list()

```
void fl_filename_free_list (
    struct dirent *** list,
    int n)
```

Free the list of filenames that is generated by [fl_filename_list\(\)](#).

Free everything that was allocated by a previous call to [fl_filename_list\(\)](#). Use the return values as parameters for this function.

Parameters

in, out	<i>list</i>	table containing the resulting directory listing
in	<i>n</i>	number of entries in the list

10.15.3.11 fl_filename_isdir()

```
int fl_filename_isdir (
    const char * n)
```

Determines if a file exists and is a directory from its filename.

```
#include <FL/filename.H>
[...
fl_filename_isdir("/etc");           // returns non-zero
fl_filename_isdir("/etc/hosts");    // returns 0
```

Parameters

in	<i>n</i>	the filename to parse

Returns

non zero if file exists and is a directory, zero otherwise

10.15.3.12 fl_filename_list()

```
int fl_filename_list (
    const char * d,
    dirent *** list,
    Fl_File_Sort_F * sort)
```

Portable and const-correct wrapper for the `scandir()` function.

For each file in that directory a "dirent" structure is created. The only portable thing about a dirent is that `dirent.d_name` is the nul-terminated file name. A pointers array to these dirent's is created and a pointer to the array is returned in `*list`. The number of entries is given as a return value. If there is an error reading the directory a number less than zero is returned, and `errno` has the reason; `errno` does not work under Windows.

Include:

```
#include <FL/filename.H>
```

Parameters

in	<i>d</i>	the name of the directory to list. It does not matter if it has a trailing slash.
out	<i>list</i>	table containing the resulting directory listing

Parameters

<code>in</code>	<code>sort</code>	sorting functor: <code>fl_alphasort</code>: The files are sorted in ascending alphabetical order; upper and lowercase letters are compared according to their ASCII ordering uppercase before lowercase. <code>fl_casealphasort</code>: The files are sorted in ascending alphabetical order; upper and lowercase letters are compared equally case is not significant. <code>fl_casenumERICsort</code>: The files are sorted in ascending "alphanumeric" order, where an attempt is made to put unpadding numbers in consecutive order; upper and lowercase letters are compared equally case is not significant. <code>fl_numericSORT</code>: The files are sorted in ascending "alphanumeric" order, where an attempt is made to put unpadding numbers in consecutive order; upper and lowercase letters are compared according to their ASCII ordering - uppercase before lowercase.

Returns

the number of entries if no error, a negative value otherwise.

Todo should support returning OS error messages

10.15.3.13 `fl_filename_match()`

```
int fl_filename_match (
    const char * s,
    const char * p)
```

Checks if a string `s` matches a pattern `p`.

The following syntax is used for the pattern:

- `*` matches any sequence of 0 or more characters.
- `?` matches any single character.
- `[set]` matches any character in the set. Set can contain any single characters, or a-z to represent a range. To match `^` or `!` they must not be the first characters.
- `[^set]` or `[!set]` matches any character not in the set.
- `{X|Y|Z}` or `{X,Y,Z}` matches any one of the subexpressions literally.
- `\x` quotes the character `x` so it has no special meaning.
- `x` all other characters are matched "exactly" on a **case-insensitive** basis.

Notes:

- `s` and `p` are matched on a char/byte basis, not as UCS codepoints or UTF-8 sequences.
- `[set]` ranges must run from low to high, i.e. `[a-z]` and not `[z-a]`
- `[set]` comparison is **case-sensitive**, i.e. `[a-z]` won't match `"A"`.
- `\x` only applies to the `fl_filename_match` special characters `*` `?` `[` `{`
- `\x` needs a double `\` or the compiler will complain about non-standard escape sequences.

Include:

```
#include <FL/filename.H>
```


Parameters

in	<i>s</i>	the string to check for a match
in	<i>p</i>	the string pattern

Returns

non zero if the string matches the pattern

10.15.3.14 fl_filename_name()

```
const char * fl_filename_name (
    const char * filename)
```

Gets the file name from a path.

Similar to `basename(3)`, exceptions shown below.

```
#include <FL/filename.H>
[...]
```

```
const char *out;
out = fl_filename_name("/usr/lib");      // out="lib"
out = fl_filename_name("/usr/");        // out=""      (basename(3) returns "usr" instead)
out = fl_filename_name("/usr");         // out="usr"
out = fl_filename_name("/");            // out=""      (basename(3) returns "/" instead)
out = fl_filename_name(".");            // out="."
out = fl_filename_name("..");           // out=".."
```

Returns

a pointer to the char after the last slash, or to `filename` if there is none.

10.15.3.15 fl_filename_name_str()

```
std::string fl_filename_name_str (
    const std::string & filename)
```

Return a new string that contains the name part of the filename.

Parameters

in	<i>filename</i>	file path and name

Returns

the name part of a filename

See also

[fl_filename_name\(const char *filename\)](#)

10.15.3.16 fl_filename_path_str()

```
std::string fl_filename_path_str (
    const std::string & filename)
```

Return a new string that contains the path part of the filename.

Parameters

in	<i>filename</i>	file path and name

Returns

the path part of a filename without the name

See also

[fl_filename_name\(const char *filename\)](#)

10.15.3.17 fl_filename_relative() [1/2]

```
int fl_filename_relative (
    char * to,
    int tolen,
    const char * from)
```

Makes a filename relative to the current working directory.

Return the *from* path made relative to the working directory, similar to C++17 `std::filesystem::path::lexically_relative`. This function can also be called with a fourth argument for a user supplied *base* directory path

These conversions are purely lexical. They do not check that the paths exist, do not follow symlinks, and do not access the filesystem at all.

Path arguments must be absolute (start at the root directory) and must not contain `.` or `..` segments, or double separators. A single trailing separator is ok.

On Windows, path arguments must start with a drive name, e.g. `c:\`. Windows network paths and other special paths starting with a double separator are not supported (`\\cloud\drive\path`, `\\?\`, etc.) . Separators can be `\` and `/` and will be preserved. Newly created separators are always the forward slash `/`.

On Windows and macOS, the path segment tests are case insensitive.

If the path can not be generated, *from* path is copied into the *to* buffer and 0 is returned.

```
#include <FL/filename.H>
[..]
fl_chdir("/var/tmp/somedir");          // set cwd to /var/tmp/somedir
[..]
char out[FL_PATH_MAX];
fl_filename_relative(out, sizeof(out), "/var/tmp/somedir/foo.txt"); // out="foo.txt", return=1
fl_filename_relative(out, sizeof(out), "/var/tmp/foo.txt");         // out="../foo.txt", return=1
fl_filename_relative(out, sizeof(out), "foo.txt");                  // out="foo.txt", return=0 (no
change)
fl_filename_relative(out, sizeof(out), "../foo.txt");              // out="../foo.txt", return=0 (no
change)
fl_filename_relative(out, sizeof(out), "../foo.txt");              // out="../foo.txt", return=0 (no
change)
```

Parameters

out	<i>to</i>	resulting relative filename
in	<i>tolen</i>	size of the relative filename buffer
in	<i>from</i>	absolute filename

Returns

0 if no change, non zero otherwise

See also

[fl_filename_relative\(char *to, int tolen, const char *from, const char *base\)](#)

10.15.3.18 fl_filename_relative() [2/2]

```
int fl_filename_relative (
    char * to,
    int tolen,
    const char * from,
    const char * base)
```

Makes a filename relative to any other directory.

Parameters

out	<i>to</i>	resulting relative filepath
in	<i>tolen</i>	size of the relative filename buffer
in	<i>from</i>	absolute filepath
in	<i>base</i>	generate filepath relative to this absolute filepath

Returns

0 if no change, non zero otherwise

See also

[fl_filename_relative\(char *to, int tolen, const char *from\)](#)

10.15.3.19 fl_filename_relative_str() [1/2]

```
std::string fl_filename_relative_str (
    const std::string & from)
```

Makes a filename relative to the current working directory.

Parameters

<i>in</i>	<i>from</i>	file path and name

Returns

the new, relative filename

See also

[fl_filename_relative\(char *to, int tolen, const char *from\)](#)

10.15.3.20 fl_filename_relative_str() [2/2]

```
std::string fl_filename_relative_str (
    const std::string & from,
    const std::string & base)
```

Makes a filename relative to any directory.

Parameters

<i>in</i>	<i>from</i>	file path and name
<i>in</i>	<i>base</i>	relative to this absolute path

Returns

the new, relative filename

See also

[fl_filename_relative\(char *to, int tolen, const char *from, const char *base\)](#)

10.15.3.21 fl_filename_setext()

```
char * fl_filename_setext (
    char * buf,
    int buflen,
    const char * ext)
```

Replaces the extension in *buf* of max.

size *buflen* with the extension in *ext*.

If there's no '.' in *buf*, *ext* is appended.

If *ext* is NULL, behaves as if it were an empty string ("").

Example

```
#include <FL/filename.H>
[...
char buf[FL_PATH_MAX] = "/path/myfile.cxx";
fl_filename_setext(buf, sizeof(buf), ".txt");           // buf[] becomes "/path/myfile.txt"
```

Returns

buf itself for calling convenience.

10.15.3.22 fl_filename_setext_str()

```
std::string fl_filename_setext_str (
    const std::string & filename,
    const std::string & new_extension)
```

Return a copy of the old filename with the new extension.

Parameters

in	<i>filename</i>	file path and name
in	<i>new_extension</i>	new filename extension, starts with a '.'

Returns

the new filename

See also

[fl_filename_setext\(char *to, int tolen, const char *ext\)](#)

10.15.3.23 fl_getcwd_str()

```
std::string fl_getcwd_str ()
```

Cross-platform function to get the current working directory as a UTF-8 encoded value in an std::string.

Returns

the CWD encoded as UTF-8

10.15.3.24 fl_open_uri()

```
int fl_open_uri (
    const char * uri,
    char * msg,
    int msglen)
```

Opens the specified Uniform Resource Identifier (URI).

Uses an operating-system dependent program or interface. For URIs using the "ftp", "http", or "https" schemes, the system default web browser is used to open the URI, while "mailto" and "news" URIs are typically opened using the system default mail reader and "file" URIs are opened using the file system navigator.

On success, the (optional) msg buffer is filled with the command that was run to open the URI; on Windows, this will always be "open uri".

On failure, the msg buffer is filled with an English error message.

Note**Platform Specific Issues: Windows**

With "file:" based URIs on Windows, you may encounter issues with anchors being ignored. Example: "file://c:/some/index.html#anchor" may open in the browser without the "#anchor" suffix. The behavior seems to vary across different Windows versions. Workaround: open a link to a separate html file that redirects to the desired "file:" URI.

Example

```
#include <FL/filename.H>
[...]
char errmsg[512];
if ( !fl_open_uri("http://google.com/", errmsg, sizeof(errmsg)) ) {
    char warnmsg[768];
    sprintf(warnmsg, "Error: %s", errmsg);
    fl_alert(warnmsg);
}
```

Parameters

<i>uri</i>	The URI to open
<i>msg</i>	Optional buffer which contains the command or error message
<i>msglen</i>	Length of optional buffer

Returns

1 on success, 0 on failure

Chapter 11

Class Documentation

11.1 `Fl_Grid::Cell` Class Reference

Public Member Functions

- `Fl_Grid_Align align ()` const
- void `align (Fl_Grid_Align align)`
- `Cell (Fl_Widget *w, int row, int col)`
- `Cell (int row, int col)`
- void `Cell_ ()`
- short `col ()` const
- short `colspan ()` const
- void `colspan (short v)`
- void `minimum_size (int *w, int *h)` const
- void `minimum_size (int w, int h)`
- `Cell * next ()`
Returns the next widget cell of the same row of this cell.
- void `next (Cell *c)`
Sets the next pointer of a grid's cell.
- short `row ()` const
- short `rowspan ()` const
- void `rowspan (short v)`
- `Fl_Widget * widget ()` const
- `~Cell ()`
The destructor deletes the cell.

Friends

- class `Fl_Grid`

11.1.1 Constructor & Destructor Documentation

11.1.1.1 `~Cell()`

`Fl_Grid::Cell::~Cell ()` [inline]

The destructor deletes the cell.

Todo `Fl_Grid`'s cell destructor should remove the cell from the grid. Currently it does nothing!

11.1.2 Member Function Documentation

11.1.2.1 next()

```
void Fl_Grid::Cell::next (
    Cell * c) [inline]
```

Sets the `next` pointer of a grid's cell.

Internal use only!

Do not use this method, it may corrupt the allocated memory.

The documentation for this class was generated from the following file:

- [Fl_Grid.H](#)

11.2 Fl_Terminal::CharStyle Class Reference

Public Member Functions

- void **attrib** ([uchar](#) val)
- [uchar](#) **attrib** (void) const
- void **bgcolor** ([Fl_Color](#) val)
- void **bgcolor** (int r, int g, int b)
- [Fl_Color](#) **bgcolor** (void) const
- void **bgcolor_xterm** ([Fl_Color](#) val)
- void **bgcolor_xterm** ([uchar](#) val)
- void **charflags** ([uchar](#) val)
- [uchar](#) **charflags** (void) const
- **CharStyle** (bool fontsize_defer)
- int **charwidth** (void) const
- void **clr_charflag** ([uchar](#) val)
- [uchar](#) **colorbits_only** ([uchar](#) inflags) const
- void **defaultbgcolor** ([Fl_Color](#) val)
- [Fl_Color](#) **defaultbgcolor** (void) const
- void **defaultfgcolor** ([Fl_Color](#) val)
- [Fl_Color](#) **defaultfgcolor** (void) const
- void **fgcolor** ([Fl_Color](#) val)
- void **fgcolor** (int r, int g, int b)
- [Fl_Color](#) **fgcolor** (void) const
- void **fgcolor_xterm** ([Fl_Color](#) val)
- void **fgcolor_xterm** ([uchar](#) val)
- [Fl_Color](#) **fltk_bg_color** ([uchar](#) ci)
- [Fl_Color](#) **fltk_fg_color** ([uchar](#) ci)
- int **fontdescent** (void) const
- void **fontface** ([Fl_Font](#) val)
- [Fl_Font](#) **fontface** (void) const
- int **fontheight** (void) const
- void **fontsize** ([Fl_Fontsize](#) val)
- [Fl_Fontsize](#) **fontsize** (void) const
- int **onoff** (bool flag, [Attrib](#) a)
- void **set_charflag** ([uchar](#) val)
- void **sgr_blink** (bool val)
- void **sgr_bold** (bool val)
- void **sgr_dbl_under** (bool val)
- void **sgr_dim** (bool val)
- void **sgr_inverse** (bool val)
- void **sgr_italic** (bool val)
- void **sgr_reset** (void)

- void **sgr_strike** (bool val)
- void **sgr_underline** (bool val)
- void **update** (void)
- void **update_fake** (void)

The documentation for this class was generated from the following files:

- [FI_Terminal.H](#)
- [FI_Terminal.cxx](#)

11.3 FI_GIF_Image::GIF_FRAME::CPAL Struct Reference

Public Attributes

- [uchar](#) **b**
- [uchar](#) **g**
- [uchar](#) **r**

The documentation for this struct was generated from the following file:

- [FI_GIF_Image.H](#)

11.4 FI_Terminal::Cursor Class Reference

Public Member Functions

- void **bgcolor** ([FI_Color](#) val)
- [FI_Color](#) **bgcolor** (void) const
- void **col** (int val)
- int **col** (void) const
- int **down** (void)
- void **fgcolor** ([FI_Color](#) val)
- [FI_Color](#) **fgcolor** (void) const
- void **h** (int val)
- int **h** (void) const
- void **home** (void)
- bool **is_rowcol** (int drow, int dcol) const
- int **left** (void)
- int **right** (void)
- void **row** (int val)
- int **row** (void) const
- void **scroll** (int nrow)
- int **up** (void)

The documentation for this class was generated from the following files:

- [FI_Terminal.H](#)
- [FI_Terminal.cxx](#)

11.5 FI_Preferences::Entry Struct Reference

Public Attributes

- char * **name**
- char * **value**

The documentation for this struct was generated from the following file:

- [FI_Preferences.H](#)

11.6 FL_Terminal::EscapeSeq Class Reference

Public Member Functions

- int **defvalmax** (int dval, int max) const
- void **esc_mode** (char val)
- char **esc_mode** (void) const
- bool **is_csi** (void) const
- int **parse** (char c)
- bool **parse_in_progress** (void) const
- void **reset** (void)
- void **restore_cursor** (int &row, int &col)
- void **save_cursor** (int row, int col)
- int **total_vals** (void) const
- int **val** (int i) const

Static Public Attributes

- static const int **completed** = 1
- static const int **fail** = -1
- static const int **maxbuff** = 80
- static const int **maxvals** = 20
- static const int **success** = 0

The documentation for this class was generated from the following files:

- [FL_Terminal.H](#)
- [FL_Terminal.cxx](#)

11.7 FL Class Reference

The [FL](#) is the FLTK global (static) class containing state information and global methods for the current application.
`#include <FL.H>`

Public Types

- enum [FL_Option](#) {
[OPTION_ARROW_FOCUS](#) = 0 , [OPTION_VISIBLE_FOCUS](#) , [OPTION_DND_TEXT](#) , [OPTION_SHOW_TOOLTIPS](#)
 ,
[OPTION_FNFC_USES_GTK](#) , [OPTION_FNFC_USES_ZENITY](#) , [OPTION_FNFC_USES_KDIALOG](#) ,
[OPTION_PRINTER_USES_GTK](#) ,
[OPTION_SHOW_SCALING](#) , [OPTION_SIMPLE_ZOOM_SHORTCUT](#) , [OPTION_LAST](#) }

Enumerator for global FLTK options.

Static Public Member Functions

- static int [abi_check](#) (const int val=[FL_ABI_VERSION](#))
Returns whether the runtime library ABI version is correct.
- static int [abi_version](#) ()
Returns the compiled-in value of the [FL_ABI_VERSION](#) constant.
- static int [add_awake_handler_](#) ([FL_Awake_Handler](#), void *)
Adds an awake handler for use in [awake\(\)](#).
- static void [add_check](#) ([FL_Timeout_Handler](#), void *=0)
FLTK will call this callback just before it flushes the display and waits for events.
- static void [add_clipboard_notify](#) ([FL_Clipboard_Notify_Handler](#) h, void *data=0)
FLTK will call the registered callback whenever there is a change to the selection buffer or the clipboard.

- static void `add_fd` (int fd, `FI_FD_Handler` cb, void *=0)
Adds file descriptor fd to listen to.
- static void `add_fd` (int fd, int when, `FI_FD_Handler` cb, void *=0)
Adds file descriptor fd to listen to.
- static void `add_handler` (`FI_Event_Handler` ha)
Install a function to parse unrecognized events.
- static void `add_handler` (`FI_Event_Handler` ha, `FI_Event_Handler` before)
Install a function to parse unrecognized events with less priority than another function.
- static void `add_idle` (`FI_Idle_Handler` cb, void *data=0)
Adds a callback function that is called every time by `FI::wait()` and also makes it act as though the timeout is zero (this makes `FI::wait()` return immediately, so if it is in a loop it is called repeatedly, and thus the idle function is called repeatedly).
- static void `add_system_handler` (`FI_System_Handler` h, void *data)
Install a function to intercept system events.
- static void `add_timeout` (double t, `FI_Timeout_Handler` cb, void *data=0)
Adds a one-shot timeout callback.
- static int `api_version` ()
Returns the compiled-in value of the `FL_API_VERSION` constant.
- static int `arg` (int argc, char **argv, int &i)
Parse a single switch from argv, starting at word i.
- static void `args` (int argc, char **argv)
Parse all command line switches matching standard FLTK options only.
- static int `args` (int argc, char **argv, int &i, `FI_Args_Handler` cb=0)
Parse command line switches using the cb argument handler.
- static int `args_to_utf8` (int argc, char **&argv)
Convert Windows commandline arguments to UTF-8.
- static int `awake` (`FI_Awake_Handler` cb, void *message=0)
See void awake(void message=0).*
- static void `awake` (void *message=0)
Sends a message pointer to the main thread, causing any pending `FI::wait()` call to terminate so that the main thread can retrieve the message and any pending redraws can be processed.
- static void `background` (uchar, uchar, uchar)
Changes fl_color(FL_BACKGROUND_COLOR) to the given color, and changes the gray ramp from 32 to 56 to black to white.
- static void `background2` (uchar, uchar, uchar)
Changes the alternative background color.
- static `FI_Widget` * `belowmouse` ()
Gets the widget that is below the mouse.
- static void `belowmouse` (`FI_Widget` *)
Sets the widget that is below the mouse.
- static int `box_border_radius_max` ()
Get the maximum border radius of all "rounded" boxtypes in pixels.
- static void `box_border_radius_max` (int R)
Set the maximum border radius of all "rounded" boxtypes in pixels.
- static `FI_Color` `box_color` (`FI_Color`)
Gets the drawing color to be used for the background of a box.
- static int `box_dh` (`FI_Boxtype`)
Returns the height offset for the given boxtype.
- static int `box_dw` (`FI_Boxtype`)
Returns the width offset for the given boxtype.
- static int `box_dx` (`FI_Boxtype`)

- Returns the X offset for the given boxtype.*

 - static int `box_dy` (`Fl_Boxtype`)
- Returns the Y offset for the given boxtype.*

 - static int `box_shadow_width` ()
- Get the box shadow width of all "shadow" boxtypes in pixels.*

 - static void `box_shadow_width` (int W)
- Set the box shadow width of all "shadow" boxtypes in pixels.*

 - static bool `cairo_autolink_context` ()
- Gets the current autolink mode for Cairo support.*

 - static void `cairo_autolink_context` (bool alink)
- When FLTK_HAVE_CAIRO is defined and `cairo_autolink_context()` is true, any current window dc is linked to a current Cairo context.*

 - static cairo_t * `cairo_cc` ()
- Gets the current Cairo context linked with a fltk window.*

 - static void `cairo_cc` (cairo_t *c, bool own=false)
- Sets the current Cairo context to c.*

 - static void `cairo_flush` (cairo_t *c)
- Flush Cairo drawings on Cairo context c.*

 - static cairo_t * `cairo_make_current` (`Fl_Window` *w)
- Provides a Cairo context for window wi.*

 - static `Fl_Callback_Reason` `callback_reason` ()
- Give the reason for calling a callback.*

 - static int `check` ()
- Same as Fl::wait(0).*

 - static void `clear_widget_pointer` (`Fl_Widget` const *w)
- Clears a widget pointer in the watch list.*

 - static int `clipboard_contains` (const char *type)
- Returns non 0 if the clipboard contains data matching type.*

 - static int `compose` (int &del)
- Any text editing widget should call this for each FL_KEYBOARD event.*

 - static void `compose_reset` ()
- If the user moves the cursor, be sure to call Fl::compose_reset().*

 - static void `copy` (const char *stuff, int len, int destination=0, const char *type=`Fl::clipboard_plain_text`)
- Copies data to the selection buffer, the clipboard, or both.*

 - static int `damage` ()
- If true then flush() will do something.*

 - static void `damage` (int d)
- If true then flush() will do something.*

 - static void `default_atclose` (`Fl_Window` *, void *)
- Default callback for window widgets.*

 - static void `delete_widget` (`Fl_Widget` *w)
- Schedules a widget for deletion at the next call to the event loop.*

 - static void `disable_im` ()
- Disables the system input methods facilities.*

 - static void `display` (const char *)
- Sets the X or Wayland display to use for all windows.*

 - static int `dnd` ()
- Initiate a Drag And Drop operation.*

 - static int `dnd_text_ops` ()
- Gets whether drag and drop text operations are supported.*

 - static void `dnd_text_ops` (int v)

- Sets whether drag and drop text operations are supported.*
- static void `do_widget_deletion` ()
Deletes widgets previously scheduled for deletion.
- static int `draw_box_active` ()
Determines if the currently drawn box is active or inactive.
- static int `draw_GL_text_with_textures` ()
returns whether whether OpenGL uses textures to draw all text.
- static void `draw_GL_text_with_textures` (int val)
sets whether OpenGL uses textures to draw all text.
- static void `enable_im` ()
Enables the system input methods facilities.
- static int `event` ()
Returns the last event that was processed.
- static int `event_alt` ()
Returns non-zero if the Alt key is pressed.
- static int `event_button` ()
Gets which particular mouse button caused the current event.
- static int `event_button1` ()
Returns non-zero if mouse button 1 is currently held down.
- static int `event_button2` ()
Returns non-zero if mouse button 2 is currently held down.
- static int `event_button3` ()
Returns non-zero if mouse button 3 is currently held down.
- static int `event_button4` ()
Returns non-zero if mouse button 4 is currently held down.
- static int `event_button5` ()
Returns non-zero if mouse button 5 is currently held down.
- static int `event_buttons` ()
Returns the mouse buttons state bits; if non-zero, then at least one button is pressed now.
- static int `event_clicks` ()
Returns non zero if we had a double click event.
- static void `event_clicks` (int i)
Manually sets the number returned by `FL::event_clicks()`.
- static void * `event_clipboard` ()
During an FL_PASTE event of non-textual data, returns a pointer to the pasted data.
- static const char * `event_clipboard_type` ()
Returns the type of the pasted data during an FL_PASTE event.
- static int `event_command` ()
Returns non-zero if the FL_COMMAND key is pressed, either FL_CTRL or on OSX FL_META.
- static int `event_ctrl` ()
Returns non-zero if the Control key is pressed.
- static `FL_Event_Dispatch` `event_dispatch` ()
Return the current event dispatch function.
- static void `event_dispatch` (`FL_Event_Dispatch` d)
Set a new event dispatch function.
- static int `event_dx` ()
Returns the current horizontal mouse scrolling associated with the FL_MOUSEWHEEL event.
- static int `event_dy` ()
Returns the current vertical mouse scrolling associated with the FL_MOUSEWHEEL event.
- static int `event_inside` (const `FL_Widget` *)
Returns whether or not the mouse event is inside a given child widget.

- static int [event_inside](#) (int, int, int, int)
Returns whether or not the mouse event is inside the given rectangle.
- static int [event_is_click](#) ()
Returns non-zero if the mouse has not moved far enough and not enough time has passed since the last `FL_PUSH` or `FL_KEYBOARD` event for it to be considered a "drag" rather than a "click".
- static void [event_is_click](#) (int i)
Clears the value returned by `Fl::event_is_click()`.
- static int [event_key](#) ()
Gets which key on the keyboard was last pushed.
- static int [event_key](#) (int key)
Returns true if the given `key` was held down (or pressed) during the last event.
- static int [event_length](#) ()
Returns the length of the text in `Fl::event_text()`.
- static int [event_original_key](#) ()
Returns the keycode of the last key event, regardless of the NumLock state.
- static int [event_shift](#) ()
Returns non-zero if the Shift key is pressed.
- static int [event_state](#) ()
Returns the keyboard and mouse button states of the last event.
- static int [event_state](#) (int mask)
Returns non-zero if any of the passed event state bits are turned on.
- static const char * [event_text](#) ()
Returns the text associated with the current event, including `FL_PASTE` or `FL_DND_RELEASE` events.
- static int [event_x](#) ()
Returns the mouse position of the event relative to the `Fl_Window` it was passed to.
- static int [event_x_root](#) ()
Returns the mouse position on the screen of the event.
- static int [event_y](#) ()
Returns the mouse position of the event relative to the `Fl_Window` it was passed to.
- static int [event_y_root](#) ()
Returns the mouse position on the screen of the event.
- static `Fl_Window` * [first_window](#) ()
Returns the first top-level window in the list of `shown()` windows.
- static void [first_window](#) (`Fl_Window` *)
Sets the window that is returned by `first_window()`.
- static void [flush](#) ()
Causes all the windows that need it to be redrawn and graphics forced out through the pipes.
- static `Fl_Widget` * [focus](#) ()
Gets the current `Fl::focus()` widget.
- static void [focus](#) (`Fl_Widget` *)
Sets the widget that will receive `FL_KEYBOARD` events.
- static void [foreground](#) (`uchar`, `uchar`, `uchar`)
Changes `fl_color(FL_FOREGROUND_COLOR)`.
- static void [free_color](#) (`Fl_Color` i, int overlay=0)
Frees the specified color from the colormap, if applicable.
- static int [get_awesome_handler_](#) (`Fl_Awake_Handler` &, void *&)
Gets the last stored awake handler for use in `awake()`.
- static `Fl_Box_Draw_F` * [get_boxtype](#) (`Fl_Boxtype`)
Gets the current box drawing function for the specified box type.
- static unsigned [get_color](#) (`Fl_Color` i)
Returns the RGB value(s) for the given FLTK color index.

- static void `get_color` (`FL_Color` i, `uchar` &red, `uchar` &green, `uchar` &blue)

Returns the RGB value(s) for the given FLTK color index.
- static void `get_color` (`FL_Color` i, `uchar` &red, `uchar` &green, `uchar` &blue, `uchar` &alpha)

Returns the RGBA value(s) for the given FLTK color index.
- static const char * `get_font` (`FL_Font`)

Gets the string for this face.
- static const char * `get_font_name` (`FL_Font`, int *attributes=0)

Get a human-readable string describing the family of this face.
- static int `get_font_sizes` (`FL_Font`, int *&sizep)

Return an array of sizes in sizep.
- static int `get_key` (int key)

Returns true if the given key is held down now.
- static void `get_mouse` (int &, int &)

Return where the mouse is on the screen by doing a round-trip query to the server.
- static void `get_system_colors` ()

Read the user preference colors from the system and use them to call `Fl::foreground()`, `Fl::background()`, and `Fl::background2()`.
- static int `gl_visual` (int, int *alist=0)

This does the same thing as `Fl::visual(int)` but also requires OpenGL drawing to work.
- static `FL_Window` * `grab` ()

Returns the window that currently receives all events.
- static void `grab` (`FL_Window` &win)

See `grab(FL_Window*)`
- static void `grab` (`FL_Window` *)

Selects the window to grab.
- static int `h` ()

Returns the height in pixels of the main screen work area.
- static int `handle` (int, `FL_Window` *)

Handle events from the window system.
- static int `handle_` (int, `FL_Window` *)

Handle events from the window system.
- static int `has_check` (`FL_Timeout_Handler`, void *=0)

Returns 1 if the check exists and has not been called yet, 0 otherwise.
- static int `has_idle` (`FL_Idle_Handler` cb, void *data=0)

Returns true if the specified idle callback is currently installed.
- static int `has_timeout` (`FL_Timeout_Handler` cb, void *data=0)

Returns true if the timeout exists and has not been called yet.
- static void `hide_all_windows` ()

Hide all visible windows to make FLTK leave `Fl::run()`.
- static int `is_scheme` (const char *name)

Returns whether the current scheme is the given name.
- static void `keyboard_screen_scaling` (int value)

Controls the possibility to scale all windows by ctrl+/-/0/ or cmd+/-/0/.
- static `FL_Event_Handler` `last_handler` ()

Returns the last function installed by a call to `Fl::add_handler()`.
- static int `lock` ()

The `lock()` method blocks the current thread until it can safely access FLTK widgets and data.
- static int `menu_linespacing` ()

Gets the default line spacing used by menus.
- static void `menu_linespacing` (int H)

Sets the default line spacing used by menus.

- static `FL_Window * modal ()`
Returns the top-most *modal()* window currently shown.
- static `FL_Window * next_window (const FL_Window *)`
Returns the next top-level window in the list of shown() windows.
- static `FL_Timestamp now (double offset=0)`
Set a time stamp at this point in time with optional signed offset in seconds.
- static bool `option (FL_Option opt)`
FLTK library options management.
- static void `option (FL_Option opt, bool val)`
Override an option while the application is running.
- static void `own_colormap ()`
Makes FLTK use its *own colormap*.
- static void `paste (FL_Widget &receiver)`
Backward compatibility only.
- static void `paste (FL_Widget &receiver, int source, const char *type=FL::clipboard_plain_text)`
Pastes the data from the selection buffer (*source is 0*) or the clipboard (*source is 1*) into receiver.
- static int `program_should_quit ()`
Returns non-zero when a request for program termination was received and accepted.
- static void `program_should_quit (int should_i)`
Indicate to the FLTK library whether a program termination request was received and accepted.
- static `FL_Widget * pushed ()`
Gets the widget that is being pushed.
- static void `pushed (FL_Widget *)`
Sets the widget that is being pushed.
- static `FL_Widget * readqueue ()`
Reads the default callback queue and returns the first widget.
- static int `ready ()`
This is similar to *FL::check()* except this does not call *FL::flush()* or any callbacks, which is useful if your program is in a state where such callbacks are illegal.
- static void `redraw ()`
Redraws all widgets.
- static void `release ()`
Releases the current grabbed window, equals *grab(0)*.
- static void `release_widget_pointer (FL_Widget *&w)`
Releases a widget pointer from the watch list.
- static int `reload_scheme ()`
Called internally when setting a new scheme according to scheme name.
- static void `remove_check (FL_Timeout_Handler, void *=0)`
Removes a check callback.
- static void `remove_clipboard_notify (FL_Clipboard_Notify_Handler h)`
Stop calling the specified callback when there are changes to the selection buffer or the clipboard.
- static void `remove_fd (int)`
Removes a file descriptor handler.
- static void `remove_fd (int, int when)`
Removes a file descriptor handler.
- static void `remove_handler (FL_Event_Handler h)`
Removes a previously added event handler.
- static void `remove_idle (FL_Idle_Handler cb, void *data=0)`
Removes the specified idle callback, if it is installed.
- static int `remove_next_timeout (FL_Timeout_Handler cb, void *data=0, void **data_return=0)`
Remove the next matching timeout callback and return its *data* pointer.

- static void [remove_system_handler](#) (FI_System_Handler h)
Removes a previously added system event handler.
- static void [remove_timeout](#) (FI_Timeout_Handler cb, void *data=0)
Remove one or more matching timeout callbacks from the timer queue.
- static void [repeat_timeout](#) (double t, FI_Timeout_Handler cb, void *data=0)
Repeats a timeout callback from the expiration of the previous timeout, allowing for more accurate timing.
- static int [run](#) ()
Calls [Fl::wait\(\)](#) repeatedly as long as any windows are displayed.
- static void [run_checks](#) ()
- static void [run_idle](#) ()
- static const char * [scheme](#) ()
*See void [scheme\(const char *name\)](#)*
- static int [scheme](#) (const char *name)
Sets the current widget scheme.
- static int [screen_count](#) ()
Gets the total count of available screens.
- static void [screen_dpi](#) (float &h, float &v, int n=0)
Gets the screen resolution in dots-per-inch for the given screen.
- static FI_Screen_Driver * [screen_driver](#) ()
Returns a pointer to the unique FI_Screen_Driver object of the platform.
- static int [screen_num](#) (int x, int y)
Gets the screen number of a screen that contains the specified screen position x, y.
- static int [screen_num](#) (int x, int y, int w, int h)
Gets the screen number of the screen which intersects the most with the rectangle defined by x, y, w, h.
- static float [screen_scale](#) (int n)
Gets the GUI scaling factor of screen number n.
- static void [screen_scale](#) (int n, float factor)
Sets the GUI scaling factor of screen number n.
- static int [screen_scaling_supported](#) ()
Returns whether scaling factors are supported by this platform.
- static void [screen_work_area](#) (int &X, int &Y, int &W, int &H)
Gets the bounding box of the work area of the screen that contains the mouse pointer.
- static void [screen_work_area](#) (int &X, int &Y, int &W, int &H, int mx, int my)
Gets the bounding box of the work area of a screen that contains the specified screen position mx, my.
- static void [screen_work_area](#) (int &X, int &Y, int &W, int &H, int n)
Gets the bounding box of the work area of the given screen.
- static void [screen_xywh](#) (int &X, int &Y, int &W, int &H)
Gets the bounding box of a screen that contains the mouse pointer.
- static void [screen_xywh](#) (int &X, int &Y, int &W, int &H, int mx, int my)
Gets the bounding box of a screen that contains the specified screen position mx, my.
- static void [screen_xywh](#) (int &X, int &Y, int &W, int &H, int mx, int my, int mw, int mh)
Gets the screen bounding rect for the screen which intersects the most with the rectangle defined by mx, my, mw, mh.
- static void [screen_xywh](#) (int &X, int &Y, int &W, int &H, int n)
Gets the screen bounding rect for the given screen.
- static int [scrollbar_size](#) ()
Gets the default scrollbar size used by [FI_Browser](#), [FI_Help_View](#), [FI_Scroll](#), and [FI_Text_Display](#) widgets.
- static void [scrollbar_size](#) (int W)
Sets the default scrollbar size that is used by the [FI_Browser](#), [FI_Help_View](#), [FI_Scroll](#), and [FI_Text_Display](#) widgets.
- static double [seconds_between](#) (FI_Timestamp &back, FI_Timestamp &further_back)
Return the time in seconds between two time stamps.
- static double [seconds_since](#) (FI_Timestamp &then)

- Return the time in seconds between now and a previously taken time stamp.*

 - static void [selection](#) ([Fl_Widget](#) &owner, const char *, int len)

Changes the current selection.
- static [Fl_Widget](#) * [selection_owner](#) ()

back-compatibility only: Gets the widget owning the current selection
- static void [selection_owner](#) ([Fl_Widget](#) *)

Back-compatibility only: The single-argument call can be used to move the selection to another widget or to set the owner to NULL, without changing the actual text of the selection.
- static int [selection_to_clipboard](#) ()

Returns the current selection_to_clipboard mode.
- static void [selection_to_clipboard](#) (int mode)

Copies selections on X11 directly to the clipboard if enabled.
- static void **set_abort** ([Fl_Abort_Handler](#) f)

For back compatibility, sets the void [Fl::fatal](#) handler callback.
- static void [set_atclose](#) ([Fl_Atclose_Handler](#) f)

For back compatibility, sets the [Fl::atclose](#) handler callback.
- static void [set_box_color](#) ([Fl_Color](#))

Sets the drawing color for the box that is currently drawn.
- static void [set_boxtype](#) ([Fl_Boxtype](#), [Fl_Box_Draw_F](#) *, [uchar](#), [uchar](#), [uchar](#), [uchar](#), [Fl_Box_Draw_Focus_F](#) *[=NULL](#))

Sets the function to call to draw a specific box type.
- static void **set_boxtype** ([Fl_Boxtype](#), [Fl_Boxtype](#) from)

Copies the from boxtype.
- static void [set_color](#) ([Fl_Color](#) i, unsigned c)

Sets an entry in the fl_color index table.
- static void [set_color](#) ([Fl_Color](#), [uchar](#), [uchar](#), [uchar](#))

Sets an entry in the fl_color index table.
- static void [set_color](#) ([Fl_Color](#), [uchar](#), [uchar](#), [uchar](#), [uchar](#))

Sets an entry in the fl_color index table.
- static void [set_font](#) ([Fl_Font](#), const char *)

Changes a face.
- static void **set_font** ([Fl_Font](#), [Fl_Font](#))

Copies one face to another.
- static [Fl_Font](#) [set_fonts](#) (const char *[=0](#))

FLTK will open the display, and add every fonts on the server to the face table.
- static void [set_idle](#) ([Fl_Old_Idle_Handler](#) cb)

Sets an idle callback.
- static void **set_labeltype** ([Fl_Labeltype](#), [Fl_Label_Draw_F](#) *, [Fl_Label_Measure_F](#) *)

Sets the functions to call to draw and measure a specific labeltype.
- static void **set_labeltype** ([Fl_Labeltype](#), [Fl_Labeltype](#) from)

Sets the functions to call to draw and measure a specific labeltype.
- static int **system** (const char *command)

Run a command line on the computer.
- static [Fl_System_Driver](#) * **system_driver** ()

Returns a pointer to the unique [Fl_System_Driver](#) object of the platform.
- static int [test_shortcut](#) ([Fl_Shortcut](#))

Tests the current event, which must be an [FL_KEYBOARD](#) or [FL_SHORTCUT](#), against a shortcut value (described in [Fl_Button](#)).
- static void * [thread_message](#) ()

The [thread_message\(\)](#) method returns the last message that was sent from a child by the [awake\(\)](#) method.
- static long [ticks_between](#) ([Fl_Timestamp](#) &back, [Fl_Timestamp](#) &further_back)

- Return the time in ticks (60Hz) between two time stamps.*

 - static long `ticks_since` (`FL_Timestamp` &then)

Return the time in ticks (60Hz) between now and a previously taken time stamp.
- static void `unlock` ()

The `unlock()` method releases the lock that was set using the `lock()` method.
- static int `use_high_res_GL` ()

returns whether GL windows should be drawn at high resolution on Apple computers with retina displays.
- static void `use_high_res_GL` (int val)

sets whether GL windows should be drawn at high resolution on Apple computers with retina displays
- static double `version` ()

Returns the compiled-in value of the `FL_VERSION` constant.
- static int `visible_focus` ()

Gets or sets the visible keyboard focus on buttons and other non-text widgets.
- static void `visible_focus` (int v)

Gets or sets the visible keyboard focus on buttons and other non-text widgets.
- static int `visual` (int)

Selects a visual so that your graphics are drawn correctly.
- static int `w` ()

Returns the width in pixels of the main screen work area.
- static int `wait` ()

Waits until "something happens" and then returns.
- static double `wait` (double time)

Waits a maximum of `time_to_wait` seconds or until "something happens".
- static void `watch_widget_pointer` (`FL_Widget` *&w)

Adds a widget pointer to the widget watch list.
- static int `x` ()

Returns the leftmost x coordinate of the main screen work area.
- static int `y` ()

Returns the topmost y coordinate of the main screen work area.

Static Public Attributes

- static void(* `atclose`)(FL_Window *, void *)

Back compatibility: default window callback handler.
- static char const *const `clipboard_image` = "image"

Denotes image data.
- static char const *const `clipboard_plain_text` = "text/plain"

Denotes plain textual data.
- static void(* `error`)(const char *,...) = FL_System_Driver::error

FLTK calls `FL::error()` to output a normal error message.
- static void(* `fatal`)(const char *,...) = FL_System_Driver::fatal

FLTK calls `FL::fatal()` to output a fatal error message.
- static const char *const `help` = helpmsg+13

Usage string displayed if `FL::args()` detects an invalid argument.
- static void(* `idle`)()

The currently executing idle callback function: DO NOT USE THIS DIRECTLY!
- static void(* `warning`)(const char *,...) = FL_System_Driver::warning

FLTK calls `FL::warning()` to output a warning message.

Friends

- class `FL_System_Driver`

11.7.1 Detailed Description

The [Fl](#) is the FLTK global (static) class containing state information and global methods for the current application.

11.7.2 Member Enumeration Documentation

11.7.2.1 Fl_Option

enum [Fl::Fl_Option](#)

Enumerator for global FLTK options.

These options can be set system wide, per user, or for the running application only.

See also

[Fl::option\(Fl_Option, bool\)](#)

[Fl::option\(Fl_Option\)](#)

Enumerator

OPTION_ARROW_FOCUS	When switched on, moving the text cursor beyond the start or end of a text in a text widget will change focus to the next text widget. (This is considered 'old' behavior) When switched off (default), the cursor will stop at the end of the text. Pressing Tab or Ctrl-Tab will advance the keyboard focus. See also: Fl_Input::tab_nav()
OPTION_VISIBLE_FOCUS	If visible focus is switched on (default), FLTK will draw a dotted rectangle inside the widget that will receive the next keystroke. If switched off, no such indicator will be drawn and keyboard navigation is disabled.
OPTION_DND_TEXT	If text drag-and-drop is enabled (default), the user can select and drag text from any text widget. If disabled, no dragging is possible, however dropping text from other applications still works.
OPTION_SHOW_TOOLTIPS	If tooltips are enabled (default), hovering the mouse over a widget with a tooltip text will open a little tooltip window until the mouse leaves the widget. If disabled, no tooltip is shown.
OPTION_FNFC_USES_GTK	When switched on (default), Fl_Native_File_Chooser runs GTK file dialogs if the GTK library is available on the platform (linux/unix only). When switched off, GTK file dialogs aren't used even if the GTK library is available.
OPTION_FNFC_USES_ZENITY	Meaningful for the Wayland/X11 platform only. When switched on, the library uses a Zenity-based file dialog. When switched off (default), no zenity-based file dialog is used.
OPTION_FNFC_USES_KDIALOG	Meaningful for the Wayland/X11 platform only. When switched on, the library uses a kdialog-based file dialog if command 'kdialog' is available on the running system. When switched off (default), no kdialog-based file dialog is used.
OPTION_PRINTER_USES_GTK	When switched on (default), Fl_Printer runs the GTK printer dialog if the GTK library is available on the platform (linux/unix only). When switched off, the GTK printer dialog isn't used even if the GTK library is available.
OPTION_SHOW_SCALING	When switched on (default), the library shows in a transient yellow window the zoom factor value. When switched off, no such window gets displayed.

Enumerator

OPTION_SIMPLE_ZOOM_SHORTCUT	When switched on and when the keyboard in use has '+' in the shifted position of its key, pressing that key and ctrl triggers the zoom-in operation. When switched off (default), the zoom-in operation requires that also the shift key is pressed. Under macOS, this option has no effect because the OS itself generates = followed by + when pressing and the '= +' key without pressing shift.
OPTION_LAST	For internal use only.

11.7.3 Member Function Documentation

11.7.3.1 `abi_check()`

```
static int Fl::abi_check (
    const int val = FL_ABI_VERSION) [inline], [static]
```

Returns whether the runtime library ABI version is correct.

This enables you to check the ABI version of the linked FLTK library at runtime.

Returns 1 (true) if the compiled ABI version (in the header files) and the linked library ABI version (used at runtime) are the same, 0 (false) otherwise.

Argument `val` can be used to query a particular library ABI version. Use for instance 10303 to query if the runtime library is compatible with FLTK ABI version 1.3.3. This is rarely useful.

The default `val` argument is `FL_ABI_VERSION`, which checks the version defined at configure time (i.e. in the header files at program compilation time) against the linked library version used at runtime. This is particularly useful if you linked with a shared object library, but it also concerns static linking.

See also

[Fl::abi_version\(\)](#)

11.7.3.2 `abi_version()`

```
int Fl::abi_version () [static]
```

Returns the compiled-in value of the `FL_ABI_VERSION` constant.

This is useful for checking the version of a shared library.

11.7.3.3 `add_check()`

```
void Fl::add_check (
    Fl_Timeout_Handler cb,
    void * argp = 0) [static]
```

FLTK will call this callback just before it flushes the display and waits for events.

This is different than an idle callback because it is only called once, then FLTK calls the system and tells it not to return until an event happens.

This can be used by code that wants to monitor the application's state, such as to keep a display up to date. The advantage of using a check callback is that it is called only when no events are pending. If events are coming in quickly, whole blocks of them will be processed before this is called once. This can save significant time and avoid the application falling behind the events.

Sample code:

```
bool state_changed; // anything that changes the display turns this on

void callback(void*) {
    if (!state_changed) return;
    state_changed = false;
    do_expensive_calculation();
    widget->redraw();
}

main() {
    Fl::add_check(callback);
    return Fl::run();
}
```

11.7.3.4 `add_fd()` [1/2]

```
void Fl::add_fd (
    int fd,
    Fl_FD_Handler cb,
    void * d = 0) [static]
```

Adds file descriptor `fd` to listen to.

See [Fl::add_fd\(int fd, int when, Fl_FD_Handler cb, void* = 0\)](#) for details

11.7.3.5 `add_fd()` [2/2]

```
void Fl::add_fd (
    int fd,
    int when,
    Fl_FD_Handler cb,
    void * d = 0) [static]
```

Adds file descriptor `fd` to listen to.

When the `fd` becomes ready for reading [Fl::wait\(\)](#) will call the callback and then return. The callback is passed the `fd` and the arbitrary `void*` argument.

This version takes a `when` bitfield, with the bits `FL_READ`, `FL_WRITE`, and `FL_EXCEPT` defined, to indicate when the callback should be done.

There can only be one callback of each type for a file descriptor. [Fl::remove_fd\(\)](#) gets rid of *all* the callbacks for a given file descriptor.

Under UNIX/Linux/macOS *any* file descriptor can be monitored (files, devices, pipes, sockets, etc.). Due to limitations in Microsoft Windows, Windows applications can only monitor sockets.

Under macOS, [Fl::add_fd\(\)](#) opens the display if that's not been done before.

11.7.3.6 `add_idle()`

```
void Fl::add_idle (
    Fl_Idle_Handler cb,
    void * data = 0) [static]
```

Adds a callback function that is called every time by [Fl::wait\(\)](#) and also makes it act as though the timeout is zero (this makes [Fl::wait\(\)](#) return immediately, so if it is in a loop it is called repeatedly, and thus the idle function is called repeatedly).

The idle function can be used to get background processing done.

You can have multiple idle callbacks. To remove an idle callback use [Fl::remove_idle\(\)](#).

[Fl::wait\(\)](#) and [Fl::check\(\)](#) call idle callbacks, but [Fl::ready\(\)](#) does not.

The idle callback can call any FLTK functions, including [Fl::wait\(\)](#), [Fl::check\(\)](#), and [Fl::ready\(\)](#).

FLTK will not recursively call the idle callback.

11.7.3.7 `add_timeout()`

```
void Fl::add_timeout (
    double time,
    Fl_Timeout_Handler cb,
    void * data = 0) [static]
```

Adds a one-shot timeout callback.

The callback function `cb` will be called by [Fl::wait\(\)](#) at `time` seconds after this function is called. The callback function must have the signature [Fl_Timeout_Handler](#). The optional `data` argument is passed to the callback (default: `NULL`).

The timer is removed from the timer queue before the callback function is called. It is safe to reschedule the timeout inside the callback function.

You can have multiple timeout callbacks, even the same timeout callback with different timeout values and/or different `data` values. They are all considered different timer objects.

To remove a timeout while it is active (pending) use [Fl::remove_timeout\(\)](#).

If you need more accurate, repeated timeouts, use [Fl::repeat_timeout\(\)](#) to reschedule the subsequent timeouts. Please see [Fl::repeat_timeout\(\)](#) for an example.

Since version 1.4, a timeout can be started from a child thread under the condition that the call to [Fl::add_timeout](#) is wrapped in [Fl::lock\(\)](#) and [Fl::unlock\(\)](#).

Parameters

in	<i>time</i>	delta time in seconds until the timer expires
in	<i>cb</i>	callback function
in	<i>data</i>	optional user data (default: NULL)

See also

[Fl_Timeout_Handler](#)

[Fl::repeat_timeout\(double time, Fl_Timeout_Handler cb, void *data\)](#)

[Fl::remove_timeout\(Fl_Timeout_Handler cb, void *data\)](#)

[Fl::has_timeout\(Fl_Timeout_Handler cb, void *data\)](#)

11.7.3.8 api_version()

```
int Fl::api_version () [static]
```

Returns the compiled-in value of the FL_API_VERSION constant.

This is useful for checking the version of a shared library.

11.7.3.9 arg()

```
int Fl::arg (
    int argc,
    char ** argv,
    int & i) [static]
```

Parse a single switch from *argv*, starting at word *i*.

Returns the number of words eaten (1 or 2, or 0 if it is not recognized) and adds the same value to *i*.

This is the default argument handler used internally by [Fl::args\(...\)](#), but you can use this function if you prefer to step through the standard FLTK switches yourself.

All standard FLTK switches except -bg2 may be abbreviated to just one letter and case is ignored:

- -bg color or -background color
Sets the background color using [Fl::background\(\)](#).
- -bg2 color or -background2 color
Sets the secondary background color using [Fl::background2\(\)](#).
- -display host:n.n
Sets the X display to use; this option is silently ignored under Windows and MacOS.
- -dnd and -nodnd
Enables or disables drag and drop text operations using [Fl::dnd_text_ops\(\)](#).
- -fg color or -foreground color
Sets the foreground color using [Fl::foreground\(\)](#).
- -geometry WxH+X+Y
Sets the initial window position and size according to the standard X geometry string.
- -iconic
Iconifies the window using [Fl_Window::iconize\(\)](#).
- -kbd and -nokbd
Enables or disables visible keyboard focus for non-text widgets using [Fl::visible_focus\(\)](#).

- `-name string`
Sets the window class using `Fl_Window::xclass()`.
- `-scheme string`
Sets the widget scheme using `Fl::scheme()`.
- `-title string`
Sets the window title using `Fl_Window::label()`.
- `-tooltips` and `-notooltips`
Enables or disables tooltips using `Fl_Tooltip::enable()`.

Color values are commonly given as three digit or six digit hex numbers.

- The order of fg, bg, and bg2 in the command line does not matter
- There is no way at the moment to set the selection color.
- Setting the bg2 color also changes the fg color to have sufficient contrast
- Explicitly setting fg color overrides the bg2/contrast constraint.
- Setting the bg color will update the color lookup table for the gray ramp, so color index values can stay the same for all apps, it's just mapped to different RGB values.
- The calculation of the gray ramp is only based on the bg color, so there is no way at the moment to create an inverted (dark mode) ramp.
- Consequently, setting bg to black creates a an all-black ramp, setting a somewhat dark bg color creates a extremely dark ramp.
- Setting the bg has no influence on bg2 or fg.

If your program requires other switches in addition to the standard FLTK options, you will need to pass your own argument handler to `Fl::args(int,char**,int&,Fl_Args_Handler)` explicitly.

See also

`fl_parse_color(const char* p, uchar& r, uchar& g, uchar& b)` to see how color values can be defined

11.7.3.10 args() [1/2]

```
void Fl::args (
    int argc,
    char ** argv) [static]
```

Parse all command line switches matching standard FLTK options only.

It parses all the switches, and if any are not recognized it calls `Fl::abort(Fl::help)`, i.e. unlike the long form, an unrecognized switch generates an error message and causes the program to exit.

11.7.3.11 args() [2/2]

```
int Fl::args (
    int argc,
    char ** argv,
    int & i,
    Fl_Args_Handler cb = 0) [static]
```

Parse command line switches using the `cb` argument handler.

Returns 0 on error, or the number of words processed.

FLTK provides this as an *entirely optional* command line switch parser. You don't have to call it if you don't want to. Everything it can do can be done with other calls to FLTK.

To use the switch parser, call `Fl::args(...)` near the start of your program. This does **not** open the display, instead switches that need the display open are stashed into static variables. Then you **must** display your first window by calling `window->show(argc,argv)`, which will do anything stored in the static variables.

Providing an argument handler callback `cb` lets you define your own switches. It is called with the same `argc` and `argv`, and with `i` set to the index of the switch to be processed. The `cb` handler should return zero if the switch is unrecognized, and not change `i`. It should return non-zero to indicate the number of words processed if the switch is recognized, i.e. 1 for just the switch, and more than 1 for the switch plus associated parameters. `i` should be incremented by the same amount.

The `cb` handler is called **before** any other tests, so *you can also override any standard FLTK switch* (this is why FLTK can use very short switches instead of the long ones all other toolkits force you to use). See [Fl::arg\(\)](#) for descriptions of the standard switches.

On return `i` is set to the index of the first non-switch. This is either:

- The first word that does not start with '-'.
- The word '-' (used by many programs to name stdin as a file)
- The first unrecognized switch (return value is 0).
- `argc`

The return value is `i` unless an unrecognized switch is found, in which case it is zero. If your program takes no arguments other than switches you should produce an error if the return value is less than `argc`.

A usage string is displayed if [Fl::args\(\)](#) detects an invalid argument on the command-line. You can change the message by setting the [Fl::help](#) pointer.

A very simple command line parser can be found in `examples/howto-parse-args.cxx`

The simpler [Fl::args\(int argc, char **argv\)](#) form is useful if your program does not have command line switches of its own.

11.7.3.12 args_to_utf8()

```
int Fl::args_to_utf8 (
    int argc,
    char **& argv) [static]
```

Convert Windows commandline arguments to UTF-8.

Note

This function does nothing on other (non-Windows) platforms, hence you may call it on all platforms or only on Windows by using platform specific code like `'#ifdef _WIN32'` etc. - it's your choice. Calling it on other platforms returns quickly w/o wasting much CPU time.

This function *must be called on Windows platforms* in `main()` before the array `argv` is used if your program uses any commandline argument strings (these should be UTF-8 encoded). This applies also to standard FLTK commandline arguments like `"-name"` (class name) and `"-title"` (window title in the title bar).

Unfortunately Windows **neither** provides commandline arguments in UTF-8 encoding **nor** as Windows "Wide Character" strings in the standard `main()` and/or the Windows specific `WinMain()` function.

On Windows platforms (no matter which build system) this function calls a Windows specific function to retrieve commandline arguments as Windows "Wide Character" strings, converts these strings to an internally allocated buffer (or multiple buffers) and returns the result in `argv`. For implementation details please refer to the source code; however these details may be changed in the future.

Note that `argv` is provided by reference so it can be overwritten.

In the recommended simple form the function overwrites the variable `argv` and allocates a new array of strings pointed to by `argv`. You may use this form on all platforms and it is as simple as adding one line to old programs to make them work with international (UTF-8) commandline arguments.

```
int main(int argc, char **argv) {
    Fl::args_to_utf8(argc, argv); // add this line
    // ... use argc and argv, e.g. for commandline parsing
    window->show(argc, argv);
    return Fl::run();
}
```

For an example see 'examples/howto-parse-args.cxx' in the FLTK sources.

If you want to retain the original `argc` and `argv` variables the following slightly longer and more complicated code works as well on all platforms.

```
int main(int argc, char **argv) {
    char **argvn = argv; // must copy argv to work on all platforms
    int argcn = Fl::args_to_utf8(argc, argvn);
```

```
// ... use argc and argvn, e.g. for commandline parsing
window->show(argc, argvn);
return Fl::run();
}
```

Parameters

in	<i>argc</i>	used only on non-Windows platforms
out	<i>argv</i>	modified only on Windows platforms

Returns

argument count (always the same as *argc*)

Since

1.4.0

11.7.3.13 background()

```
void Fl::background (
    uchar r,
    uchar g,
    uchar b) [static]
```

Changes `fl_color(FL_BACKGROUND_COLOR)` to the given color, and changes the gray ramp from 32 to 56 to black to white.

These are the colors used as backgrounds by almost all widgets and used to draw the edges of all the boxtypes.

11.7.3.14 background2()

```
void Fl::background2 (
    uchar r,
    uchar g,
    uchar b) [static]
```

Changes the alternative background color.

This color is used as a background by [Fl_Input](#) and other text widgets.

This call may change `fl_color(FL_FOREGROUND_COLOR)` if it does not provide sufficient contrast to `FL_BACKGROUND2_COLOR`.

11.7.3.15 box_border_radius_max() [1/2]

```
static int Fl::box_border_radius_max () [inline], [static]
```

Get the maximum border radius of all "rounded" boxtypes in pixels.

Since

1.4.0

11.7.3.16 box_border_radius_max() [2/2]

```
static void Fl::box_border_radius_max (
    int R) [inline], [static]
```

Set the maximum border radius of all "rounded" boxtypes in pixels.

Must be at least 5, default = 15.

Note

This does **not** apply to the "round" boxtypes which have really round sides (i.e. composed of half circles) as opposed to "rounded" boxtypes that have only rounded corners with a straight border between corners.

The box border radius of "rounded" boxtypes is typically calculated as about 2/5 of the box height or width, whichever is smaller. The upper limit can be set by this method for all "rounded" boxtypes.

Since

1.4.0

11.7.3.17 box_color()

```
Fl_Color Fl::box_color (
    Fl_Color c) [static]
```

Gets the drawing color to be used for the background of a box.

This method is only useful inside box drawing code. It returns the color to be used, either `fl_inactive(c)` if the widget is `!active_r()` or `c` otherwise.

11.7.3.18 box_dh()

```
int Fl::box_dh (
    Fl_Boxtype t) [static]
```

Returns the height offset for the given boxtype.

See also

[box_dy\(\)](#).

11.7.3.19 box_dw()

```
int Fl::box_dw (
    Fl_Boxtype t) [static]
```

Returns the width offset for the given boxtype.

See also

[box_dy\(\)](#).

11.7.3.20 box_dx()

```
int Fl::box_dx (
    Fl_Boxtype t) [static]
```

Returns the X offset for the given boxtype.

See also

[box_dy\(\)](#)

11.7.3.21 box_dy()

```
int Fl::box_dy (
    Fl_Boxtype t) [static]
```

Returns the Y offset for the given boxtype.

These functions return the offset values necessary for a given boxtype, useful for computing the area inside a box's borders, to prevent overdrawing the borders.

For instance, in the case of a boxtype like `FL_DOWN_BOX` where the border width might be 2 pixels all around, the above functions would return 2, 2, 4, and 4 for `box_dx`, `box_dy`, `box_dw`, and `box_dh` respectively.

An example to compute the area inside a widget's box():

```
int X = yourwidget->x() + Fl::box_dx(yourwidget->box());
int Y = yourwidget->y() + Fl::box_dy(yourwidget->box());
int W = yourwidget->w() - Fl::box_dw(yourwidget->box());
int H = yourwidget->h() - Fl::box_dh(yourwidget->box());
```

These functions are mainly useful in the `draw()` code for deriving custom widgets, where one wants to avoid drawing over the widget's own border box().

11.7.3.22 box_shadow_width() [1/2]

```
static int Fl::box_shadow_width () [inline], [static]
```

Get the box shadow width of all "shadow" boxtypes in pixels.

Since

1.4.0

11.7.3.23 box_shadow_width() [2/2]

```
static void Fl::box_shadow_width (
    int W) [inline], [static]
```

Set the box shadow width of all "shadow" boxtypes in pixels.

Must be at least 1, default = 3. There is no upper limit.

Since

1.4.0

11.7.3.24 check()

```
int Fl::check () [static]
```

Same as Fl::wait(0).

Calling this during a big calculation will keep the screen up to date and the interface responsive:

```
while (!calculation_done()) {
    calculate();
    Fl::check();
    if (user_hit_abort_button()) break;
}
```

This returns non-zero if any windows are displayed, and 0 if no windows are displayed (this is likely to change in future versions of FLTK).

11.7.3.25 display()

```
void Fl::display (
    const char * d) [static]
```

Sets the X or Wayland display to use for all windows.

This sets the environment variable \$DISPLAY or \$WAYLAND_DISPLAY to the passed string, so this only works before you show() the first window or otherwise open the display.

This does nothing on other platforms.

11.7.3.26 dnd_text_ops() [1/2]

```
static int Fl::dnd_text_ops () [inline], [static]
```

Gets whether drag and drop text operations are supported.

This returns whether selected text can be dragged from text fields or dragged within a text field as a cut/paste shortcut.

11.7.3.27 dnd_text_ops() [2/2]

```
static void Fl::dnd_text_ops (
    int v) [inline], [static]
```

Sets whether drag and drop text operations are supported.

This specifically affects whether selected text can be dragged from text fields or dragged within a text field as a cut/paste shortcut.

11.7.3.28 draw_box_active()

```
int Fl::draw_box_active () [static]
```

Determines if the currently drawn box is active or inactive.

If inactive, the box color should be changed to the inactive color.

See also

[Fl::box_color\(Fl_Color c\)](#)

11.7.3.29 draw_GL_text_with_textures() [1/2]

```
static int Fl::draw_GL_text_with_textures () [inline], [static]
```

returns whether whether OpenGL uses textures to draw all text.

Default is yes.

See also

[draw_GL_text_with_textures\(int val\)](#)

Version

1.4.0

11.7.3.30 draw_GL_text_with_textures() [2/2]

```
static void Fl::draw_GL_text_with_textures (
    int val) [inline], [static]
```

sets whether OpenGL uses textures to draw all text.

By default, FLTK draws OpenGL text using textures, if the necessary hardware support is available. Call `Fl::draw_GL_text_with_textures(0)` once in your program before the first call to [gl_font\(\)](#) to have FLTK draw instead OpenGL text using a legacy, platform-dependent procedure. It's recommended not to deactivate textures under the MacOS platform because the MacOS legacy procedure is extremely rudimentary.

Parameters

<i>val</i>	use 0 to prevent FLTK from drawing GL text with textures

See also

[gl_texture_pile_height\(int max\)](#)

Version

1.4.0

11.7.3.31 flush()

```
void Fl::flush () [static]
```

Causes all the windows that need it to be redrawn and graphics forced out through the pipes.

This is what [wait\(\)](#) does before looking for events.

Note: in multi-threaded applications you should only call [Fl::flush\(\)](#) from the main thread. If a child thread needs to trigger a redraw event, it should instead call [Fl::awake\(\)](#) to get the main thread to process the event queue.

11.7.3.32 get_system_colors()

```
void Fl::get_system_colors () [static]
```

Read the user preference colors from the system and use them to call [Fl::foreground\(\)](#), [Fl::background\(\)](#), and [Fl::background2\(\)](#).

This is done by `Fl_Window::show(argc,argv)` before applying the `-fg` and `-bg` switches.

On X this reads some common values from the Xdefaults database. KDE users can set these values by running the "krdb" program, and newer versions of KDE set this automatically if you check the "apply style to other X programs" switch in their control panel.

11.7.3.33 gl_visual()

```
static int Fl::gl_visual (
    int ,
    int * alist = 0) [static]
```

This does the same thing as [Fl::visual\(int\)](#) but also requires OpenGL drawing to work.

This *must* be done if you want to draw in normal windows with OpenGL with [gl_start\(\)](#) and [gl_end\(\)](#). It may be useful to call this so your X windows use the same visual as an [Fl_Gl_Window](#), which on some servers will reduce colormap flashing.

See [Fl_Gl_Window](#) for a list of additional values for the argument.

11.7.3.34 has_timeout()

```
int Fl::has_timeout (
    Fl_Timeout_Handler cb,
    void * data = 0) [static]
```

Returns true if the timeout exists and has not been called yet.

Both arguments *cb* and *data* must match with at least one timer in the queue of active timers to return true (1).

Note

It is a known inconsistency that [Fl::has_timeout\(\)](#) does not use the *data* argument as a wildcard (match all) if it is zero (NULL) which [Fl::remove_timeout\(\)](#) does. This is so for backwards compatibility with FLTK 1.3.x. Therefore using 0 (zero, NULL) as the timeout *data* value is discouraged unless you're sure that you don't need to use [Fl::has_timeout\(callback, \(void *\)0\)](#); or [Fl::remove_timeout\(callback, \(void *\)0\)](#);

Parameters

in	<i>cb</i>	Timer callback
in	<i>data</i>	User data

Returns

whether the timer was found in the queue

Return values

0	not found
1	found

11.7.3.35 hide_all_windows()

```
void Fl::hide_all_windows () [static]
```

Hide all visible windows to make FLTK leave [Fl::run\(\)](#).

[Fl::run\(\)](#) will run as long as there are visible windows. Call [Fl::hide_all_windows\(\)](#) to hide (close) all currently shown (visible) windows, effectively terminating the [Fl::run\(\)](#) loop.

See also

[Fl::run\(\)](#)

Since

1.4.0

11.7.3.36 is_scheme()

```
static int Fl::is_scheme (
    const char * name) [inline], [static]
```

Returns whether the current scheme is the given name.

This is a fast inline convenience function to support scheme-specific code in widgets, e.g. in their draw() methods, if required.

Use a valid scheme name, not `NULL` (although `NULL` is allowed, this is not a useful argument - see below).

If `Fl::scheme()` has not been set or has been set to the default scheme ("none" or "base"), then this will always return 0 regardless of the argument, because `Fl::scheme()` is `NULL` in this case.

Note

The stored scheme name is always lowercase, and this method will do a case-sensitive compare, so you **must** provide a lowercase string to return the correct value. This is intentional for performance reasons.

Example:

```
if (Fl::is_scheme("gtk+")) { your_code_here(); }
```

Parameters

in	<i>name</i>	lowercase string of requested scheme name.

Returns

1 if the given scheme is active, 0 otherwise.

See also

[Fl::scheme\(const char *name\)](#)

11.7.3.37 menu_linespacing() [1/2]

```
int Fl::menu_linespacing () [static]
```

Gets the default line spacing used by menus.

Returns

The default line spacing, in pixels.

Since

1.4.0

11.7.3.38 menu_linespacing() [2/2]

```
void Fl::menu_linespacing (
    int H) [static]
```

Sets the default line spacing used by menus.

Default is 4.

Parameters

in	<i>H</i>	The new default line spacing between menu items, in pixels.

Since

1.4.0

11.7.3.39 now()

```
Fl_Timestamp Fl::now (
    double offset = 0) [static]
```

Set a time stamp at this point in time with optional signed offset in seconds.

The time stamp is an opaque type and does not represent the time of day or some time and date in the calendar. It is used with [Fl::seconds_between\(\)](#) and [Fl::seconds_since\(\)](#) to measure elapsed time.

```
Fl_Timestamp start = Fl::now();
// do something
double s = Fl::seconds_since(start);
printf("That operation took %g seconds\n", s);
```

Depending on the system the resolution may be milliseconds or microseconds. Under certain conditions (particularly on Windows) the value in member `sec` may wrap around and does not represent a real time (maybe runtime of the system). Function [seconds_since\(\)](#) below uses this to subtract two timestamps which is always a correct delta time with milliseconds or microseconds resolution.

Parameters

<i>offset</i>	optional signed offset in seconds added to the current time

Returns

this moment in time offset by `offset` as an opaque time stamp

Todo [Fl::system_driver\(\)->gettime\(\)](#) was implemented for the Forms library and has a limited resolution (on Windows: milliseconds). On POSIX platforms it uses `gettimeofday()` with microsecond resolution. A new function could use a better resolution on Windows with its multimedia timers which requires a new dependency: `winmm.lib` (dll). This could be a future improvement, maybe set as a build option or generally (requires Win95 or 98?).

See also

[Fl::seconds_since\(Fl_Timestamp& then\)](#)
[Fl::seconds_between\(Fl_Timestamp& back, Fl_Timestamp& further_back\)](#)
[Fl::ticks_since\(Fl_Timestamp& then\)](#)
[Fl::ticks_between\(Fl_Timestamp& back, Fl_Timestamp& further_back\)](#)

11.7.3.40 option() [1/2]

```
bool Fl::option (
    Fl_Option opt) [static]
```

FLTK library options management.

Options provide a way for the user to modify the behavior of an FLTK application. For example, clearing the `OPTION_SHOW_TOOLTIPS` will disable tooltips for all FLTK applications.

Options are set by the user or the administrator on user or machine level. In 1.3, FLUID has an Options dialog for that. In 1.4, there is an app named `fltk-options` that can be used from the command line or as a GUI tool. The machine level setting is read first, and the user setting can override the machine setting.

This function is used throughout FLTK to quickly query the user's wishes. There are options for using a native file chooser instead of the FLTK one wherever possible, disabling tooltips, disabling visible focus, disabling FLTK file chooser preview, etc. .

See [Fl::Fl_Option](#) for a list of available options.

Example:

```
if ( Fl::option(Fl::OPTION_ARROW_FOCUS) )
{ ..on.. }
else
{ ..off.. }
```

Note

Since FLTK 1.4.0 options can be managed with the `fltk-options` program. In FLTK 1.3.x options can be set in FLUID.

Parameters

<i>opt</i>	which option

Returns

true or false

See also

enum [Fl::Fl_Option](#)

[Fl::option\(Fl_Option, bool\)](#)

fltk-options application in command line and GUI mode

Since

FLTK 1.3.0

11.7.3.41 option() [2/2]

```
void Fl::option (
    Fl_Option opt,
    bool val) [static]
```

Override an option while the application is running.

Apps can override the machine settings and the user settings by calling `Fl::option(option, bool)`. The override takes effect immediately for this option for all widgets in the app for the life time of the app.

The override is not saved anywhere, and relaunching the app will restore the old settings.

Example:

```
Fl::option(Fl::OPTION_ARROW_FOCUS, true);    // on
Fl::option(Fl::OPTION_ARROW_FOCUS, false);   // off
```

Parameters

<i>opt</i>	which option
<i>val</i>	set to true or false

See also

enum [Fl::Fl_Option](#)

bool [Fl::option\(Fl_Option\)](#)

11.7.3.42 own_colormap()

```
void Fl::own_colormap () [static]
```

Makes FLTK use its [own colormap](#).

This may make FLTK display better and will reduce conflicts with other programs that want lots of colors. However the colors may flash as you move the cursor between windows.

This does nothing if the current visual is not colormapped.

11.7.3.43 program_should_quit() [1/2]

```
static int Fl::program_should_quit () [inline], [static]
```

Returns non-zero when a request for program termination was received and accepted.

On the MacOS platform, the "Quit xxx" item of the application menu is such a request, that is considered accepted when all windows are closed. On other platforms, this function returns 0 until `Fl::program_should_quit(1)` is called.

Version

1.4.0

11.7.3.44 `program_should_quit()` [2/2]

```
static void Fl::program_should_quit (
    int should_i) [inline], [static]
```

Indicate to the FLTK library whether a program termination request was received and accepted.

A program may set this to 1, for example, while performing a platform-independent command asking the program to cleanly terminate, similarly to the "Quit xxx" item of the application menu under MacOS.

Version

1.4.0

11.7.3.45 `readqueue()`

```
Fl_Widget * Fl::readqueue () [static]
```

Reads the default callback queue and returns the first widget.

All `Fl_Widget`s that don't have a callback defined use the default callback `static Fl_Widget::default_callback()` that puts a pointer to the widget in a queue. This method reads the oldest widget out of this queue.

The queue (FIFO) is limited (currently 20 items). If the queue overflows, the oldest entry (`Fl_Widget *`) is discarded. Relying on the default callback and reading the callback queue with `Fl::readqueue()` is not recommended. If you need a callback, you should set one with `Fl_Widget::callback(Fl_Callback *cb, void *data)` or one of its variants.

See also

`Fl_Widget::callback()`
`Fl_Widget::callback(Fl_Callback *cb, void *data)`
`Fl_Widget::default_callback()`

11.7.3.46 `ready()`

```
int Fl::ready () [static]
```

This is similar to `Fl::check()` except this does *not* call `Fl::flush()` or any callbacks, which is useful if your program is in a state where such callbacks are illegal.

This returns true if `Fl::check()` would do anything (it will continue to return true until you call `Fl::check()` or `Fl::wait()`).

```
while (!calculation_done()) {
    calculate();
    if (Fl::ready()) {
        do_expensive_cleanup();
        Fl::check();
        if (user_hit_abort_button()) break;
    }
}
```

11.7.3.47 `release()`

```
static void Fl::release () [inline], [static]
```

Releases the current grabbed window, equals `grab(0)`.

Deprecated Use `Fl::grab(0)` instead.

See also

`grab(Fl_Window*)`

11.7.3.48 `reload_scheme()`

```
int Fl::reload_scheme () [static]
```

Called internally when setting a new scheme according to scheme name.

Loads or reloads the current scheme selection.

Returns

Always 1 (this may change in the future)

See void `Fl::scheme(const char *name)`

11.7.3.49 remove_check()

```
void Fl::remove_check (
    Fl_Timeout_Handler cb,
    void * argp = 0) [static]
```

Removes a check callback.

It is harmless to remove a check callback that no longer exists.

11.7.3.50 remove_next_timeout()

```
int Fl::remove_next_timeout (
    Fl_Timeout_Handler cb,
    void * data = 0,
    void ** data_return = 0) [static]
```

Remove the next matching timeout callback and return its `data` pointer.

This method removes only the next matching timeout and returns in `data_return` (if non-NULL) the `data` member given when the timeout was scheduled.

This method is useful if you remove a timeout before it is scheduled and you need to get and use its data value, for instance to `free()` or delete the data associated with the timeout.

This method returns non-zero if a matching timeout was found and zero if no timeout matched the request.

If the return value is $N > 1$ then there are $N - 1$ more matching timeouts pending.

If you need to remove all timeouts with a particular callback `cb` you must repeat this call until it returns 1 (all timeouts removed) or zero (no matching timeout), whichever occurs first.

Parameters

in	<i>cb</i>	Timer callback to be removed (must match)
in	<i>data</i>	Wildcard if NULL, must match otherwise
in, out	<i>data_return</i>	Pointer to (void *) to receive the data value

Returns

non-zero if a timer was found and removed

Return values

0	no matching timer was found
1	the last matching timeout was found and removed
$N > 1$	a matching timeout was removed and there are (N - 1) matching timeouts pending

See also

[Fl::remove_timeout\(Fl_Timeout_Handler cb, void *data\)](#)

Since

1.4.0

11.7.3.51 remove_timeout()

```
void Fl::remove_timeout (
    Fl_Timeout_Handler cb,
    void * data = 0) [static]
```

Remove one or more matching timeout callbacks from the timer queue.

This method removes **all** matching timeouts, not just the first one.

If the `data` argument is `NULL` (the default!) only the callback `cb` must match, i.e. all timer entries with this callback are removed.

It is harmless to remove a timeout callback that no longer exists.

If you want to remove only the next matching timeout you can use [Fl::remove_next_timeout\(Fl_Timeout_Handler cb, void *data, void *\)](#) (available since FLTK 1.4.0).

Parameters

in	<i>cb</i>	Timer callback to be removed (must match)
in	<i>data</i>	Wildcard if <code>NULL</code> (default), must match otherwise

See also

[Fl::remove_next_timeout\(Fl_Timeout_Handler cb, void *data, void **data_return\)](#)

11.7.3.52 repeat_timeout()

```
void Fl::repeat_timeout (
    double time,
    Fl_Timeout_Handler cb,
    void * data = 0) [static]
```

Repeats a timeout callback from the expiration of the previous timeout, allowing for more accurate timing.

You should call this method only inside a timeout callback of the same or a logically related timer from whose expiration time the new timeout shall be scheduled. Otherwise the timing accuracy can't be improved and the exact behavior is undefined.

If you call this outside a timeout callback the behavior is the same as [Fl::add_timeout\(\)](#).

Example: The following code will print "TICK" each second on stdout with a fair degree of accuracy:

```
#include <FL/Fl.H>
#include <FL/Fl_Window.H>
#include <stdio.h>

void callback(void *) {
    printf("TICK\n");
    Fl::repeat_timeout(1.0, callback); // retrigger timeout
}

int main() {
    Fl_Window win(100, 100);
    win.show();
    Fl::add_timeout(1.0, callback); // set up first timeout
    return Fl::run();
}
```

Parameters

in	<i>time</i>	delta time in seconds until the timer expires
in	<i>cb</i>	callback function
in	<i>data</i>	optional user data (default: <code>NULL</code>)

11.7.3.53 run()

```
int Fl::run () [static]
```

Calls [Fl::wait\(\)](#) repeatedly as long as any windows are displayed.

When all the windows are closed it returns zero (supposedly it would return non-zero on any errors, but FLTK calls exit directly for these). A normal program will end `main()` with return [Fl::run\(\)](#);

Note

[Fl::run\(\)](#) and [Fl::wait\(\)](#) (but not [Fl::wait\(double\)](#)) both return when all FLTK windows are closed. Therefore, a MacOS FLTK application possessing [Fl_Sys_Menu_Bar](#) items able to create new windows and expected to keep running without any open window cannot use these two functions. One solution is to run the event loop as follows:

```
while (!Fl::program_should_quit()) Fl::wait(1e20);
```

11.7.3.54 `scheme()`

```
int Fl::scheme (
    const char * s) [static]
```

Sets the current widget scheme.

NULL will use the scheme defined in the FLTK_SCHEME environment variable or the scheme resource under X11. Otherwise, any of the following schemes can be used:

- "none" - This is the default look-n-feel which resembles old Windows (95/98/Me/NT/2000) and old GTK/KDE
- "base" - This is an alias for "none"
- "plastic" - This scheme is inspired by the Aqua user interface on macOS
- "gtk+" - This scheme is inspired by the Red Hat Bluecurve theme
- "gleam" - This scheme is inspired by the Clearlooks Glossy scheme. (Colin Jones and Edmanuel Torres).
- "oxy" - This is a subset of Dmitrij K's oxy scheme (STR 2675, 3477)

If the given scheme name is unknown, the default scheme will be used.

Setting the scheme (name) is case insensitive, but the stored scheme name will always be lowercase and `Fl::scheme()` will return this lowercase name or NULL if no scheme or the default scheme ("none" or "base") was set.

Parameters

<code>in</code>	<code>s</code>	Scheme name or NULL

Return values

<code>0</code>	if the scheme has not been set or is the default scheme
<code>1</code>	if a scheme other than "none"/"base" was set

See also

`Fl::scheme()` to get the name of the current [scheme](#)

`Fl::is_scheme(const char*)` to test if the specified [scheme](#) is set

11.7.3.55 `scrollbar_size()` [1/2]

```
int Fl::scrollbar_size () [static]
```

Gets the default scrollbar size used by [Fl_Browser_](#), [Fl_Help_View](#), [Fl_Scroll](#), and [Fl_Text_Display](#) widgets.

Returns

The default size for widget scrollbars, in pixels.

11.7.3.56 `scrollbar_size()` [2/2]

```
void Fl::scrollbar_size (
    int w) [static]
```

Sets the default scrollbar size that is used by the [Fl_Browser_](#), [Fl_Help_View](#), [Fl_Scroll](#), and [Fl_Text_Display](#) widgets.

Parameters

<code>in</code>	<code>w</code>	The new default size for widget scrollbars, in pixels.

11.7.3.57 seconds_between()

```
double Fl::seconds_between (
    Fl_Timestamp & back,
    Fl_Timestamp & further_back) [static]
```

Return the time in seconds between two time stamps.

Parameters

in	<i>back</i>	a previously taken time stamp
in	<i>further_back</i>	an even earlier time stamp

Returns

elapsed seconds and fractions of a second

See also

[Fl::seconds_since\(Fl_Timestamp& then\)](#)

[Fl::now\(\)](#)

11.7.3.58 seconds_since()

```
double Fl::seconds_since (
    Fl_Timestamp & then) [static]
```

Return the time in seconds between now and a previously taken time stamp.

Parameters

in	<i>then</i>	a previously taken time stamp

Returns

elapsed seconds and fractions of a second

See also

[Fl::seconds_between\(Fl_Timestamp& back, Fl_Timestamp& further_back\)](#)

[Fl::now\(\)](#)

[Fl::distant_past\(\)](#)

11.7.3.59 set_box_color()

```
void Fl::set_box_color (
    Fl_Color c) [static]
```

Sets the drawing color for the box that is currently drawn.

This method sets the current drawing color [fl_color\(\)](#) depending on the widget's state to either *c* or [fl_inactive\(c\)](#).

It should be used whenever a box background is drawn in the box (type) drawing code instead of calling [fl_color\(Fl_Color bg\)](#) with the background color *bg*, usually [Fl_Widget::color\(\)](#).

This method is only useful inside box drawing code. Whenever a box is drawn with one of the standard box drawing methods, a static variable is set depending on the widget's current state - if the widget is [!active_r\(\)](#) then the internal variable is false (0), otherwise it is true (1). This is faster than calling [Fl_Widget::active_r\(\)](#) because the state is cached.

See also

[Fl::draw_box_active\(\)](#)

[Fl::box_color\(Fl_Color\)](#)

11.7.3.60 set_boxtype()

```
void Fl::set_boxtype (
    Fl_Boxtype t,
    Fl_Box_Draw_F * f,
    uchar dx,
    uchar dy,
    uchar dw,
    uchar dh,
    Fl_Box_Draw_Focus_F * ff = NULL) [static]
```

Sets the function to call to draw a specific box type.

Parameters

in	<i>t</i>	index of the box type between 0 (FL_NO_BOX) and up to and including FL_MAX_BOXTYPE
in	<i>f</i>	callback function that draws the box
in	<i>dx,dy</i>	top left frame width, distance in pixels to box contents
in	<i>dw,dh</i>	left plus right frame width, top plus bottom frame width
in	<i>ff</i>	optional callback that draws the box focus, defaults to a rectangle, inset by dx, dy, dw, dh

11.7.3.61 set_idle()

```
static void Fl::set_idle (
    Fl_Old_Idle_Handler cb) [inline], [static]
```

Sets an idle callback.

Deprecated This method is obsolete - use the [add_idle\(\)](#) method instead.

11.7.3.62 ticks_between()

```
long Fl::ticks_between (
    Fl_Timestamp & back,
    Fl_Timestamp & further_back) [static]
```

Return the time in ticks (60Hz) between two time stamps.

Parameters

in	<i>back</i>	a previously taken time stamp
in	<i>further_back</i>	an even earlier time stamp

Returns

elapsed ticks in 60th of a second

See also

[Fl::ticks_since\(Fl_Timestamp& then\)](#)

[Fl::now\(\)](#)

11.7.3.63 ticks_since()

```
long Fl::ticks_since (
    Fl_Timestamp & then) [static]
```

Return the time in ticks (60Hz) between now and a previously taken time stamp.

Ticks are a convenient way to time animations 'per frame'. Even though modern computers use all kinds of screen refresh rates, 60Hz is a very good base for animation that is typically shown in user interface graphics.

Parameters

<code>in</code>	<code>then</code>	a previously taken time stamp

Returns

elapsed ticks in 60th of a second

See also

[Fl::ticks_between\(Fl_Timestamp& back, Fl_Timestamp& further_back\)](#)

[Fl::now\(\)](#)

11.7.3.64 use_high_res_GL() [1/2]

```
static int Fl::use_high_res_GL () [inline], [static]
```

returns whether GL windows should be drawn at high resolution on Apple computers with retina displays. Default is no.

Version

1.3.4

11.7.3.65 use_high_res_GL() [2/2]

```
static void Fl::use_high_res_GL (
    int val) [inline], [static]
```

sets whether GL windows should be drawn at high resolution on Apple computers with retina displays

Version

1.3.4

11.7.3.66 version()

```
double Fl::version () [static]
```

Returns the compiled-in value of the FL_VERSION constant. This is useful for checking the version of a shared library.

Deprecated Use [int Fl::api_version\(\)](#) instead.

11.7.3.67 visible_focus() [1/2]

```
static int Fl::visible_focus () [inline], [static]
```

Gets or sets the visible keyboard focus on buttons and other non-text widgets. The default mode is to enable keyboard focus for all widgets.

11.7.3.68 visible_focus() [2/2]

```
static void Fl::visible_focus (
    int v) [inline], [static]
```

Gets or sets the visible keyboard focus on buttons and other non-text widgets. The default mode is to enable keyboard focus for all widgets.

11.7.3.69 visual()

```
int Fl::visual (
    int flags) [static]
```

Selects a visual so that your graphics are drawn correctly.

This is only allowed before you call `show()` on any windows. This does nothing if the default visual satisfies the capabilities, or if no visual satisfies the capabilities, or on systems that don't have such brain-dead notions.

Only the following combinations do anything useful:

- `Fl::visual(FL_RGB)`
Full/true color (if there are several depths FLTK chooses the largest). Do this if you use `fl_draw_image` for much better (non-dithered) output.
- `Fl::visual(FL_RGB8)`
Full color with at least 24 bits of color. `FL_RGB` will always pick this if available, but if not it will happily return a less-than-24 bit deep visual. This call fails if 24 bits are not available.

This returns true if the system has the capabilities by default or FLTK succeeded in turning them on. Your program will still work even if this returns false (it just won't look as good).

11.7.3.70 wait() [1/2]

```
int Fl::wait () [static]
```

Waits until "something happens" and then returns.

Call this repeatedly to "run" your program. You can also check what happened each time after this returns, which is quite useful for managing program state.

What this really does is call all idle callbacks, all elapsed timeouts, call [Fl::flush\(\)](#) to get the screen to update, and then wait some time (zero if there are idle callbacks, the shortest of all pending timeouts, or infinity), for any events from the user or any [Fl::add_fd\(\)](#) callbacks. It then handles the events and calls the callbacks and then returns.

Returns

non-zero if there are any visible windows - this may change in future versions of FLTK.

11.7.3.71 wait() [2/2]

```
double Fl::wait (
    double time_to_wait) [static]
```

Waits a maximum of `time_to_wait` seconds or until "something happens".

See [Fl::wait\(\)](#) for the description of operations performed when "something happens".

Returns

Always 1 on Windows. Otherwise, it is positive if an event or fd happens before the time elapsed. It is zero if nothing happens. It is negative if an error occurs (this will happen on X11 if a signal happens).

11.7.4 Member Data Documentation**11.7.4.1 help**

```
const char *const Fl::help = helpmsg+13 [static]
```

Usage string displayed if [Fl::args\(\)](#) detects an invalid argument.

This may be changed to point to customized text at run-time.

11.7.4.2 idle

```
void(* Fl::idle) () [static]
```

The currently executing idle callback function: DO NOT USE THIS DIRECTLY!

This is now used as part of a higher level system allowing multiple idle callback functions to be called.

See also

[add_idle\(\)](#), [remove_idle\(\)](#)

The documentation for this class was generated from the following files:

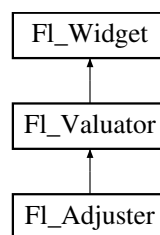
- [Fl.H](#)
- [Fl.cxx](#)
- [Fl_abort.cxx](#)
- [Fl_add_idle.cxx](#)
- [Fl_arg.cxx](#)
- [fl_boxttype.cxx](#)
- [Fl_Cairo.cxx](#)
- [fl_color.cxx](#)
- [Fl_compose.cxx](#)
- [Fl_display.cxx](#)
- [Fl_get_system_colors.cxx](#)
- [Fl_grab.cxx](#)
- [fl_labeltype.cxx](#)
- [Fl_lock.cxx](#)
- [Fl_own_colormap.cxx](#)
- [fl_set_font.cxx](#)
- [fl_shortcut.cxx](#)
- [Fl_Timeout.cxx](#)
- [Fl_visual.cxx](#)
- [Fl_Widget.cxx](#)
- [Fl_Window.cxx](#)
- [screen_xywh.cxx](#)

11.8 Fl_Adjuster Class Reference

The [Fl_Adjuster](#) widget was stolen from Prisms, and has proven to be very useful for values that need a large dynamic range.

```
#include <Fl_Adjuster.H>
```

Inheritance diagram for [Fl_Adjuster](#):



Public Member Functions

- [Fl_Adjuster](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [Fl_Adjuster](#) widget using the given position, size, and label string.
- int [soft](#) () const
If "soft" is turned on, the user is allowed to drag the value outside the range.
- void [soft](#) (int s)
If "soft" is turned on, the user is allowed to drag the value outside the range.

Public Member Functions inherited from FI_Valuator

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)
Sets the maximum value for the valuator.
- double **minimum** () const
Gets the minimum value for the valuator.
- void **minimum** (double a)
Sets the minimum value for the valuator.
- void **precision** (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void **range** (double a, double b)
Sets the minimum and maximum values for the valuator.
- double **round** (double)
Round the passed value to the nearest step increment.
- double **step** () const
Gets or sets the step value.
- void **step** (double a, int b)
See double FI_Valuator::step() const
- void **step** (double s)
See double FI_Valuator::step() const.
- void **step** (int a)
See double FI_Valuator::step() const
- double **value** () const
Gets the floating point(double) value.
- int **value** (double)
Sets the current value.
- ~FI_Valuator () **FL_OVERRIDE**
Destructor is accessible despite protected constructor.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- FI_Align **align** () const

- Gets the label alignment.*

 - void [align](#) ([FI_Align](#) alignment)
- Sets the label alignment.*

 - long [argument](#) () const
- Gets the current user data (long) argument that is passed to the callback function.*

 - void [argument](#) (long v)
- Sets the current user data (long) argument that is passed to the callback function.*

 - virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
- Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).*

 - virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).*

 - virtual [FI_Group](#) * [as_group](#) ()
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*

 - virtual [FI_Window](#) const * [as_window](#) ()
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*

 - virtual [FI_Window](#) const * [as_window](#) () const
- Sets the image to use as part of the widget label when in the inactive state.*

 - void [bind_deimage](#) ([FI_Image](#) *img)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*

 - void [bind_image](#) ([FI_Image](#) *img)
- Sets the image to use as part of the widget label when in the active state.*

 - void [bind_image](#) (int f)
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*

 - [FI_Boxtype](#) [box](#) () const
- Gets the box type of the widget.*

 - void [box](#) ([FI_Boxtype](#) new_box)
- Sets the box type for the widget.*

 - [FI_Callback_p](#) [callback](#) () const
- Gets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, void *p)
- Sets the current callback function and data for the widget.*

 - void [callback](#) ([FI_Callback0](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)
- Sets the current callback function for the widget.*

 - unsigned int [changed](#) () const
- Checks if the widget value changed since the last callback.*

 - void [clear_active](#) ()
- Marks the widget as inactive without sending events or changing focus.*

 - void [clear_changed](#) ()
- Marks the value of the widget as unchanged.*

 - void [clear_damage](#) (uchar c=0)
- Clears or sets the damage flags.*

 - void [clear_output](#) ()
- Sets a widget to accept input.*

 - void [clear_visible](#) ()

- Hides the widget.*
- void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*
- `FL_Color` `color` () const
- Gets the background color of the widget.*
- void `color` (`FL_Color` bg)
- Sets the background color of the widget.*
- void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*
- `FL_Color` `color2` () const
- For back compatibility only.*
- void `color2` (unsigned a)
- For back compatibility only.*
- int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*
- void `copy_label` (const char *new_label)
- Sets the current label.*
- void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*
- `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*
- void `damage` (`uchar` c)
- Sets the damage bits for the widget.*
- void `damage` (`uchar` c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int `damage_resize` (int, int, int, int)
- Internal use only.*
- void `deactivate` ()
- Deactivates the widget.*
- `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*
- const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with default arguments.*
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*
- void `draw_label` (int, int, int, int, `FL_Align`) const
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int `h` () const
- Gets the widget height.*
- virtual void `hide` ()
- Makes a widget invisible.*

- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FI_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FI_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FI_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FI_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FI_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FI_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FI_Color](#) c)
Sets the label color.
- [FI_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FI_Font](#) f)
Sets the font to use.
- [FI_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FI_Fonsize](#) pix)
Sets the font size in pixels.
- [FI_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FI_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

- Returns if a widget is used for output only.*

 - `Fl_Group * parent () const`

Returns a pointer to the parent widget.
- `void parent (Fl_Group *p)`

Internal use only - "for hacks only".
- `void position (int X, int Y)`

Repositions the window or widget.
- `void redraw ()`

Schedules the drawing of the widget.
- `void redraw_label ()`

Schedules the drawing of the label.
- `virtual void resize (int x, int y, int w, int h)`

Changes the size or position of the widget.
- `Fl_Color selection_color () const`

Gets the selection color.
- `void selection_color (Fl_Color a)`

Sets the selection color.
- `void set_active ()`

Marks the widget as active without sending events or changing focus.
- `void set_changed ()`

Marks the value of the widget as changed.
- `void set_output ()`

Sets a widget to output only.
- `void set_visible ()`

Makes the widget visible.
- `void set_visible_focus ()`

Enables keyboard focus navigation with this widget.
- `int shortcut_label () const`

Returns whether the widget's label uses '&' to indicate shortcuts.
- `void shortcut_label (int value)`

Sets whether the widget's label uses '&' to indicate shortcuts.
- `virtual void show ()`

Makes a widget visible.
- `void size (int W, int H)`

Changes the size of the widget.
- `int take_focus ()`

Gives the widget the keyboard focus.
- `unsigned int takeevents () const`

Returns if the widget is able to take events.
- `int test_shortcut ()`

Returns true if the widget's label contains the entered '&x' shortcut.
- `const char * tooltip () const`

Gets the current tooltip text.
- `void tooltip (const char *text)`

Sets the current tooltip text.
- `Fl_Window * top_window () const`

Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`

Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`

Gets the widget type.

- void **type** (uchar t)
Sets the widget type.
- int **use_accents_menu** ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const
Gets the user data for this widget.
- void **user_data** (Fl_Callback_User_Data *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **Fl_When** when () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual ~**Fl_Widget** ()
Destroys the widget.

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
Draws the widget.
- int **handle** (int) **FL_OVERRIDE**
Handles the specified event.
- void **value_damage** () **FL_OVERRIDE**
Asks for partial redraw.

Protected Member Functions inherited from [FI_Valuator](#)

- [FI_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FI_Valuator](#) widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an [FL_WHEN_CHANGED](#) event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an [FL_WHEN_RELEASE](#) event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an [FL_HORIZONTAL](#) one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If [FL_ALIGN_IMAGE_BACKDROP](#) is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)

Internal use only.

- void `y` (int `v`)

Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *`widget`, void *`data`)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *`t`)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *`t`, const bool `require_alt`=false)
Returns true if the given text `t` contains the entered '&x' shortcut.

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
 ,
`OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
 ,
`MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
 ,
`GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
 = 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

11.8.1 Detailed Description

The `FI_Adjuster` widget was stolen from Prisms, and has proven to be very useful for values that need a large dynamic range.

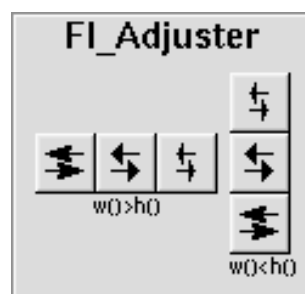


Figure 11.1 `FI_Adjuster`

When you press a button and drag to the right the value increases. When you drag to the left it decreases. The largest button adjusts by 100 * `step()`, the next by 10 * `step()` and that smallest button by `step()`. Clicking on the buttons increments by 10 times the amount dragging by a pixel does. Shift + click decrements by 10 times the amount.

11.8.2 Constructor & Destructor Documentation

11.8.2.1 Fl_Adjuster()

```
Fl_Adjuster::Fl_Adjuster (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Adjuster](#) widget using the given position, size, and label string. It looks best if one of the dimensions is 3 times the other. Inherited destructor destroys the Valuator.

11.8.3 Member Function Documentation

11.8.3.1 draw()

```
void Fl_Adjuster::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                  // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.8.3.2 handle()

```
int Fl_Adjuster::handle (
    int event) [protected], [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

11.8.3.3 `soft()` [1/2]

```
int Fl_Adjuster::soft () const [inline]
```

If "soft" is turned on, the user is allowed to drag the value outside the range.

If they drag the value to one of the ends, let go, then grab again and continue to drag, they can get to any value. Default is one.

11.8.3.4 `soft()` [2/2]

```
void Fl_Adjuster::soft (
    int s) [inline]
```

If "soft" is turned on, the user is allowed to drag the value outside the range.

If they drag the value to one of the ends, let go, then grab again and continue to drag, they can get to any value. Default is one.

11.8.3.5 `value_damage()`

```
void Fl_Adjuster::value_damage () [protected], [virtual]
```

Asks for partial redraw.

Reimplemented from [Fl_Valuator](#).

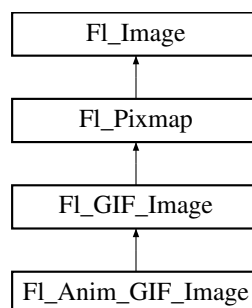
The documentation for this class was generated from the following files:

- `Fl_Adjuster.H`
- `Fl_Adjuster.cxx`

11.9 `Fl_Anim_GIF_Image` Class Reference

The [Fl_Anim_GIF_Image](#) class supports loading, caching, and drawing of animated Compuserve GIFSM images.

Inheritance diagram for `Fl_Anim_GIF_Image`:



Public Types

- enum `Flags` {
`DONT_START` = 1 , `DONT_RESIZE_CANVAS` = 2 , `DONT_SET_AS_IMAGE` = 4 , `OPTIMIZE_MEMORY` = 8 ,
`LOG_FLAG` = 64 , `DEBUG_FLAG` = 128 }

When opening an [Fl_Anim_GIF_Image](#) there are some options that can be passed in a `flags` value.

Public Member Functions

- `Fl_Widget * canvas () const`
Gets the current widget, that is used to display the frame images.
- `void canvas (Fl_Widget *canvas, unsigned short flags=0)`
Link the image back to a widget for automated animation.
- `int canvas_h () const`
Return the height of the animation canvas.

- int `canvas_w` () const
Return the width of the animation canvas.
- void `color_average` (FI_Color c, float i) FL_OVERRIDE
Applies a color average to all frames.
- FI_Image * `copy` () const
- FI_Image * `copy` (int W, int H) const FL_OVERRIDE
Copy and resize the animation frames.
- int `debug` () const
- void `delay` (int frame, double delay)
Set the delay of frame [0-frames() -1] in seconds.
- double `delay` (int frame_) const
Return the delay of frame [0-frames() -1] in seconds.
- void `desaturate` () FL_OVERRIDE
Desaturate to all frames of the animation.
- void `draw` (int x, int y, int w, int h, int cx=0, int cy=0) FL_OVERRIDE
Draw the current frame of the animation.
- FI_Anim_GIF_Image ()
Create an empty animated GIF image shell.
- FI_Anim_GIF_Image (const char *filename, FI_Widget *canvas=0, unsigned short flags=0)
Load an animated GIF image from a file.
- FI_Anim_GIF_Image (const char *imagename, const unsigned char *data, const size_t length, FI_Widget *canvas=0, unsigned short flags=0)
Load an animated GIF image from memory.
- int `frame` () const
Return the current frame.
- void `frame` (int frame)
Set the current frame.
- int `frame_h` (int frame) const
Return the frame dimensions of a frame.
- bool `frame_uncache` () const
Return the active frame_uncache() setting.
- void `frame_uncache` (bool uncache)
Use frame_uncache() to set or forbid frame image uncaching.
- int `frame_w` (int frame) const
Return the frame dimensions of a frame.
- int `frame_x` (int frame) const
Return the frame position of a frame.
- int `frame_y` (int frame) const
Return the frame position of a frame.
- int `frames` () const
Get the number of frames in the animation.
- FI_Image * `image` () const
Return the current frame image.
- FI_Image * `image` (int frame) const
Return the image of the given frame index.
- bool `is_animated` () const
Check if this is a valid animation with more than one frame.
- bool `load` (const char *name, const unsigned char *imgdata=NULL, size_t imglength=0)
Load an animation from a file or from a memory block.
- const char * `name` () const
Return the name of the played file as specified in the constructor.

- bool `next` ()
Show the next frame if the animation is stopped.
- bool `playing` () const
Return if the animation is currently running or stopped.
- `Fl_Anim_GIF_Image` & `resize` (double `scale`)
Resizes the image to the specified size, replacing the current image.
- `Fl_Anim_GIF_Image` & `resize` (int `w`, int `h`)
Resizes the image to the specified size, replacing the current image.
- double `speed` () const
Get the animation speed factor.
- void `speed` (double `speed`)
Set the animation speed factor.
- bool `start` ()
The `start()` method (re-)starts the playing of the frames.
- bool `stop` ()
The `stop()` method stops the playing of the frames.
- void `uncache` () `FL_OVERRIDE`
Uncache all cached image data now.
- bool `valid` () const
Check if animation is valid.
- `~Fl_Anim_GIF_Image` () `FL_OVERRIDE`
Release the image and all cached data.

Public Member Functions inherited from `Fl_GIF_Image`

- `Fl_GIF_Image` (const char *filename)
This constructor loads a GIF image from the given file.
- `Fl_GIF_Image` (const char *imagename, const unsigned char *data)
This constructor loads a GIF image from memory (deprecated).
- `Fl_GIF_Image` (const char *imagename, const unsigned char *data, const size_t length)
This constructor loads a GIF image from memory.

Public Member Functions inherited from `Fl_Pixmap`

- int `cache_h` ()
- int `cache_w` ()
- void `color_average` (`Fl_Color` c, float i) `FL_OVERRIDE`
The `color_average()` method averages the colors in the image with the provided FLTK color value.
- `Fl_Image` * `copy` () const
- `Fl_Image` * `copy` (int W, int H) const `FL_OVERRIDE`
Creates a resized copy of the image.
- void `desaturate` () `FL_OVERRIDE`
The `desaturate()` method converts an image to grayscale.
- void `draw` (int X, int Y)
- void `draw` (int X, int Y, int W, int H, int cx=0, int cy=0) `FL_OVERRIDE`
Draws the image to the current drawing surface with a bounding box.
- `Fl_Pixmap` (char *const *D)
The constructors create a new pixmap from the specified XPM data.
- `Fl_Pixmap` (const char *const *D)
The constructors create a new pixmap from the specified XPM data.
- `Fl_Pixmap` (const uchar *const *D)
The constructors create a new pixmap from the specified XPM data.

- [FI_Pixmap](#) ([uchar](#) *const *D)
The constructors create a new pixmap from the specified XPM data.
- void [label](#) ([FI_Menu_Item](#) *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.
- void [label](#) ([FI_Widget](#) *w) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a widget or menu item.
- void [uncache](#) () [FL_OVERRIDE](#)
If the image has been cached for display, delete the cache data.
- virtual [~FI_Pixmap](#) ()
The destructor frees all memory and server resources that are used by the pixmap.

Public Member Functions inherited from [FI_Image](#)

- virtual class [FI_Shared_Image](#) * [as_shared_image](#) ()
Returns whether an image is an [FI_Shared_Image](#) or not.
- [FI_Image](#) * [copy](#) () const
Creates a copy of the image in the same size.
- int [count](#) () const
Returns the number of data values associated with the image.
- int [d](#) () const
Returns the image depth.
- const char *const * [data](#) () const
Returns a pointer to the current image data array.
- int [data_h](#) () const
Returns the height of the image data.
- int [data_w](#) () const
Returns the width of the image data.
- void [draw](#) (int X, int Y)
Draws the image to the current drawing surface.
- int [fail](#) () const
Returns a value that is not 0 if there is currently no image available.
- [FI_Image](#) (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int [h](#) () const
Returns the current image drawing height in FLTK units.
- void [inactive](#) ()
The [inactive\(\)](#) method calls [color_average\(FI_BACKGROUND_COLOR, 0.33f\)](#) to produce an image that appears grayed out.
- int [ld](#) () const
Returns the current line data size in bytes.
- virtual void [release](#) ()
Releases an [FI_Image](#) - the same as 'delete this'.
- virtual void [scale](#) (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- int [w](#) () const
Returns the current image drawing width in FLTK units.
- virtual [~FI_Image](#) ()
The destructor is a virtual method that frees all memory used by the image.

Static Public Member Functions

- static int [frame_count](#) (const char *name, const unsigned char *imgdata=NULL, size_t imglength=0)
Get the number of frames in a GIF file or in a GIF compressed data block.

Static Public Member Functions inherited from [FI_GIF_Image](#)

- static bool **is_animated** (const char *name_)

Static Public Member Functions inherited from [FI_Image](#)

- static [FI_Labeltype](#) **define_FL_IMAGE_LABEL** ()
- static [FI_RGB_Scaling](#) **RGB_scaling** ()
Returns the currently used RGB image scaling method.
- static void **RGB_scaling** ([FI_RGB_Scaling](#))
Sets the RGB image scaling method used for copy(int, int).
- static [FI_RGB_Scaling](#) **scaling_algorithm** ()
Gets what algorithm is used when resizing a source image to draw it.
- static void **scaling_algorithm** ([FI_RGB_Scaling](#) algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Static Public Attributes

- static bool **loop** = true
The loop flag can be used to (dis-)allow loop count.
- static double **min_delay** = 0.
The min_delay value can be used to set a minimum value for the frame delay for playback.

Static Public Attributes inherited from [FI_GIF_Image](#)

- static bool **animate** = false
Sets how the shared image core routine should treat animated GIF files.

Static Public Attributes inherited from [FI_Image](#)

- static const int **ERR_FILE_ACCESS** = -2
- static const int **ERR_FORMAT** = -3
- static const int **ERR_MEMORY_ACCESS** = -4
- static const int **ERR_NO_IMAGE** = -1
- static bool **register_images_done** = false
True after [fl_register_images\(\)](#) was called, false before.

Protected Member Functions

- void **clear_frames** ()
- bool **next_frame** ()
- void **on_extension_data** ([FI_GIF_Image::GIF_FRAME](#) &f) [FL_OVERRIDE](#)
- void **on_frame_data** ([FI_GIF_Image::GIF_FRAME](#) &f) [FL_OVERRIDE](#)
- void **scale_frame** ()
- void **set_frame** ()
- void **set_frame** (int frame)

Protected Member Functions inherited from [FI_GIF_Image](#)

- [FI_GIF_Image](#) ()
The default constructor creates an empty GIF image.
- [FI_GIF_Image](#) (const char *filename, bool anim)
- [FI_GIF_Image](#) (const char *imagename, const unsigned char *data, const size_t length, bool anim)
- void **load** (const char *filename, bool anim)
The protected [load\(\)](#) methods are used by [FI_Anim_GIF_Image](#) to request loading of animated GIF's.
- void **load** (const char *imagename, const unsigned char *data, const size_t length, bool anim)
- void **load_gif_** (class [FI_Image_Reader](#) &rdr, bool anim=false)

Protected Member Functions inherited from FI_Pixmap

- void **measure** ()

Protected Member Functions inherited from FI_Image

- void **d** (int D)
Sets the current image depth.
- void **data** (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void **draw_empty** (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int **draw_scaled** (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void **h** (int H)
Sets the height of the image data.
- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Static Protected Member Functions

- static void **cb_animate** (void *d)

Static Protected Member Functions inherited from FI_Image

- static void **labeltype** (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void **measure** (const [FI_Label](#) *lo, int &lw, int &lh)

Additional Inherited Members

Public Attributes inherited from FI_Pixmap

- int **alloc_data**

11.9.1 Detailed Description

The [FI_Anim_GIF_Image](#) class supports loading, caching, and drawing of animated Compuserve GIFSM images. The class loads all images contained in the file and animates them by cycling through them as defined by the delay times in the image file.

The user must supply an FLTK widget as "container" in order to see the animation by specifying it in the constructor or later using the [canvas\(\)](#) method.

11.9.2 Member Enumeration Documentation

11.9.2.1 Flags

enum [FI_Anim_GIF_Image::Flags](#)

When opening an [FI_Anim_GIF_Image](#) there are some options that can be passed in a `flags` value.

Enumerator

DONT_START	This flag indicates to the loader that it should not start the animation immediately after successful load, which is the default. It can be started later using the start() method.

Enumerator

DONT_RESIZE_CANVAS	This flag indicates to the loader that it should not resize the canvas widget of the animation to the dimensions of the animation, which is the default. Needed for special use cases.
DONT_SET_AS_IMAGE	This flag indicates to the loader that it should not set the animation as image() member of the canvas widget, which is the default. Needed for special use cases.
OPTIMIZE_MEMORY	Often frames change just a small area of the animation canvas. This flag indicates to the loader to try using less memory by storing frame data not as canvas-sized images but use the sizes defined in the GIF file. The drawbacks are higher cpu usage during playback and maybe minor artifacts when resized.
LOG_FLAG	This flag can be used to print informations about the decoding process to the console.
DEBUG_FLAG	This flag can be used to print even more informations about the decoding process to the console.

11.9.3 Constructor & Destructor Documentation

11.9.3.1 Fl_Anim_GIF_Image() [1/2]

```
Fl_Anim_GIF_Image::Fl_Anim_GIF_Image (
    const char * filename,
    Fl_Widget * canvas = 0,
    unsigned short flags = 0)
```

Load an animated GIF image from a file.

This constructor creates an animated image from a GIF-formatted file. Optionally it applies the [canvas\(\)](#) method after successful load. If [DONT_START](#) is not specified in the `flags` parameter it calls [start\(\)](#) after successful load.

Parameters

in	<i>filename</i>	path and name of GIF file in the file system
in	<i>canvas</i>	a widget that will show and animate the GIF, or NULL
in	<i>flags</i>	see Flags for details, or 0

Note

The GIF image must be decoupled from the canvas by calling `myGif->canvas (NULL) ;` before deleting the canvas.

11.9.3.2 Fl_Anim_GIF_Image() [2/2]

```
Fl_Anim_GIF_Image::Fl_Anim_GIF_Image (
    const char * imagename,
    const unsigned char * data,
    const size_t length,
    Fl_Widget * canvas = 0,
    unsigned short flags = 0)
```

Load an animated GIF image from memory.

This constructor creates an animated image from a GIF-formatted block in memory. Optionally it applies the [canvas\(\)](#) method after successful load. If [DONT_START](#) is not specified in the `flags` parameter it calls [start\(\)](#) after successful load.

`imagename` can be NULL. If a name is given, the image is added to the list of shared images and will be available by that name.

Parameters

in	<i>imagename</i>	a name given to this image or NULL
in	<i>data</i>	pointer to the start of the GIF image in memory
in	<i>length</i>	length of the GIF image in memory
in	<i>canvas</i>	a widget that will show and animate the GIF, or NULL
in	<i>flags</i>	see Flags for details, or 0

Note

The GIF image must be decoupled from the canvas by calling `myGif->canvas(NULL)` ; before deleting the canvas.

11.9.3.3 ~Fl_Anim_GIF_Image()

```
Fl_Anim_GIF_Image::~Fl_Anim_GIF_Image ()
```

Release the image and all cached data.

Also removes the animation timer.

11.9.4 Member Function Documentation**11.9.4.1 canvas() [1/2]**

```
Fl_Widget * Fl_Anim_GIF_Image::canvas () const
```

Gets the current widget, that is used to display the frame images.

Returns

a pointer to a widget

11.9.4.2 canvas() [2/2]

```
void Fl_Anim_GIF_Image::canvas (
    Fl_Widget * canvas,
    unsigned short flags = 0)
```

Link the image back to a widget for automated animation.

This method sets current widget, that is used to display the frame images. The `flags` parameter specifies whether the canvas widget is resized to the animation dimensions and/or its `image()` method will be used to set the current frame image during animation.

Parameters

in	<i>canvas</i>	a pointer to the widget that will show the animation
in	<i>flags</i>	see Flags

Note

The GIF image must be decoupled from the canvas by calling `myGif->canvas(NULL)` ; before deleting the canvas.

11.9.4.3 canvas_h()

```
int Fl_Anim_GIF_Image::canvas_h () const
```

Return the height of the animation canvas.

Returns

the width in pixel units

11.9.4.4 canvas_w()

```
int Fl_Anim_GIF_Image::canvas_w () const
```

Return the width of the animation canvas.

Returns

the width in pixel units

11.9.4.5 color_average()

```
void Fl_Anim_GIF_Image::color_average (
    Fl_Color c,
    float i) [virtual]
```

Applies a color average to all frames.

The `color_average()` method averages the colors in the image with the provided FLTK color value.

Parameters

in	<i>c</i>	blend color
in	<i>i</i>	a value between 0.0 and 1.0 where 0 results in the blend color, and 1 returns the original image

Reimplemented from [Fl_Image](#).

11.9.4.6 copy()

```
Fl_Image * Fl_Anim_GIF_Image::copy (
    int W,
    int H) const [virtual]
```

Copy and resize the animation frames.

The virtual `copy()` method makes a copy of the animated image and resizes all of its frame images to W x H using the current resize method.

Parameters

in	<i>W,H</i>	new size in FLTK pixel units

Returns

the resized copy of the animation

Reimplemented from [Fl_Image](#).

11.9.4.7 delay() [1/2]

```
void Fl_Anim_GIF_Image::delay (
    int frame,
    double delay)
```

Set the delay of frame [0-`frames()` -1] in seconds.

Parameters

in	<i>frame</i>	index into frame list
in	<i>delay</i>	to next frame in seconds

11.9.4.8 delay() [2/2]

```
double Fl_Anim_GIF_Image::delay (
    int frame) const
```

Return the delay of frame [0-`frames()` -1] in seconds.

Parameters

in	<i>frame</i>	index into frame list

Returns

delay to next frame in seconds

11.9.4.9 desaturate()

```
void Fl_Anim_GIF_Image::desaturate () [virtual]
```

Desaturate to all frames of the animation.

Reimplemented from [Fl_Image](#).

11.9.4.10 draw()

```
void Fl_Anim_GIF_Image::draw (
    int x,
    int y,
    int w,
    int h,
    int cx = 0,
    int cy = 0) [virtual]
```

Draw the current frame of the animation.

Parameters

in	<i>x,y,w,h</i>	target rectangle
in	<i>cx,cy</i>	source offset

Reimplemented from [Fl_Image](#).

11.9.4.11 frame() [1/2]

```
int Fl_Anim_GIF_Image::frame () const
```

Return the current frame.

Returns

the current frame index in the range for 0 to [frames\(\)](#) - 1.

-1 if the image has no frames.

11.9.4.12 frame() [2/2]

```
void Fl_Anim_GIF_Image::frame (
    int frame)
```

Set the current frame.

Parameters

in	<i>frame</i>	index into list of frames

11.9.4.13 frame_count()

```
int Fl_Anim_GIF_Image::frame_count (
    const char * name,
    const unsigned char * imgdata = NULL,
    size_t imglength = 0) [static]
```

Get the number of frames in a GIF file or in a GIF compressed data block.

The static [frame_count\(\)](#) method is just a convenience method for getting the number of images (frames) stored in a GIF file.

As this count is not readily available in the GIF header, the whole GIF file has be parsed (which is done here by using a temporary [Fl_Anim_GIF_Image](#) object for simplicity). So this call may be slow with large files.

If `imgdata` is `NULL`, the image will be read from the file. Otherwise, it will be read from memory.

Parameters

in	<i>name</i>	path and name of GIF file in the file system, ignored when reading from memeory
in	<i>imgdata</i>	pointer to the start of the GIF image in memory, or <code>NULL</code> to read from a file
in	<i>imglength</i>	length of the GIF image in memory, or 0

Returns

the number of frames in the animation

11.9.4.14 frame_h()

```
int Fl_Anim_GIF_Image::frame_h (
    int frame) const
```

Return the frame dimensions of a frame.

Usefull only if loaded with 'optimize_mem' and the animation also has size optimized frames.

Parameters

in	<i>frame</i>	index into frame list

Returns

height in FLTK pixle units

11.9.4.15 frame_uncache() [1/2]

```
bool Fl_Anim_GIF_Image::frame_uncache () const
```

Return the active [frame_uncache\(\)](#) setting.

Returns

true if caching is disabled

11.9.4.16 frame_uncache() [2/2]

```
void Fl_Anim_GIF_Image::frame_uncache (
    bool uncache)
```

Use [frame_uncache\(\)](#) to set or forbid frame image uncaching.

If frame uncaching is set, frame images are not offscreen cached for re-use and will be re-created every time they are displayed. This saves a lot of memory on the expense of cpu usage and should be carefully considered. Per default frame caching will be done.

Parameters

in	<i>uncache</i>	true to disable caching

11.9.4.17 frame_w()

```
int Fl_Anim_GIF_Image::frame_w (  
    int frame) const
```

Return the frame dimensions of a frame.

Usefull only if loaded with 'optimize_mem' and the animation also has size optimized frames.

Parameters

in	<i>frame</i>	index into frame list

Returns

width in FLTK pixle units

11.9.4.18 frame_x()

```
int Fl_Anim_GIF_Image::frame_x (  
    int frame) const
```

Return the frame position of a frame.

Usefull only if loaded with 'optimize_mem' and the animation also has size optimized frames.

Parameters

in	<i>frame</i>	index into frame list

Returns

x position in FLTK pixle units

11.9.4.19 frame_y()

```
int Fl_Anim_GIF_Image::frame_y (  
    int frame) const
```

Return the frame position of a frame.

Usefull only if loaded with 'optimize_mem' and the animation also has size optimized frames.

Parameters

in	<i>frame</i>	index into frame list

Returns

y position in FLTK pixle units

11.9.4.20 frames()

```
int Fl_Anim_GIF_Image::frames () const
```

Get the number of frames in the animation.

Returns

the number of frames

11.9.4.21 image() [1/2]

```
Fl_Image * Fl_Anim_GIF_Image::image () const
```

Return the current frame image.

Returns

a pointer to the image or NULL if this is not an animation.

11.9.4.22 image() [2/2]

```
Fl_Image * Fl_Anim_GIF_Image::image (
    int frame_) const
```

Return the image of the given frame index.

Parameters

in	<i>frame</i> ↔	index into list of frames
	—	

Returns

image data or NULL if the frame number is not valid.

11.9.4.23 is_animated()

```
bool Fl_Anim_GIF_Image::is_animated () const
```

Check if this is a valid animation with more than one frame.

The `is_animated()` method is just a convenience method for testing the valid flag and the frame count being greater 1.

Returns

true if the animation is valid and has multiple frames.

11.9.4.24 load()

```
bool Fl_Anim_GIF_Image::load (
    const char * name,
    const unsigned char * imgdata = NULL,
    size_t imglength = 0)
```

Load an animation from a file or from a memory block.

The `load()` method is either used from the constructor to load the image from the given file, or to re-load an existing animation from another file.

Parameters

in	<i>name</i>	path and name of GIF file in the file system, or the image name when reading from memory
in	<i>imgdata</i>	pointer to the start of the GIF image in memory, or NULL to read from a file
in	<i>imglength</i>	length of the GIF image in memory, or 0

Returns

true if the animation loaded correctly

11.9.4.25 name()

```
const char * Fl_Anim_GIF_Image::name () const
```

Return the name of the played file as specified in the constructor.
If read from a memory block, this returns the name of the animation.

Returns

pointer to a C string

11.9.4.26 next()

```
bool Fl_Anim_GIF_Image::next ()
```

Show the next frame if the animation is stopped.

Returns

true if the animation has frames

11.9.4.27 on_extension_data()

```
void Fl_Anim_GIF_Image::on_extension_data (
    Fl_GIF_Image::GIF_FRAME & f) [protected], [virtual]
```

Reimplemented from [Fl_GIF_Image](#).

11.9.4.28 on_frame_data()

```
void Fl_Anim_GIF_Image::on_frame_data (
    Fl_GIF_Image::GIF_FRAME & f) [protected], [virtual]
```

Reimplemented from [Fl_GIF_Image](#).

11.9.4.29 playing()

```
bool Fl_Anim_GIF_Image::playing () const [inline]
```

Return if the animation is currently running or stopped.

Returns

true if the animation is running

11.9.4.30 resize() [1/2]

```
Fl_Anim_GIF_Image & Fl_Anim_GIF_Image::resize (
    double scale)
```

Resizes the image to the specified size, replacing the current image.
If [DONT_RESIZE_CANVAS](#) is not set, the canvas widget will also be resized.

Parameters

<i>in</i>	<i>scale</i>	rescale factor in relation to current size

11.9.4.31 resize() [2/2]

```
Fl_Anim_GIF_Image & Fl_Anim_GIF_Image::resize (
    int w,
    int h)
```

Resizes the image to the specified size, replacing the current image.
If [DONT_RESIZE_CANVAS](#) is not set, the canvas widget will also be resized.

Parameters

in	<i>w,h</i>	new size of teh naimtion frames

11.9.4.32 speed() [1/2]

```
double Fl_Anim_GIF_Image::speed () const
```

Get the animation speed factor.

Returns

the current speed factor

11.9.4.33 speed() [2/2]

```
void Fl_Anim_GIF_Image::speed (
    double speed)
```

Set the animation speed factor.

The [speed\(\)](#) method changes the playing speed to *speed* x original speed. E.g. to play at half speed call it with 0.5, for double speed with 2.

Parameters

in	<i>speed</i>	floating point speed factor

11.9.4.34 start()

```
bool Fl_Anim_GIF_Image::start ()
```

The [start\(\)](#) method (re-)starts the playing of the frames.

Returns

true if the animation has frames

11.9.4.35 stop()

```
bool Fl_Anim_GIF_Image::stop ()
```

The [stop\(\)](#) method stops the playing of the frames.

Returns

true if the animation has frames

11.9.4.36 uncache()

```
void Fl_Anim_GIF_Image::uncache () [virtual]
```

Uncache all cached image data now.

Re-implemented from [Fl_Pixmap](#).

Reimplemented from [Fl_Image](#).

11.9.4.37 valid()

```
bool Fl_Anim_GIF_Image::valid () const
```

Check if animation is valid.

Returns

true if the class has successfully loaded and the image has at least one frame.

11.9.5 Member Data Documentation

11.9.5.1 loop

```
bool Fl_Anim_GIF_Image::loop = true [static]
```

The loop flag can be used to (dis-)allow loop count.

If set (which is the default), the animation will be stopped after the number of repeats specified in the GIF file (typically this count is set to 'forever' anyway). If cleared the animation will always be 'forever', regardless of what is specified in the GIF file.

11.9.5.2 min_delay

```
double Fl_Anim_GIF_Image::min_delay = 0. [static]
```

The min_delay value can be used to set a minimum value for the frame delay for playback.

This is to prevent CPU hogs caused by images with very low delay rates. This is a global value for all [Fl_Anim_GIF_Image](#) objects.

The documentation for this class was generated from the following files:

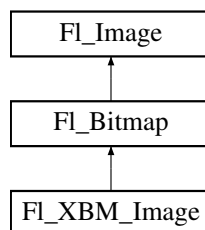
- [Fl_Anim_GIF_Image.H](#)
- [Fl_Anim_GIF_Image.cxx](#)

11.10 Fl_Bitmap Class Reference

The [Fl_Bitmap](#) class supports caching and drawing of mono-color (bitmap) images.

```
#include <Fl_Bitmap.H>
```

Inheritance diagram for [Fl_Bitmap](#):



Public Member Functions

- `int cache_h ()`
- `int cache_w ()`
- `Fl\_Image * copy () const`
- `Fl_Image * copy (int W, int H) const FL_OVERRIDE
Creates a resized copy of the image.`
- `void draw (int X, int Y)`
- `void draw (int X, int Y, int W, int H, int cx=0, int cy=0) FL_OVERRIDE
Draws the image to the current drawing surface with a bounding box.`
- `Fl_Bitmap (const char *bits, int bits_length, int W, int H)
The constructors create a new bitmap from the specified bitmap data.`
- `Fl_Bitmap (const char *bits, int W, int H)
The constructors create a new bitmap from the specified bitmap data.`
- `Fl_Bitmap (const uchar *bits, int bits_length, int W, int H)
The constructors create a new bitmap from the specified bitmap data.`
- `Fl_Bitmap (const uchar *bits, int W, int H)
The constructors create a new bitmap from the specified bitmap data.`
- `void label (Fl_Menu_Item *m) FL_OVERRIDE
This method is an obsolete way to set the image attribute of a menu item.`

- void `label (FL_Widget *w) FL_OVERRIDE`
This method is an obsolete way to set the image attribute of a widget or menu item.
- void `uncache () FL_OVERRIDE`
If the image has been cached for display, delete the cache data.
- virtual `~FL_Bitmap ()`
The destructor frees all memory and server resources that are used by the bitmap.

Public Member Functions inherited from `FL_Image`

- virtual class `FL_Shared_Image * as_shared_image ()`
Returns whether an image is an `FL_Shared_Image` or not.
- virtual void `color_average (FL_Color c, float i)`
The `color_average()` method averages the colors in the image with the provided FLTK color value.
- `FL_Image * copy () const`
Creates a copy of the image in the same size.
- int `count () const`
Returns the number of data values associated with the image.
- int `d () const`
Returns the image depth.
- const char *const * `data () const`
Returns a pointer to the current image data array.
- int `data_h () const`
Returns the height of the image data.
- int `data_w () const`
Returns the width of the image data.
- virtual void `desaturate ()`
The `desaturate()` method converts an image to grayscale.
- void `draw (int X, int Y)`
Draws the image to the current drawing surface.
- int `fail () const`
Returns a value that is not 0 if there is currently no image available.
- `FL_Image (int W, int H, int D)`
The constructor creates an empty image with the specified width, height, and depth.
- int `h () const`
Returns the current image drawing height in FLTK units.
- void `inactive ()`
The `inactive()` method calls `color_average(FL_BACKGROUND_COLOR, 0.33f)` to produce an image that appears grayed out.
- int `ld () const`
Returns the current line data size in bytes.
- virtual void `release ()`
Releases an `FL_Image` - the same as 'delete this'.
- virtual void `scale (int width, int height, int proportional=1, int can_expand=0)`
Sets the drawing size of the image.
- int `w () const`
Returns the current image drawing width in FLTK units.
- virtual `~FL_Image ()`
The destructor is a virtual method that frees all memory used by the image.

Public Attributes

- int **alloc_array**
Non-zero if array points to bitmap data allocated internally.
- const [uchar](#) * **array**
pointer to raw bitmap data

Friends

- class **FI_Graphics_Driver**

Additional Inherited Members**Static Public Member Functions inherited from [FI_Image](#)**

- static [FI_Labeltype](#) **define_FL_IMAGE_LABEL** ()
- static [FI_RGB_Scaling](#) **RGB_scaling** ()
Returns the currently used RGB image scaling method.
- static void **RGB_scaling** ([FI_RGB_Scaling](#))
Sets the RGB image scaling method used for copy(int, int).
- static [FI_RGB_Scaling](#) **scaling_algorithm** ()
Gets what algorithm is used when resizing a source image to draw it.
- static void **scaling_algorithm** ([FI_RGB_Scaling](#) algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Static Public Attributes inherited from [FI_Image](#)

- static const int **ERR_FILE_ACCESS** = -2
- static const int **ERR_FORMAT** = -3
- static const int **ERR_MEMORY_ACCESS** = -4
- static const int **ERR_NO_IMAGE** = -1
- static bool **register_images_done** = false
True after [fl_register_images\(\)](#) was called, false before.

Protected Member Functions inherited from [FI_Image](#)

- void **d** (int D)
Sets the current image depth.
- void **data** (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void **draw_empty** (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int **draw_scaled** (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void **h** (int H)
Sets the height of the image data.
- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Static Protected Member Functions inherited from [FI_Image](#)

- static void **labeltype** (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void **measure** (const [FI_Label](#) *lo, int &lw, int &lh)

11.10.1 Detailed Description

The [Fl_Bitmap](#) class supports caching and drawing of mono-color (bitmap) images. Images are drawn using the current color.

11.10.2 Constructor & Destructor Documentation

11.10.2.1 Fl_Bitmap() [1/4]

```
Fl_Bitmap::Fl_Bitmap (
    const uchar * bits,
    int W,
    int H) [inline]
```

The constructors create a new bitmap from the specified bitmap data.

See also

[Fl_Bitmap\(const uchar *bits, int bits_length, int W, int H\)](#)

11.10.2.2 Fl_Bitmap() [2/4]

```
Fl_Bitmap::Fl_Bitmap (
    const char * bits,
    int W,
    int H) [inline]
```

The constructors create a new bitmap from the specified bitmap data.

See also

[Fl_Bitmap\(const char *bits, int bits_length, int W, int H\)](#)

11.10.2.3 Fl_Bitmap() [3/4]

```
Fl_Bitmap::Fl_Bitmap (
    const uchar * bits,
    int bits_length,
    int W,
    int H)
```

The constructors create a new bitmap from the specified bitmap data.

If the provided array is too small to contain all the image data, the constructor will not generate the bitmap to avoid illegal memory read access and instead set `data` to `NULL` and `ld` to `ERR_MEMORY_ACCESS`.

Parameters

<i>bits</i>	bitmap data, one pixel per bit, rows are rounded to the next byte
<i>bits_length</i>	length of the <i>bits</i> array in bytes
<i>W</i>	image width in pixels
<i>H</i>	image height in pixels

See also

[Fl_Bitmap\(const char *bits, int bits_length, int W, int H\)](#), [Fl_Bitmap\(const uchar *bits, int W, int H\)](#)

11.10.2.4 Fl_Bitmap() [4/4]

```
Fl_Bitmap::Fl_Bitmap (
    const char * bits,
    int bits_length,
```

```
int W,
int H)
```

The constructors create a new bitmap from the specified bitmap data.

If the provided array is too small to contain all the image data, the constructor will not generate the bitmap to avoid illegal memory read access and instead set `data` to `NULL` and `ld` to `ERR_MEMORY_ACCESS`.

Parameters

<i>bits</i>	bitmap data, one pixel per bit, rows are rounded to the next byte
<i>bits_length</i>	length of the <i>bits</i> array in bytes
<i>W</i>	image width in pixels
<i>H</i>	image height in pixels

See also

[Fl_Bitmap\(const uchar *bits, int bits_length, int W, int H\)](#), [Fl_Bitmap\(const char *bits, int W, int H\)](#)

11.10.3 Member Function Documentation

11.10.3.1 copy()

```
Fl_Image * Fl_Bitmap::copy (
    int W,
    int H) const [virtual]
```

Creates a resized copy of the image.

It is recommended not to call this member function to reduce the size of an image to the size of the area where this image will be drawn, and to use [Fl_Image::scale\(\)](#) instead.

The new image should be released when you are done with it.

Note: since FLTK 1.4.0 you can use [Fl_Image::release\(\)](#) for all types of images (i.e. all subclasses of [Fl_Image](#)) instead of operator `delete` for [Fl_Image](#)'s and [Fl_Image::release\(\)](#) for [Fl_Shared_Image](#)'s.

The new image data will be converted to the requested size. RGB images are resized using the algorithm set by [Fl_Image::RGB_scaling\(\)](#).

For the new image the following equations are true:

- `w() == data_w() == W`
- `h() == data_h() == H`

Parameters

in	<i>W,H</i>	Requested width and height of the new image

Note

The returned image can be safely cast to the same image type as that of the source image provided this type is one of [Fl_RGB_Image](#), [Fl_SVG_Image](#), [Fl_Pixmap](#), [Fl_Bitmap](#), [Fl_Tiled_Image](#), [Fl_Anim_GIF_Image](#) and [Fl_Shared_Image](#). Returned objects copied from images of other, derived, image classes belong to the parent class appearing in this list. For example, the copy of an [Fl_GIF_Image](#) is an object of class [Fl_Pixmap](#).

Since FLTK 1.4.0 this method is 'const'. If you derive your own class from [Fl_Image](#) or any subclass your overridden methods of '[Fl_Image::copy\(\) const](#)' and '[Fl_Image::copy\(int, int\) const](#)' **must** also be 'const' for inheritance to work properly. This is different than in FLTK 1.3.x and earlier where these methods have not been 'const'.

Reimplemented from [Fl_Image](#).

11.10.3.2 draw()

```
void Fl_Bitmap::draw (
    int X,
    int Y,
    int W,
    int H,
    int cx = 0,
    int cy = 0) [virtual]
```

Draws the image to the current drawing surface with a bounding box.

Arguments *X*, *Y*, *W*, *H* specify a bounding box for the image, with the origin (upper-left corner) of the image offset by the *cx* and *cy* arguments.

In other words: `fl_push_clip(X,Y,W,H)` is applied, the image is drawn with its upper-left corner at *X-cx*, *Y-cy* and its own width and height, `fl_pop_clip()` is applied.

Reimplemented from [Fl_Image](#).

11.10.3.3 label() [1/2]

```
void Fl_Bitmap::label (
    Fl_Menu_Item * m) [virtual]
```

This method is an obsolete way to set the image attribute of a menu item.

Deprecated Please use [Fl_Menu_Item::image\(\)](#) instead.

Reimplemented from [Fl_Image](#).

11.10.3.4 label() [2/2]

```
void Fl_Bitmap::label (
    Fl_Widget * widget) [virtual]
```

This method is an obsolete way to set the image attribute of a widget or menu item.

Deprecated Please use [Fl_Widget::image\(\)](#) or [Fl_Widget::deimage\(\)](#) instead.

Reimplemented from [Fl_Image](#).

11.10.3.5 uncache()

```
void Fl_Bitmap::uncache () [virtual]
```

If the image has been cached for display, delete the cache data.

This allows you to change the data used for the image and then redraw it without recreating an image object.

Reimplemented from [Fl_Image](#).

The documentation for this class was generated from the following files:

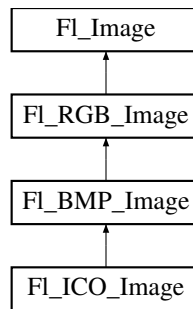
- [Fl_Bitmap.H](#)
- [Fl_Bitmap.cxx](#)

11.11 Fl_BMP_Image Class Reference

The [Fl_BMP_Image](#) class supports loading, caching, and drawing of Windows Bitmap (BMP) image files.

```
#include <Fl_BMP_Image.H>
```

Inheritance diagram for [Fl_BMP_Image](#):



Public Member Functions

- [FI_BMP_Image](#) (const char *filename)
This constructor loads the named BMP image from the given BMP filename.
- [FI_BMP_Image](#) (const char *imagename, const unsigned char *data, const long length=-1)
This constructor loads a BMP image from memory.

Public Member Functions inherited from [FI_RGB_Image](#)

- virtual [FI_SVG_Image](#) * [as_svg_image](#) ()
Returns whether an image is an [FI_SVG_Image](#) or not.
- int [cache_h](#) ()
- int [cache_w](#) ()
- void [color_average](#) ([FI_Color](#) c, float i) [FL_OVERRIDE](#)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [FI_Image](#) * [copy](#) () const
- [FI_Image](#) * [copy](#) (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void [desaturate](#) () [FL_OVERRIDE](#)
The [desaturate\(\)](#) method converts an image to grayscale.
- void [draw](#) (int X, int Y)
- void [draw](#) (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [FI_RGB_Image](#) (const [FI_Pixmap](#) *pixmap, [FI_Color](#) bg=[FL_GRAY](#))
The constructor creates a new RGBA image from the specified [FI_Pixmap](#).
- [FI_RGB_Image](#) (const uchar *bits, int bits_length, int W, int H, int D, int LD)
The constructor creates a new image from the specified data.
- [FI_RGB_Image](#) (const uchar *bits, int W, int H, int D=3, int LD=0)
The constructor creates a new image from the specified data.
- void [label](#) ([FI_Menu_Item](#) *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.
- void [label](#) ([FI_Widget](#) *w) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a widget or menu item.
- virtual void [normalize](#) ()
Makes sure the object is fully initialized.
- void [uncache](#) () [FL_OVERRIDE](#)
If the image has been cached for display, delete the cache data.
- ~[FI_RGB_Image](#) () [FL_OVERRIDE](#)
The destructor frees all memory and server resources that are used by the image.

Public Member Functions inherited from **FI_Image**

- virtual class **FI_Shared_Image** * **as_shared_image** ()
*Returns whether an image is an **FI_Shared_Image** or not.*
- **FI_Image** * **copy** () const
Creates a copy of the image in the same size.
- int **count** () const
Returns the number of data values associated with the image.
- int **d** () const
Returns the image depth.
- const char *const * **data** () const
Returns a pointer to the current image data array.
- int **data_h** () const
Returns the height of the image data.
- int **data_w** () const
Returns the width of the image data.
- void **draw** (int X, int Y)
Draws the image to the current drawing surface.
- int **fail** () const
Returns a value that is not 0 if there is currently no image available.
- **FI_Image** (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int **h** () const
Returns the current image drawing height in FLTK units.
- void **inactive** ()
*The **inactive()** method calls **color_average(FI_BACKGROUND_COLOR, 0.33f)** to produce an image that appears grayed out.*
- int **ld** () const
Returns the current line data size in bytes.
- virtual void **release** ()
*Releases an **FI_Image** - the same as 'delete this'.*
- virtual void **scale** (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- int **w** () const
Returns the current image drawing width in FLTK units.
- virtual ~**FI_Image** ()
The destructor is a virtual method that frees all memory used by the image.

Protected Member Functions

- void **load_bmp_** (class **FI_Image_Reader** &rdr, int ico_height=0, int ico_width=0)

Protected Member Functions inherited from **FI_Image**

- void **d** (int D)
Sets the current image depth.
- void **data** (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void **draw_empty** (int X, int Y)
*The protected method **draw_empty()** draws a box with an X in it.*
- int **draw_scaled** (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void **h** (int H)

- *Sets the height of the image data.*
- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Additional Inherited Members

Static Public Member Functions inherited from FI_RGB_Image

- static size_t **max_size** ()
Returns the maximum allowed image size in bytes when creating an FI_RGB_Image object.
- static void **max_size** (size_t size)
Sets the maximum allowed image size in bytes when creating an FI_RGB_Image object.

Static Public Member Functions inherited from FI_Image

- static FI_Labeltype **define_FL_IMAGE_LABEL** ()
- static FI_RGB_Scaling **RGB_scaling** ()
Returns the currently used RGB image scaling method.
- static void **RGB_scaling** (FI_RGB_Scaling)
Sets the RGB image scaling method used for copy(int, int).
- static FI_RGB_Scaling **scaling_algorithm** ()
Gets what algorithm is used when resizing a source image to draw it.
- static void **scaling_algorithm** (FI_RGB_Scaling algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Public Attributes inherited from FI_RGB_Image

- int **alloc_array**
If non-zero, the object's data array is delete[]'d when deleting the object.
- const uchar * **array**
Points to the start of the object's data array.

Static Public Attributes inherited from FI_Image

- static const int **ERR_FILE_ACCESS** = -2
- static const int **ERR_FORMAT** = -3
- static const int **ERR_MEMORY_ACCESS** = -4
- static const int **ERR_NO_IMAGE** = -1
- static bool **register_images_done** = false
True after fl_register_images() was called, false before.

Static Protected Member Functions inherited from FI_Image

- static void **labeltype** (const FI_Label *lo, int lx, int ly, int lw, int lh, FI_Align la)
- static void **measure** (const FI_Label *lo, int &lw, int &lh)

11.11.1 Detailed Description

The FI_BMP_Image class supports loading, caching, and drawing of Windows Bitmap (BMP) image files.

11.11.2 Constructor & Destructor Documentation

11.11.2.1 FI_BMP_Image() [1/2]

```
Fl_BMP_Image::Fl_BMP_Image (
    const char * filename)
```

This constructor loads the named BMP image from the given BMP filename.

The destructor frees all memory and server resources that are used by the image.

Use [Fl_Image::fail\(\)](#) to check if [FI_BMP_Image](#) failed to load. [fail\(\)](#) returns `ERR_FILE_ACCESS` if the file could not be opened or read, `ERR_FORMAT` if the BMP format could not be decoded, and `ERR_NO_IMAGE` if the image could not be loaded for another reason.

Parameters

in	<i>filename</i>	a full path and name pointing to a BMP file.

See also

[Fl_BMP_Image::Fl_BMP_Image\(const char* imagename, const unsigned char *data, const long length = -1\);](#)

11.11.2.2 FI_BMP_Image() [2/2]

```
Fl_BMP_Image::Fl_BMP_Image (
    const char * imagename,
    const unsigned char * data,
    const long length = -1)
```

This constructor loads a BMP image from memory.

Construct an image from a block of memory inside the application. Fluid offers "binary data" chunks as a great way to add image data into the C++ source code. *imagename* can be NULL. If a name is given, the image is added to the list of shared images and will be available by that name.

The destructor frees all memory and server resources that are used by the image.

The (new and optional) third parameter *length* **should** be used so buffer overruns (i.e. truncated images) can be checked. See note below.

If *length* is not used

- it defaults to -1 (unlimited size)
- buffer overruns will not be checked.

Note

The optional parameter *length* is available since FLTK 1.4.0. Not using it is deprecated and old code should be modified to use it. This parameter will likely become mandatory in a future FLTK version.

Use [Fl_Image::fail\(\)](#) to check if [FI_BMP_Image](#) failed to load. [fail\(\)](#) returns `ERR_FILE_ACCESS` if the image could not be read from memory, `ERR_FORMAT` if the BMP format could not be decoded, and `ERR_NO_IMAGE` if the image could not be loaded for another reason.

Parameters

in	<i>imagename</i>	A name given to this image or NULL
in	<i>data</i>	Pointer to the start of the BMP image in memory.
in	<i>length</i>	Length of the BMP image in memory.

See also

[Fl_BMP_Image::Fl_BMP_Image\(const char *filename\)](#)

[Fl_Shared_Image](#)

The documentation for this class was generated from the following files:

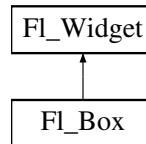
- [FI_BMP_Image.H](#)
- [FI_BMP_Image.cxx](#)

11.12 Fl_Box Class Reference

This widget simply draws its box, and possibly its label.

```
#include <Fl_Box.H>
```

Inheritance diagram for Fl_Box:



Public Member Functions

- [Fl_Box](#) ([Fl_Boxtype](#) B, int X, int Y, int W, int H, const char *L)
Creates a new [Fl_Box](#) widget with the given boxtype, coordinates, size, and label.
- [Fl_Box](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Box](#) widget with the given coordinates, size, and label.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [Fl_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [Fl_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([Fl_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()
Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [Fl_Group](#) * [as_group](#) ()
Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).
- virtual [Fl_Group](#) const * [as_group](#) () const
- virtual [Fl_Window](#) * [as_window](#) ()
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- virtual [Fl_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)

- Sets the image to use as part of the widget label when in the active state.*

 - void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const

Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)

Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const

Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()

Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.
- void [clear_output](#) ()

Sets a widget to accept input.
- void [clear_visible](#) ()

Hides the widget.
- void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const

Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)

Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)

Sets the background and selection color of the widget.
- [FI_Color](#) [color2](#) () const

For back compatibility only.
- void [color2](#) (unsigned a)

For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const

Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)

Sets the current label.
- void [copy_tooltip](#) (const char *text)

Sets the current tooltip text.
- uchar [damage](#) () const

Returns non-zero if [draw\(\)](#) needs to be called.

- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)

- Sets the current label pointer.*

 - void `label` (`FI_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const

Gets the label color.
- void `labelcolor` (`FI_Color` c)

Sets the label color.
- `FI_Font` `labelfont` () const

Gets the font to use.
- void `labelfont` (`FI_Font` f)

Sets the font to use.
- `FI_Fonsize` `labelsize` () const

Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)

Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

Gets the label type.
- void `labeltype` (`FI_Labeltype` a)

Sets the label type.
- void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FI_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FI_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- virtual void `resize` (int x, int y, int w, int h)

Changes the size or position of the widget.
- `FI_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`FI_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.

- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const

- Returns whether a widget and all its parents are visible.*
- int **w** () const
Gets the widget width.
- **FL_When** when () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **FL_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual ~**FL_Widget** ()
Destroys the widget.

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
Draws the widget.

Protected Member Functions inherited from **FL_Widget**

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (**FL_Boxtype** t, **FL_Color** c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (**FL_Boxtype** t, int x, int y, int w, int h, **FL_Color** c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (**FL_Boxtype** t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (**FL_Boxtype** t, int x, int y, int w, int h, **FL_Color** bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **FL_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)

Internal use only.

- void `x` (int v)

Internal use only.

- void `y` (int v)

Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget` *widget, void *data)

The default callback for all widgets that don't set a callback.

- static unsigned int `label_shortcut` (const char *t)

Returns the Unicode value of the '&x' shortcut in a given text.

- static int `test_shortcut` (const char *, const bool require_alt=false)

Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `Fl_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
 ,
`OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
 ,
`MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
 ,
`GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
 = 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }

flags possible values enumeration.

11.12.1 Detailed Description

This widget simply draws its box, and possibly its label.

Putting it before some other widgets and making it big enough to surround them will let you draw a frame around them.

11.12.2 Constructor & Destructor Documentation

11.12.2.1 `Fl_Box()` [1/2]

```
Fl_Box::Fl_Box (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new `Fl_Box` widget with the given coordinates, size, and label.

This constructor sets `box()` to `FL_NO_BOX`, which means it is invisible.

However such widgets are useful as placeholders or `Fl_Group::resizable()` values. To change the box to something visible, use `box(Fl_Boxtype)`.

The destructor removes the box from its parent group.

Parameters

in	<i>X,Y</i>	the position of the widget relative to the enclosing window
in	<i>W,H</i>	size of the widget in pixels
in	<i>L</i>	optional text for the widget label (default: no label)

See also

[Fl_Box\(Fl_Boxtype b, int X, int Y, int W, int H, const char *L\)](#)

11.12.2.2 Fl_Box() [2/2]

```
Fl_Box::Fl_Box (
    Fl_Boxtype B,
    int X,
    int Y,
    int W,
    int H,
    const char * L)
```

Creates a new [Fl_Box](#) widget with the given boxtype, coordinates, size, and label.

This constructor sets [box\(\)](#) to the given Fl_Boxtype B.

You must also specify a label but it can be `nullptr` (0, NULL) if you don't want or need a visible label.

The destructor removes the box from its parent group.

Parameters

in	<i>B</i>	boxtype of the widget
in	<i>X,Y</i>	the position of the widget relative to the enclosing window
in	<i>W,H</i>	size of the widget in pixels
in	<i>L</i>	optional text for the widget label (see description)

See also

[Fl_Box\(int X, int Y, int W, int H, const char *L\)](#)

11.12.3 Member Function Documentation**11.12.3.1 draw()**

```
void Fl_Box::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.12.3.2 handle()

```
int Fl_Box::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

The documentation for this class was generated from the following files:

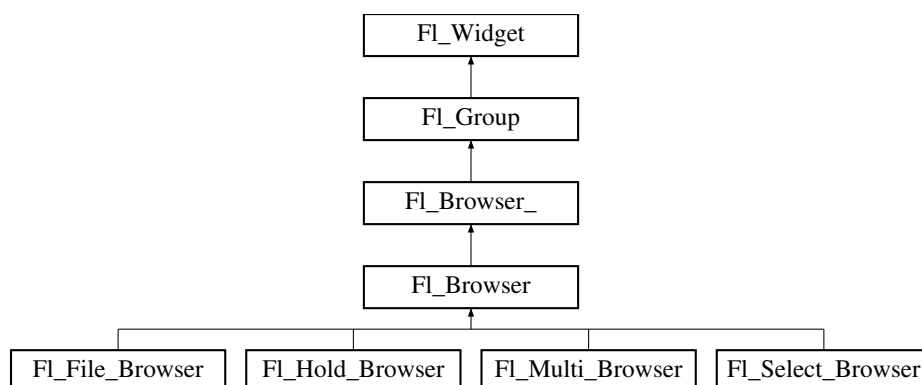
- [Fl_Box.H](#)
- [Fl_Box.cxx](#)

11.13 Fl_Browser Class Reference

The [Fl_Browser](#) widget displays a scrolling list of text lines, and manages all the storage for the text.

```
#include <Fl_Browser.H>
```

Inheritance diagram for Fl_Browser:



Public Types

- enum [Fl_Line_Position](#) { **TOP** , **BOTTOM** , **MIDDLE** }

For internal use only?

Public Types inherited from [FI_Browser_](#)

- enum {
[HORIZONTAL](#) = 1 , [VERTICAL](#) = 2 , [BOTH](#) = 3 , [ALWAYS_ON](#) = 4 ,
[HORIZONTAL_ALWAYS](#) = 5 , [VERTICAL_ALWAYS](#) = 6 , [BOTH_ALWAYS](#) = 7 }
 Values for [has_scrollbar\(\)](#).

Public Member Functions

- void [add](#) (const char *newtext, void *d=0)
 Adds a new line to the end of the browser.
- void [bottomline](#) (int line)
 Scrolls the browser so the bottom item in the browser is showing the specified *line*.
- void [clear](#) ()
 Removes all the lines in the browser.
- char [column_char](#) () const
 Gets the current column separator character.
- void [column_char](#) (char c)
 Sets the column separator to *c*.
- const int * [column_widths](#) () const
 Gets the current column width array.
- void [column_widths](#) (const int *arr)
 Sets the current array to *arr*.
- void * [data](#) (int line) const
 Returns the user [data\(\)](#) for specified *line*.
- void [data](#) (int line, void *d)
 Sets the user data for specified *line* to *d*.
- void [display](#) (int line, int val=1)
 For back compatibility.
- int [displayed](#) (int line) const
 Returns non-zero if *line* has been scrolled to a position where it is being displayed.
- [FI_Browser](#) (int X, int Y, int W, int H, const char *L=0)
 The constructor makes an empty browser.
- char [format_char](#) () const
 Gets the current format code prefix character, which by default is '@'.
- void [format_char](#) (char c)
 Sets the current format code prefix character to *c*.
- void [hide](#) () [FL_OVERRIDE](#)
 Hides the entire [FI_Browser](#) widget – opposite of [show\(\)](#).
- void [hide](#) (int line)
 Makes *line* invisible, preventing selection by the user.
- [FI_Image](#) * [icon](#) (int line) const
 Returns the icon currently defined for *line*.
- void [icon](#) (int line, [FI_Image](#) *icon)
 Set the image icon for *line* to the value *icon*.
- void [insert](#) (int line, const char *newtext, void *d=0)
 Insert a new entry whose label is *newtext* above given *line*, optional data *d*.
- void [lineposition](#) (int line, [FI_Line_Position](#) pos)
 Updates the browser so that *line* is shown at position *pos*.
- int [load](#) (const char *filename)
 Clears the browser and reads the file, adding each line from the file to the browser.
- void [make_visible](#) (int line)

- Make the item at the specified *line* [visible\(\)](#).
- void [middleline](#) (int line)

Scrolls the browser so the middle item in the browser is showing the specified *line*.
 - void [move](#) (int to, int from)

Line *from* is removed and reinserted at *to*.
 - void [remove](#) (int line)

Remove entry for given *line* number, making the browser one line shorter.
 - void [remove_icon](#) (int line)

Removes the icon for *line*.
 - void [replace](#) (int a, const char *b)

For back compatibility only.
 - int [select](#) (int line, int val=1)

Sets the selection state of the item at *line* to the value *val*.
 - int [selected](#) (int line) const

Returns 1 if specified *line* is selected, 0 if not.
 - void [show](#) () [FL_OVERRIDE](#)

Shows the entire [FL_Browser](#) widget – opposite of [hide\(\)](#).
 - void [show](#) (int line)

Makes *line* visible, and available for selection by user.
 - int [size](#) () const

Returns how many lines are in the browser.
 - void [size](#) (int W, int H)
 - void [swap](#) (int a, int b)

Swaps two browser lines *a* and *b*.
 - const char * [text](#) (int line) const

Returns the label text for the specified *line*.
 - void [text](#) (int line, const char *newtext)

Sets the text for the specified *line* to *newtext*.
 - [FL_Fontsize](#) [textsize](#) () const

Gets the default text size (in pixels) for the lines in the browser.
 - void [textsize](#) ([FL_Fontsize](#) newSize)

Sets the default text size (in pixels) for the lines in the browser to *newSize*.
 - int [topline](#) () const

Returns the line that is currently visible at the top of the browser.
 - void [topline](#) (int line)

Scrolls the browser so the top item in the browser is showing the specified *line*.
 - int [value](#) () const

Returns the line number of the currently selected line, or 0 if none selected.
 - void [value](#) (int line)

Sets the browser's [value\(\)](#), which selects the specified *line*.
 - int [visible](#) (int line) const

Returns non-zero if the specified *line* is visible, 0 if hidden.
 - [~FL_Browser](#) ()

The destructor deletes all list items and destroys the browser.

Public Member Functions inherited from [FI_Browser_](#)

- int [deselect](#) (int docallbacks=0)
Deselects all items in the list and returns 1 if the state changed or 0 if it did not.
- void [display](#) (void *item)
*Displays the *item*, scrolling the list as necessary.*
- int [handle](#) (int event) [FL_OVERRIDE](#)
*Handles the *event* within the normal widget bounding box.*
- [uchar](#) [has_scrollbar](#) () const
Returns the current scrollbar mode, see [FI_Browser_::has_scrollbar\(uchar\)](#)
- void [has_scrollbar](#) ([uchar](#) mode)
Sets whether the widget should have scrollbars or not (default [FI_Browser_::BOTH](#)).
- int [hposition](#) () const
*Gets the horizontal scroll position of the list as a pixel position *pos*.*
- void [hposition](#) (int)
*Sets the horizontal scroll position of the list to pixel position *pos*.*
- int [linespacing](#) () const
Return the height of additional spacing between browser lines.
- void [linespacing](#) (int pixels)
Add some space between browser lines.
- int [position](#) () const
- void [position](#) (int pos)
- void [position](#) (int *x*, int *y*)
- void [resize](#) (int *X*, int *Y*, int *W*, int *H*) [FL_OVERRIDE](#)
Repositions and/or resizes the browser.
- void [scrollbar_left](#) ()
Moves the vertical scrollbar to the lefthand side of the list.
- void [scrollbar_right](#) ()
Moves the vertical scrollbar to the righthand side of the list.
- int [scrollbar_size](#) () const
Gets the current size of the scrollbars' troughs, in pixels.
- void [scrollbar_size](#) (int newSize)
*Sets the pixel size of the scrollbars' troughs to *newSize*, in pixels.*
- int [scrollbar_width](#) () const
Returns the global value [FI::scrollbar_size\(\)](#).
- void [scrollbar_width](#) (int width)
Sets the global [FI::scrollbar_size\(\)](#), and forces this instance of the widget to use it.
- int [select](#) (void *item, int val=1, int docallbacks=0)
*Sets the selection state of *item* to *val*, and returns 1 if the state changed or 0 if it did not.*
- int [select_only](#) (void *item, int docallbacks=0)
*Selects *item* and returns 1 if the state changed or 0 if it did not.*
- void [sort](#) (int flags=0)
*Sort the items in the browser based on *flags*.*
- [FI_Color](#) [textcolor](#) () const
Gets the default text color for the lines in the browser.
- void [textcolor](#) ([FI_Color](#) col)
*Sets the default text color for the lines in the browser to color *col*.*
- [FI_Font](#) [textfont](#) () const
Gets the default text font for the lines in the browser.
- void [textfont](#) ([FI_Font](#) font)
*Sets the default text font for the lines in the browser to *font*.*

- **FL_Fontsize** **textsize** () const
Gets the default text size (in pixels) for the lines in the browser.
- void **textsize** (FL_Fontsize newSize)
*Sets the default text size (in pixels) for the lines in the browser to *size*.*
- int **vposition** () const
*Gets the vertical scroll position of the list as a pixel position *pos*.*
- void **vposition** (int pos)
*Sets the vertical scroll position of the list to pixel position *pos*.*

Public Member Functions inherited from FL_Group

- **FL_Widget** *& **_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** (FL_Widget &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** (FL_Widget *o)
See void FL_Group::add(FL_Widget &w)
- void **add_resizable** (FL_Widget &o)
Adds a widget to the group and makes it the resizable widget.
- **FL_Widget** *const * **array** () const
Returns a pointer to the internal array of children.
- **FL_Group** const * **as_group** () const **FL_OVERRIDE**
- **FL_Group** * **as_group** () **FL_OVERRIDE**
Returns an FL_Group pointer if this widget is an FL_Group.
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- **FL_Widget** * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
*Removes the widget at *index* from the group and deletes it.*
- void **end** ()
Exactly the same as current(this->parent()).
- int **find** (const FL_Widget &o) const
*See int FL_Group::find(const FL_Widget *w) const.*
- int **find** (const FL_Widget *) const
Searches the children for the widget and returns the index.
- **FL_Group** (int, int, int, int, const char * = 0)
Creates a new FL_Group widget using the given position, size, and label string.
- void **focus** (FL_Widget *W)
- void **forms_end** ()
This is for forms compatibility only.
- int **handle** (int) **FL_OVERRIDE**
Handles the specified event.

- void `init_sizes` ()
Resets the internal array of widget sizes and positions.
- void `insert` (`FL_Widget` &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void `insert` (`FL_Widget` &o, `FL_Widget` *before)
This does `insert(w, find(before))`.
- void `remove` (`FL_Widget` &)
Removes a widget from the group but does not delete it.
- void `remove` (`FL_Widget` *o)
Removes the widget o from the group.
- void `remove` (int index)
Removes the widget at index from the group but does not delete it.
- `FL_Widget` * `resizable` () const
Returns the group's resizable widget.
- void `resizable` (`FL_Widget` &o)
Sets the group's resizable widget.
- void `resizable` (`FL_Widget` *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- void `resize` (int, int, int, int) `FL_OVERRIDE`
Resizes the `FL_Group` widget and all of its children.
- virtual `~FL_Group` ()
The destructor also deletes all the children.

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
Activates the widget.
- unsigned int `active` () const
Returns whether the widget is active.
- int `active_r` () const
Returns whether the widget and all of its parents are active.
- `FL_Align` `align` () const
Gets the label alignment.
- void `align` (`FL_Align` alignment)
Sets the label alignment.
- long `argument` () const
Gets the current user data (long) argument that is passed to the callback function.
- void `argument` (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class `FL_Gl_Window` * `as_gl_window` ()
Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.
- virtual class `FL_Gl_Window` const * `as_gl_window` () const
- virtual `FL_Window` * `as_window` ()
Returns an `FL_Window` pointer if this widget is an `FL_Window`.
- virtual `FL_Window` const * `as_window` () const
- void `bind_deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- void `bind_deimage` (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.

- void [bind_image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FL_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FL_Boxtype](#) new_box)
Sets the box type for the widget.
- [FL_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FL_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FL_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) ([uchar](#) c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FL_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FL_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FL_Color](#) bg, [FL_Color](#) sel)
Sets the background and selection color of the widget.
- [FL_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FL_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const

- Returns non-zero if [draw\(\)](#) needs to be called.*
- void [damage](#) ([uchar](#) c)
 - Sets the damage bits for the widget.*
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
 - Sets the damage bits for an area inside the widget.*
- int [damage_resize](#) (int, int, int, int)
 - Internal use only.*
- void [deactivate](#) ()
 - Deactivates the widget.*
- [FL_Image](#) * [deimage](#) ()
 - Gets the image that is used as part of the widget label when in the inactive state.*
- const [FL_Image](#) * [deimage](#) () const
 - Gets the image that is used as part of the widget label when in the inactive state.*
- void [deimage](#) ([FL_Image](#) &img)
 - Sets the image to use as part of the widget label when in the inactive state.*
- void [deimage](#) ([FL_Image](#) *img)
 - Sets the image to use as part of the widget label when in the inactive state.*
- int [deimage_bound](#) () const
 - Returns whether the inactive image is managed by the widget.*
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
 - Calls the widget callback function with default arguments.*
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
 - Calls the widget callback function with arbitrary arguments.*
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
 - Calls the widget callback function with arbitrary arguments.*
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
 - Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int [h](#) () const
 - Gets the widget height.*
- int [horizontal_label_margin](#) ()
 - Get the spacing between the label and the horizontal edge of the widget.*
- void [horizontal_label_margin](#) (int px)
 - Set the spacing between the label and the horizontal edge of the widget.*
- [FL_Image](#) * [image](#) ()
 - Gets the image that is used as part of the widget label when in the active state.*
- const [FL_Image](#) * [image](#) () const
 - Gets the image that is used as part of the widget label when in the active state.*
- void [image](#) ([FL_Image](#) &img)
 - Sets the image to use as part of the widget label when in the active state.*
- void [image](#) ([FL_Image](#) *img)
 - Sets the image to use as part of the widget label when in the active state.*
- int [image_bound](#) () const
 - Returns whether the image is managed by the widget.*
- int [inside](#) (const [FL_Widget](#) *wgt) const
 - Checks if this widget is a child of wgt.*
- int [is_label_copied](#) () const
 - Returns whether the current label was assigned with [copy_label\(\)](#).*
- const char * [label](#) () const
 - Gets the current label text.*
- void [label](#) (const char *text)
 - Sets the current label pointer.*

- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [FL_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()

- Makes the widget visible.*

 - void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

Gets the current tooltip text.
- void [tooltip](#) (const char *text)

Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const

Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const

Gets the widget type.
- void [type](#) ([uchar](#) t)

Sets the widget type.
- int [use_accents_menu](#) ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const

Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)

Sets the user data for this widget.
- void [user_data](#) (void *v)

Sets the user data for this widget.
- int [vertical_label_margin](#) ()

Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const

Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const

Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)

Modifies keyboard focus navigation.
- int [visible_r](#) () const

Returns whether a widget and all its parents are visible.
- int [w](#) () const

Gets the widget width.
- [Fl_When](#) [when](#) () const

Returns the conditions under which the callback is called.

- void `when` (uchar i)
Sets the flags used to decide when a callback is called.
- `FL_Window * window` () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual `~FL_Widget` ()
Destroys the widget.

Protected Member Functions

- `FL_BLINE * _remove` (int line)
Removes the item at the specified line.
- `FL_BLINE * find_line` (int line) const
Returns the item for specified line.
- int `full_height` () const `FL_OVERRIDE`
The height of the entire list of all `visible()` items in pixels.
- int `incr_height` () const `FL_OVERRIDE`
The default 'average' item height (including inter-item spacing) in pixels.
- void `insert` (int line, `FL_BLINE *item`)
Insert specified item above line.
- void * `item_at` (int line) const `FL_OVERRIDE`
Return the item at specified line.
- void `item_draw` (void *item, int X, int Y, int W, int H) const `FL_OVERRIDE`
Draws item at the position specified by X Y W H.
- void * `item_first` () const `FL_OVERRIDE`
Returns the very first item in the list.
- int `item_height` (void *item) const `FL_OVERRIDE`
Returns height of item in pixels.
- void * `item_last` () const `FL_OVERRIDE`
Returns the very last item in the list.
- void * `item_next` (void *item) const `FL_OVERRIDE`
Returns the next item after item.
- void * `item_prev` (void *item) const `FL_OVERRIDE`
Returns the previous item before item.
- void `item_select` (void *item, int val) `FL_OVERRIDE`
Change the selection state of item to the value val.
- int `item_selected` (void *item) const `FL_OVERRIDE`
See if item is selected.
- void `item_swap` (void *a, void *b) `FL_OVERRIDE`
Swap the items a and b.
- const char * `item_text` (void *item) const `FL_OVERRIDE`
Returns the label text for item.
- int `item_width` (void *item) const `FL_OVERRIDE`
Returns width of item in pixels.
- int `lineno` (void *item) const
Returns line number corresponding to item, or zero if not found.
- void `swap` (`FL_BLINE *a`, `FL_BLINE *b`)
Swap the two items a and b.

Protected Member Functions inherited from [FI_Browser_](#)

- void [bbox](#) (int &X, int &Y, int &W, int &H) const
Returns the bounding box for the interior of the list's display window, inside the scrollbars.
- void [deleting](#) (void *item)
*This method should be used when *item* is being deleted from the list.*
- int [displayed](#) (void *item) const
*Returns non-zero if *item* has been scrolled to a position where it is being displayed.*
- void [draw](#) () [FL_OVERRIDE](#)
Draws the list within the normal widget bounding box.
- void * [find_item](#) (int ypos)
*This method returns the item under mouse y position *ypos*.*
- [FI_Browser_](#) (int X, int Y, int W, int H, const char *L=0)
The constructor makes an empty browser.
- virtual int [full_width](#) () const
This method may be provided by the subclass to indicate the full width of the item list, in pixels.
- void [inserting](#) (void *a, void *b)
This method should be used when an item is in the process of being inserted into the list.
- virtual int [item_quick_height](#) (void *item) const
*This method may be provided by the subclass to return the height of the *item*, in pixels.*
- int [leftedge](#) () const
This method returns the X position of the left edge of the list area after adjusting for the scrollbar and border, if any.
- void [new_list](#) ()
This method should be called when the list data is completely replaced or cleared.
- void [redraw_line](#) (void *item)
*This method should be called when the contents of *item* has changed, but not its height.*
- void [redraw_lines](#) ()
This method will cause the entire list to be redrawn.
- void [replacing](#) (void *a, void *b)
*This method should be used when item *a* is being replaced by item *b*.*
- void * [selection](#) () const
Returns the item currently selected, or NULL if there is no selection.
- void [swapping](#) (void *a, void *b)
*This method should be used when two items *a* and *b* are being swapped.*
- void * [top](#) () const
Returns the item that appears at the top of the list.

Protected Member Functions inherited from [FI_Group](#)

- [FI_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void [draw_child](#) ([FI_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [FI_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) ([FI_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int [on_move](#) (int, int)

- *Allow derived groups to act when a widget is moved within the group.*
- virtual void **on_remove** (int)
 - *Allow derived groups to act when a child widget is removed from the group.*
- int * **sizes** ()
 - *Returns the internal array of widget sizes and positions.*
- void **update_child** (FI_Widget &widget) const
 - *Draws a child only if it needs it.*

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
 - *Clears a flag in the flags mask.*
- void **draw_backdrop** () const
 - *If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.*
- void **draw_box** () const
 - *Draws the widget box according its box style.*
- void **draw_box** (FI_Boxtype t, FI_Color c) const
 - *Draws a box of type t, of color c at the widget's position and size.*
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
 - *Draws a box of type t, of color c at the position X,Y and size W,H.*
- void **draw_focus** () const
 - *Draws a focus rectangle around the widget.*
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
 - *Draws a focus rectangle around the widget.*
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
 - *Draws a focus box for the widget at the given position and size.*
- void **draw_label** () const
 - *Draws the widget's label at the defined label position.*
- void **draw_label** (int, int, int, int) const
 - *Draws the label in an arbitrary bounding box.*
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
 - *Creates a widget at the given position and size.*
- unsigned int **flags** () const
 - *Gets the widget flags mask.*
- void **h** (int v)
 - *Internal use only.*
- void **set_flag** (unsigned int c)
 - *Sets a flag in the flags mask.*
- void **w** (int v)
 - *Internal use only.*
- void **x** (int v)
 - *Internal use only.*
- void **y** (int v)
 - *Internal use only.*

Additional Inherited Members

Static Public Member Functions inherited from FI_Group

- static FI_Group * **current** ()
 - *Returns the currently active group.*
- static void **current** (FI_Group *g)
 - *Sets the current group.*

Static Public Member Functions inherited from [FI_Widget](#)

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Public Attributes inherited from [FI_Browser_](#)

- [FI_Scrollbar](#) hscrollbar
Horizontal scrollbar.
- [FI_Scrollbar](#) scrollbar
Vertical scrollbar.

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.13.1 Detailed Description

The [FI_Browser](#) widget displays a scrolling list of text lines, and manages all the storage for the text. This is not a text editor or spreadsheet! But it is useful for showing a vertical list of named objects to the user.



Figure 11.2 [FI_Hold_Browser](#)



Figure 11.3 [FI_Multi_Browser](#)

Each line in the browser is identified by number. *The numbers start at one* (this is so that zero can be reserved for "no line" in the selective browsers). *Unless otherwise noted, the methods do not check to see if the passed line number is in range and legal. It must always be greater than zero and $\leq$ `size()`.*

Each line contains a null-terminated string of text and a void * data pointer. The text string is displayed, the void * pointer can be used by the callbacks to reference the object the text describes.

The base class does nothing when the user clicks on it. The subclasses [Fl_Select_Browser](#), [Fl_Hold_Browser](#), and [Fl_Multi_Browser](#) react to user clicks to select lines in the browser and do callbacks.

The base class [Fl_Browser_](#) provides the scrolling and selection mechanisms of this and all the subclasses, but the dimensions and appearance of each item are determined by the subclass. You can use [Fl_Browser_](#) to display information other than text, or text that is dynamically produced from your own data structures. If you find that loading the browser is a lot of work or is inefficient, you may want to make a subclass of [Fl_Browser_](#).

Some common coding patterns used for working with [Fl_Browser](#):

```
// How to loop through all the items in the browser
for ( int t=1; t<=browser->size(); t++ ) {           // index 1 based..!
    printf("item #%d, label='%s'\n", t, browser->text(t));
}
```

Note: If you are *subclassing* [Fl_Browser](#), it's more efficient to use the protected methods [item_first\(\)](#) and [item_next\(\)](#), since [Fl_Browser](#) internally uses linked lists to manage the browser's items. For more info, see [find_item\(int\)](#).

11.13.2 Constructor & Destructor Documentation

11.13.2.1 Fl_Browser()

```
Fl_Browser::Fl_Browser (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

The constructor makes an empty browser.

Parameters

in	<i>X,Y,W,H</i>	position and size.
in	<i>L</i>	label string, may be NULL.

11.13.3 Member Function Documentation

11.13.3.1 _remove()

```
FL_BLINE * Fl_Browser::_remove (
    int line) [protected]
```

Removes the item at the specified line.

Caveat: See efficiency note in [find_line\(\)](#). You must call [redraw\(\)](#) to make any changes visible.

Parameters

in	<i>line</i>	The line number to be removed. (1 based) Must be in range!

Returns

Pointer to browser item that was removed (and is no longer valid).

See also

[add\(\)](#), [insert\(\)](#), [remove\(\)](#), [swap\(int,int\)](#), [clear\(\)](#)

11.13.3.2 add()

```
void Fl_Browser::add (
    const char * newtext,
    void * d = 0)
```

Adds a new line to the end of the browser.

The text string `newtext` may contain format characters; see [format_char\(\)](#) for details. `newtext` is copied using the `strdup()` function, and can be NULL to make a blank line.

The optional `void*` argument `d` will be the [data\(\)](#) for the new item.

Parameters

in	<i>newtext</i>	The label text used for the added item
in	<i>d</i>	Optional user data() for the item (0 if unspecified)

See also

[add\(\)](#), [insert\(\)](#), [remove\(\)](#), [swap\(int,int\)](#), [clear\(\)](#)

11.13.3.3 bottomline()

```
void Fl_Browser::bottomline (
    int line) [inline]
```

Scrolls the browser so the bottom item in the browser is showing the specified `line`.

Parameters

in	<i>line</i>	The line to be displayed at the bottom.

See also

[topline\(\)](#), [middleline\(\)](#), [bottomline\(\)](#), [displayed\(\)](#), [lineposition\(\)](#)

11.13.3.4 clear()

```
void Fl_Browser::clear ()
```

Removes all the lines in the browser.

See also

[add\(\)](#), [insert\(\)](#), [remove\(\)](#), [swap\(int,int\)](#), [clear\(\)](#)

11.13.3.5 column_char() [1/2]

```
char Fl_Browser::column_char () const [inline]
```

Gets the current column separator character.

The default is '\t' (tab).

See also

[column_char\(\)](#), [column_widths\(\)](#)

11.13.3.6 column_char() [2/2]

```
void Fl_Browser::column_char (
    char c) [inline]
```

Sets the column separator to `c`.

This will only have an effect if you also set [column_widths\(\)](#). The default is '\t' (tab).

See also

[column_char\(\)](#), [column_widths\(\)](#)

11.13.3.7 column_widths() [1/2]

```
const int * Fl_Browser::column_widths () const [inline]
```

Gets the current column width array.

This array is zero-terminated and specifies the widths in pixels of each column. The text is split at each [column_char\(\)](#) and each part is formatted into it's own column. After the last column any remaining text is formatted into the space between the last column and the right edge of the browser, even if the text contains instances of [column_char\(\)](#) . The default value is a one-element array of just a zero, which means there are no columns.

Example:

```
Fl_Browser *b = new Fl_Browser(..);
static int widths[] = { 50, 50, 50, 70, 70, 40, 40, 70, 70, 50, 0 }; // widths for each column
b->column_widths(widths); // assign array to widget
b->column_char('\t'); // use tab as the column character
b->add("USER\tPID\tCPU\tMEM\tVSZ\tRSS\tTTY\tSTAT\tSTART\tTIME\tCOMMAND");
b->add("root\t2888\t0.0\t0.0\t1352\t0\ttty3\tSW\tAug15\t0:00\t@b@f/sbin/mingetty tty3");
b->add("root\t13115\t0.0\t0.0\t1352\t0\ttty2\tSW\tAug30\t0:00\t@b@f/sbin/mingetty tty2");
[...]
```

See also

[column_char\(\)](#), [column_widths\(\)](#)

11.13.3.8 column_widths() [2/2]

```
void Fl_Browser::column_widths (
    const int * arr) [inline]
```

Sets the current array to `arr`.

Make sure the last entry is zero.

See also

[column_char\(\)](#), [column_widths\(\)](#)

11.13.3.9 data() [1/2]

```
void * Fl_Browser::data (
    int line) const
```

Returns the user [data\(\)](#) for specified `line`.

Return value can be NULL if `line` is out of range or no user [data\(\)](#) was defined. The parameter `line` is 1 based (1 will be the first item in the list).

Parameters

in	<i>line</i>	The line number of the item whose data() is returned. (1 based)

Returns

The user data pointer (can be NULL)

11.13.3.10 data() [2/2]

```
void Fl_Browser::data (
    int line,
    void * d)
```

Sets the user data for specified `line` to `d`.

Does nothing if `line` is out of range.

Parameters

in	<i>line</i>	The line of the item whose data() is to be changed. (1 based)
in	<i>d</i>	The new data to be assigned to the item. (can be NULL)

11.13.3.11 display()

```
void Fl_Browser::display (
    int line,
    int val = 1)
```

For back compatibility.

This calls `show(line)` if `val` is true, and `hide(line)` otherwise. If `val` is not specified, the default is 1 (makes the line visible).

See also

[show\(int\)](#), [hide\(int\)](#), [display\(\)](#), [visible\(\)](#), [make_visible\(\)](#)

11.13.3.12 displayed()

```
int Fl_Browser::displayed (
    int line) const [inline]
```

Returns non-zero if `line` has been scrolled to a position where it is being displayed.

Checks to see if the item's vertical position is within the top and bottom edges of the display window. This does NOT take into account the [hide\(\)](#)/[show\(\)](#) status of the widget or item.

Parameters

<i>in</i>	<i>line</i>	The line to be checked

Returns

1 if visible, 0 if not visible.

See also

[topline\(\)](#), [middleline\(\)](#), [bottomline\(\)](#), [displayed\(\)](#), [lineposition\(\)](#)

11.13.3.13 find_line()

```
FL_BLINE * Fl_Browser::find_line (
    int line) const [protected]
```

Returns the item for specified `line`.

Note: This call is slow. It's fine for e.g. responding to user clicks, but slow if called often, such as in a tight sorting loop. Finding an item 'by line' involves a linear lookup on the internal linked list. The performance hit can be significant if the browser's contents is large, and the method is called often (e.g. during a sort). If you're writing a subclass, use the protected methods [item_first\(\)](#), [item_next\(\)](#), etc. to access the internal linked list more efficiently.

Parameters

<i>in</i>	<i>line</i>	The line number of the item to return. (1 based)

Return values

<i>item</i>	that was found.
<i>NULL</i>	if line is out of range.

See also

[item_at\(\)](#), [find_line\(\)](#), [lineno\(\)](#)

11.13.3.14 format_char() [1/2]

```
char Fl_Browser::format_char () const [inline]
```

Gets the current format code prefix character, which by default is '@'.

A string of formatting codes at the start of each column are stripped off and used to modify how the rest of the line is printed:

- '@.' Print rest of line, don't look for more '@' signs
- '@@' Doubling the format character prints the format character once, followed by the rest of line
- '@l' Use a LARGE (24 point) font
- '@m' Use a medium large (18 point) font
- '@s' Use a small (11 point) font
- '@b' Use a **bold** font (adds FL_BOLD to font)
- '@i' Use an *italic* font (adds FL_ITALIC to font)
- '@f' or '@t' Use a fixed-pitch font (sets font to FL_COURIER)
- '@c' Center the line horizontally
- '@r' Right-justify the text
- '@N' Use fl_inactive_color() to draw the text
- '@B0', '@B1', ... '@B255' Fill the background with fl_color(n)
- '@C0', '@C1', ... '@C255' Use fl_color(n) to draw the text
- '@F0', '@F1', ... Use fl_font(n) to draw the text
- '@S1', '@S2', ... Use point size n to draw the text
- '@u' or '@_' Underline the text.
- '@-' draw an engraved line through the middle.

Notice that the '@.' command can be used to reliably terminate the parsing. To print a random string in a random color, use `sprintf("@C%d@.%s", color, string)` and it will work even if the string starts with a digit or has the format character in it.

11.13.3.15 format_char() [2/2]

```
void Fl_Browser::format_char (
    char c) [inline]
```

Sets the current format code prefix character to c.

The default prefix is '@'. Set the prefix to 0 to disable formatting.

See also

[format_char\(\)](#) for list of '@' codes

11.13.3.16 full_height()

```
int Fl_Browser::full_height () const [protected], [virtual]
```

The height of the entire list of all [visible\(\)](#) items in pixels.

This returns the accumulated height of *all* the items in the browser that are not hidden with [hide\(\)](#), including items scrolled off screen.

Returns

The accumulated size of all the visible items in pixels.

See also

[item_height\(\)](#), [item_width\(\)](#),
[incr_height\(\)](#), [full_height\(\)](#)

Reimplemented from [Fl_Browser_](#).

11.13.3.17 [hide\(\)](#) [1/2]

```
void Fl_Browser::hide () [inline], [virtual]
```

Hides the entire [Fl_Browser](#) widget – opposite of [show\(\)](#).

Reimplemented from [Fl_Widget](#).

11.13.3.18 [hide\(\)](#) [2/2]

```
void Fl_Browser::hide (
    int line)
```

Makes `line` invisible, preventing selection by the user.

The line can still be selected under program control. This changes the [full_height\(\)](#) if the state was changed. When a line is made invisible, lines below it are moved up in the display. [redraw\(\)](#) is called automatically if a change occurred.

Parameters

<code>in</code>	<i>line</i>	The line to be hidden. (1 based)

See also

[show\(int\)](#), [hide\(int\)](#), [display\(\)](#), [visible\(\)](#), [make_visible\(\)](#)

11.13.3.19 [icon\(\)](#) [1/2]

```
Fl_Image * Fl_Browser::icon (
    int line) const
```

Returns the icon currently defined for `line`.

If no icon is defined, NULL is returned.

Parameters

<code>in</code>	<i>line</i>	The line whose icon is returned.

Returns

The icon defined, or NULL if none.

11.13.3.20 [icon\(\)](#) [2/2]

```
void Fl_Browser::icon (
    int line,
    Fl_Image * icon)
```

Set the image icon for `line` to the value `icon`.

Caller is responsible for keeping the icon allocated. The `line` is automatically redrawn.

Parameters

<code>in</code>	<i>line</i>	The line to be modified. If out of range, nothing is done.
<code>in</code>	<i>icon</i>	The image icon to be assigned to the <code>line</code> . If NULL, any previous icon is removed.

11.13.3.21 incr_height()

```
int Fl_Browser::incr_height () const [protected], [virtual]
```

The default 'average' item height (including inter-item spacing) in pixels.

This currently returns [textsize\(\)](#) + 2.

Returns

The value in pixels.

See also

[item_height\(\)](#), [item_width\(\)](#),
[incr_height\(\)](#), [full_height\(\)](#)

Reimplemented from [Fl_Browser_](#).

11.13.3.22 insert() [1/2]

```
void Fl_Browser::insert (
    int line,
    const char * newtext,
    void * d = 0)
```

Insert a new entry whose label is *newtext* *above* given *line*, optional data *d*.

Text may contain format characters; see [format_char\(\)](#) for details. *newtext* is copied using the `strdup()` function, and can be NULL to make a blank line.

The optional void * argument *d* will be the [data\(\)](#) of the new item.

Parameters

in	<i>line</i>	Line position for insert. (1 based) If <i>line</i> > size() , the entry will be added at the end.
in	<i>newtext</i>	The label text for the new line.
in	<i>d</i>	Optional pointer to user data to be associated with the new line.

11.13.3.23 insert() [2/2]

```
void Fl_Browser::insert (
    int line,
    FL_BLINE * item) [protected]
```

Insert specified *item* *above* *line*.

If *line* > [size\(\)](#) then the line is added to the end.

Caveat: See efficiency note in [find_line\(\)](#).

Parameters

in	<i>line</i>	The new line will be inserted above this line (1 based).
in	<i>item</i>	The item to be added.

11.13.3.24 item_at()

```
void * Fl_Browser::item_at (
    int line) const [inline], [protected], [virtual]
```

Return the item at specified *line*.

Parameters

in	<i>line</i>	The line of the item to return. (1 based)

Returns

The item, or NULL if line out of range.

See also

[item_at\(\)](#), [find_line\(\)](#), [lineno\(\)](#)

Reimplemented from [Fl_Browser_](#).

11.13.3.25 item_draw()

```
void Fl_Browser::item_draw (
    void * item,
    int X,
    int Y,
    int W,
    int H) const [protected], [virtual]
```

Draws *item* at the position specified by X Y W H.

The W and H values are used for clipping. Should only be called within the context of an FLTK [draw\(\)](#).

Parameters

in	<i>item</i>	The item to be drawn
in	<i>X,Y,W,H</i>	position and size.

Implements [Fl_Browser_](#).

11.13.3.26 item_first()

```
void * Fl_Browser::item_first () const [protected], [virtual]
```

Returns the very first item in the list.

Example of use:

```
// Walk the browser from beginning to end
for ( void *i=item_first(); i; i=item_next(i) ) {
    printf("item label='%s'\n", item_text(i));
}
```

Returns

The first item, or NULL if list is empty.

See also

[item_first\(\)](#), [item_last\(\)](#), [item_next\(\)](#), [item_prev\(\)](#)

Implements [Fl_Browser_](#).

11.13.3.27 item_height()

```
int Fl_Browser::item_height (
    void * item) const [protected], [virtual]
```

Returns height of *item* in pixels.

This takes into account embedded @ codes within the [text\(\)](#) label.

Parameters

in	<i>item</i>	The item whose height is returned.

Returns

The height of the item in pixels.

See also

[item_height\(\)](#), [item_width\(\)](#),
[incr_height\(\)](#), [full_height\(\)](#)

Implements [Fl_Browser_](#).

11.13.3.28 item_last()

```
void * Fl_Browser::item_last () const [protected], [virtual]
```

Returns the very last item in the list.

Example of use:

```
// Walk the browser in reverse, from end to start
for ( void *i=item_last(); i; i=item_prev(i) ) {
    printf("item label='%s'\n", item_text(i));
}
```

Returns

The last item, or NULL if list is empty.

See also

[item_first\(\)](#), [item_last\(\)](#), [item_next\(\)](#), [item_prev\(\)](#)

Reimplemented from [Fl_Browser_](#).

11.13.3.29 item_next()

```
void * Fl_Browser::item_next (
    void * item) const [protected], [virtual]
```

Returns the next item after *item*.

Parameters

<i>in</i>	<i>item</i>	The 'current' item

Returns

The next item after *item*, or NULL if there are none after this one.

See also

[item_first\(\)](#), [item_last\(\)](#), [item_next\(\)](#), [item_prev\(\)](#)

Implements [Fl_Browser_](#).

11.13.3.30 item_prev()

```
void * Fl_Browser::item_prev (
    void * item) const [protected], [virtual]
```

Returns the previous item before *item*.

Parameters

<i>in</i>	<i>item</i>	The 'current' item

Returns

The previous item before *item*, or NULL if there are none before this one.

See also

[item_first\(\)](#), [item_last\(\)](#), [item_next\(\)](#), [item_prev\(\)](#)

Implements [Fl_Browser_](#).

11.13.3.31 item_select()

```
void Fl_Browser::item_select (
    void * item,
    int val) [protected], [virtual]
```

Change the selection state of *item* to the value *val*.

Parameters

in	<i>item</i>	The item to be changed.
in	<i>val</i>	The new selection state: 1 selects, 0 de-selects.

See also

[select\(\)](#), [selected\(\)](#), [value\(\)](#), [item_select\(\)](#), [item_selected\(\)](#)

Reimplemented from [Fl_Browser_](#).

11.13.3.32 item_selected()

```
int Fl_Browser::item_selected (
    void * item) const [protected], [virtual]
```

See if *item* is selected.

Parameters

in	<i>item</i>	The item whose selection state is to be checked.

Returns

1 if selected, 0 if not.

See also

[select\(\)](#), [selected\(\)](#), [value\(\)](#), [item_select\(\)](#), [item_selected\(\)](#)

Reimplemented from [Fl_Browser_](#).

11.13.3.33 item_swap()

```
void Fl_Browser::item_swap (
    void * a,
    void * b) [inline], [protected], [virtual]
```

Swap the items *a* and *b*.

You must call [redraw\(\)](#) to make any changes visible.

Parameters

in	<i>a,b</i>	the items to be swapped.

See also

[swap\(int,int\)](#), [item_swap\(\)](#)

Reimplemented from [Fl_Browser_](#).

11.13.3.34 item_text()

```
const char * Fl_Browser::item_text (
    void * item) const [protected], [virtual]
```

Returns the label text for *item*.

Parameters

<i>in</i>	<i>item</i>	The item whose label text is returned.

Returns

The item's text string. (Can be NULL)

Reimplemented from [Fl_Browser_](#).

11.13.3.35 item_width()

```
int Fl_Browser::item_width (
    void * item) const [protected], [virtual]
```

Returns width of *item* in pixels.

This takes into account embedded @ codes within the [text\(\)](#) label.

Parameters

<i>in</i>	<i>item</i>	The item whose width is returned.

Returns

The width of the item in pixels.

See also

[item_height\(\)](#), [item_width\(\)](#),
[incr_height\(\)](#), [full_height\(\)](#)

Implements [Fl_Browser_](#).

11.13.3.36 lineno()

```
int Fl_Browser::lineno (
    void * item) const [protected]
```

Returns line number corresponding to *item*, or zero if not found.

Caveat: See efficiency note in [find_line\(\)](#).

Parameters

<i>in</i>	<i>item</i>	The item to be found

Returns

The line number of the item, or 0 if not found.

See also

[item_at\(\)](#), [find_line\(\)](#), [lineno\(\)](#)

11.13.3.37 lineposition()

```
void Fl_Browser::lineposition (
    int line,
    Fl\_Line\_Position pos)
```

Updates the browser so that *line* is shown at position *pos*.

Parameters

in	<i>line</i>	line number. (1 based)
in	<i>pos</i>	position.

See also

[topline\(\)](#), [middleline\(\)](#), [bottomline\(\)](#)

11.13.3.38 load()

```
int Fl_Browser::load (
    const char * filename)
```

Clears the browser and reads the file, adding each line from the file to the browser.

If the filename is NULL or a zero-length string then this just clears the browser. This returns zero if there was any error in opening or reading the file, in which case `errno` is set to the system error. The [data\(\)](#) of each line is set to NULL.

Parameters

in	<i>filename</i>	The filename to load

Returns

1 if OK, 0 on error (`errno` has reason)

See also

[add\(\)](#)

11.13.3.39 make_visible()

```
void Fl_Browser::make_visible (
    int line) [inline]
```

Make the item at the specified `line` [visible\(\)](#).

Functionally similar to [show\(int line\)](#). If `line` is out of range, redisplay top or bottom of list as appropriate.

Parameters

in	<i>line</i>	The line to be made visible.

See also

[show\(int\)](#), [hide\(int\)](#), [display\(\)](#), [visible\(\)](#), [make_visible\(\)](#)

11.13.3.40 middleline()

```
void Fl_Browser::middleline (
    int line) [inline]
```

Scrolls the browser so the middle item in the browser is showing the specified `line`.

Parameters

in	<i>line</i>	The line to be displayed in the middle.

See also

[topline\(\)](#), [middleline\(\)](#), [bottomline\(\)](#), [displayed\(\)](#), [lineposition\(\)](#)

11.13.3.41 move()

```
void Fl_Browser::move (
    int to,
    int from)
```

Line `from` is removed and reinserted at `to`.

Note: `to` is calculated *after* line `from` gets removed.

Parameters

in	<i>to</i>	Destination line number (calculated <i>after</i> line <code>from</code> is removed)
in	<i>from</i>	Line number of item to be moved

11.13.3.42 remove()

```
void Fl_Browser::remove (
    int line)
```

Remove entry for given `line` number, making the browser one line shorter.

You must call [redraw\(\)](#) to make any changes visible.

Parameters

in	<i>line</i>	Line to be removed. (1 based) If <code>line</code> is out of range, no action is taken.

See also

[add\(\)](#), [insert\(\)](#), [remove\(\)](#), [swap\(int,int\)](#), [clear\(\)](#)

11.13.3.43 remove_icon()

```
void Fl_Browser::remove_icon (
    int line)
```

Removes the icon for `line`.

It's ok to remove an icon if none has been defined.

Parameters

in	<i>line</i>	The line whose icon is to be removed.

11.13.3.44 select()

```
int Fl_Browser::select (
    int line,
    int val = 1)
```

Sets the selection state of the item at `line` to the value `val`.

If `val` is not specified, the default is 1 (selects the item).

Parameters

in	<i>line</i>	The line number of the item to be changed. (1 based)
in	<i>val</i>	The new selection state (1=select, 0=de-select).

Returns

1 if the state changed, 0 if not.

See also

[select\(\)](#), [selected\(\)](#), [value\(\)](#), [item_select\(\)](#), [item_selected\(\)](#)

11.13.3.45 selected()

```
int Fl_Browser::selected (
    int line) const
```

Returns 1 if specified `line` is selected, 0 if not.

Parameters

<code>in</code>	<i>line</i>	The line being checked (1 based)

Returns

1 if item selected, 0 if not.

See also

[select\(\)](#), [selected\(\)](#), [value\(\)](#), [item_select\(\)](#), [item_selected\(\)](#)

11.13.3.46 show() [1/2]

```
void Fl_Browser::show () [inline], [virtual]
```

Shows the entire [Fl_Browser](#) widget – opposite of [hide\(\)](#).
Reimplemented from [Fl_Widget](#).

11.13.3.47 show() [2/2]

```
void Fl_Browser::show (
    int line)
```

Makes `line` visible, and available for selection by user.

Opposite of [hide\(int\)](#). This changes the [full_height\(\)](#) if the state was changed. [redraw\(\)](#) is called automatically if a change occurred.

Parameters

<code>in</code>	<i>line</i>	The line to be shown. (1 based)

See also

[show\(int\)](#), [hide\(int\)](#), [display\(\)](#), [visible\(\)](#), [make_visible\(\)](#)

11.13.3.48 size()

```
int Fl_Browser::size () const [inline]
```

Returns how many lines are in the browser.

The last line number is equal to this. Returns 0 if browser is empty.

11.13.3.49 swap() [1/2]

```
void Fl_Browser::swap (
    FL_BLINE * a,
    FL_BLINE * b) [protected]
```

Swap the two items `a` and `b`.

Uses [swapping\(\)](#) to ensure list updates correctly.

Parameters

in	<i>a,b</i>	The two items to be swapped.

See also

[swap\(int,int\)](#), [item_swap\(\)](#)

11.13.3.50 swap() [2/2]

```
void Fl_Browser::swap (
    int a,
    int b)
```

Swaps two browser lines *a* and *b*.

You must call [redraw\(\)](#) to make any changes visible.

Parameters

in	<i>a,b</i>	The two lines to be swapped. (both 1 based)

See also

[swap\(int,int\)](#), [item_swap\(\)](#)

11.13.3.51 text() [1/2]

```
const char * Fl_Browser::text (
    int line) const
```

Returns the label text for the specified *line*.

Return value can be NULL if *line* is out of range or unset. The parameter *line* is 1 based.

Parameters

in	<i>line</i>	The line number of the item whose text is returned. (1 based)

Returns

The text string (can be NULL)

11.13.3.52 text() [2/2]

```
void Fl_Browser::text (
    int line,
    const char * newtext)
```

Sets the text for the specified *line* to *newtext*.

Text may contain format characters; see [format_char\(\)](#) for details. *newtext* is copied using the `strdup()` function, and can be NULL to make a blank line.

Does nothing if *line* is out of range.

Parameters

in	<i>line</i>	The line of the item whose text will be changed. (1 based)
in	<i>newtext</i>	The new string to be assigned to the item.

11.13.3.53 `textsize()`

```
void Fl_Browser::textsize (
    Fl_Fontsize newSize)
```

Sets the default text size (in pixels) for the lines in the browser to `newSize`.

This method recalculates all item heights and caches the total height internally for optimization of later item changes.

This can be slow if there are many items in the browser.

It returns immediately (w/o recalculation) if `newSize` equals the current `textsize()`.

You *may* need to call `redraw()` to see the effect and to have the scrollbar positions recalculated.

You should set the text size *before* populating the browser with items unless you really need to *change* the size later.

11.13.3.54 `topline()` [1/2]

```
int Fl_Browser::topline () const
```

Returns the line that is currently visible at the top of the browser.

If there is no vertical scrollbar then this will always return 1.

Returns

The `lineno()` of the `top()` of the browser.

11.13.3.55 `topline()` [2/2]

```
void Fl_Browser::topline (
    int line) [inline]
```

Scrolls the browser so the top item in the browser is showing the specified `line`.

Parameters

<code>in</code>	<code>line</code>	The line to be displayed at the top.

See also

`topline()`, `middleline()`, `bottomline()`, `displayed()`, `lineposition()`

11.13.3.56 `value()` [1/2]

```
int Fl_Browser::value () const
```

Returns the line number of the currently selected line, or 0 if none selected.

Returns

The line number of current selection, or 0 if none selected.

See also

`select()`, `selected()`, `value()`, `item_select()`, `item_selected()`

11.13.3.57 `value()` [2/2]

```
void Fl_Browser::value (
    int line) [inline]
```

Sets the browser's `value()`, which selects the specified `line`.

This is the same as calling `select(line)`.

See also

`select()`, `selected()`, `value()`, `item_select()`, `item_selected()`

11.13.3.58 visible()

```
int Fl_Browser::visible (
    int line) const
```

Returns non-zero if the specified `line` is visible, 0 if hidden.

Use [show\(int\)](#), [hide\(int\)](#), or [make_visible\(int\)](#) to change an item's visible state.

Parameters

in	<i>line</i>	The line in the browser to be tested. (1 based)

See also

[show\(int\)](#), [hide\(int\)](#), [display\(\)](#), [visible\(\)](#), [make_visible\(\)](#)

The documentation for this class was generated from the following files:

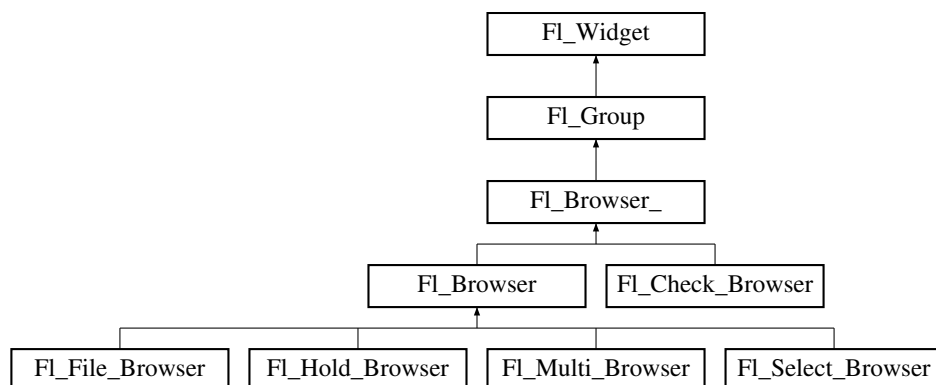
- Fl_Browser.H
- Fl_Browser.cxx
- Fl_Browser_load.cxx

11.14 Fl_Browser_ Class Reference

This is the base class for browsers.

```
#include <Fl_Browser_.H>
```

Inheritance diagram for Fl_Browser_:



Public Types

- enum {
[HORIZONTAL](#) = 1 , [VERTICAL](#) = 2 , [BOTH](#) = 3 , [ALWAYS_ON](#) = 4 ,
[HORIZONTAL_ALWAYS](#) = 5 , [VERTICAL_ALWAYS](#) = 6 , [BOTH_ALWAYS](#) = 7 }
 Values for [has_scrollbar\(\)](#).

Public Member Functions

- int [deselect](#) (int docallbacks=0)
 Deselects all items in the list and returns 1 if the state changed or 0 if it did not.
- void [display](#) (void *item)
 Displays the *item*, scrolling the list as necessary.
- int [handle](#) (int event) [FL_OVERRIDE](#)
 Handles the *event* within the normal widget bounding box.
- uchar [has_scrollbar](#) () const

- Returns the current scrollbar mode, see [Fl_Browser_::has_scrollbar\(uchar\)](#)*
- void [has_scrollbar](#) ([uchar](#) mode)
 - Sets whether the widget should have scrollbars or not (default [Fl_Browser_::BOTH](#)).*
- int [hposition](#) () const
 - Gets the horizontal scroll position of the list as a pixel position *pos*.*
- void [hposition](#) (int)
 - Sets the horizontal scroll position of the list to pixel position *pos*.*
- int [linespacing](#) () const
 - Return the height of additional spacing between browser lines.*
- void [linespacing](#) (int pixels)
 - Add some space between browser lines.*
- int [position](#) () const
- void [position](#) (int pos)
- void **position** (int *x*, int *y*)
- void [resize](#) (int *X*, int *Y*, int *W*, int *H*) [FL_OVERRIDE](#)
 - Repositions and/or resizes the browser.*
- void [scrollbar_left](#) ()
 - Moves the vertical scrollbar to the lefthand side of the list.*
- void [scrollbar_right](#) ()
 - Moves the vertical scrollbar to the righthand side of the list.*
- int [scrollbar_size](#) () const
 - Gets the current size of the scrollbars' troughs, in pixels.*
- void [scrollbar_size](#) (int newSize)
 - Sets the pixel size of the scrollbars' troughs to *newSize*, in pixels.*
- int [scrollbar_width](#) () const
 - Returns the global value [Fl::scrollbar_size\(\)](#).*
- void [scrollbar_width](#) (int width)
 - Sets the global [Fl::scrollbar_size\(\)](#), and forces this instance of the widget to use it.*
- int [select](#) (void *item, int val=1, int docallbacks=0)
 - Sets the selection state of *item* to *val*, and returns 1 if the state changed or 0 if it did not.*
- int [select_only](#) (void *item, int docallbacks=0)
 - Selects *item* and returns 1 if the state changed or 0 if it did not.*
- void [sort](#) (int flags=0)
 - Sort the items in the browser based on *flags*.*
- [Fl_Color](#) **textcolor** () const
 - Gets the default text color for the lines in the browser.*
- void **textcolor** ([Fl_Color](#) col)
 - Sets the default text color for the lines in the browser to color *col*.*
- [Fl_Font](#) **textfont** () const
 - Gets the default text font for the lines in the browser.*
- void **textfont** ([Fl_Font](#) font)
 - Sets the default text font for the lines in the browser to *font*.*
- [Fl_Fontsize](#) **textsize** () const
 - Gets the default text size (in pixels) for the lines in the browser.*
- void **textsize** ([Fl_Fontsize](#) newSize)
 - Sets the default text size (in pixels) for the lines in the browser to *size*.*
- int [vposition](#) () const
 - Gets the vertical scroll position of the list as a pixel position *pos*.*
- void [vposition](#) (int pos)
 - Sets the vertical scroll position of the list to pixel position *pos*.*

Public Member Functions inherited from FL_Group

- [FL_Widget](#) *[_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void **add** ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void **add_resizable** ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * **array** () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * **as_group** () const [FL_OVERRIDE](#)
- [FL_Group](#) * **as_group** () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
*Removes the widget at *index* from the group and deletes it.*
- void **end** ()
*Exactly the same as *current(this->parent())*.*
- int **find** (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int **find** (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char * = 0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void **focus** ([FL_Widget](#) *W)
- void **forms_end** ()
This is for forms compatibility only.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** ([FL_Widget](#) &o, [FL_Widget](#) *before)
*This does *insert(w, find(before))*.*
- void **remove** ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void **remove** ([FL_Widget](#) *o)

- Removes the widget `o` from the group.*

 - void `remove` (int index)

Removes the widget at `index` from the group but does not delete it.
- `FL_Widget * resizable` () const

Returns the group's resizable widget.
- void `resizable` (`FL_Widget &o`)

Sets the group's resizable widget.
- void `resizable` (`FL_Widget *o`)

The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- void `resize` (int, int, int, int) `FL_OVERRIDE`

Resizes the `FL_Group` widget and all of its children.
- virtual `~FL_Group` ()

The destructor also deletes all the children.

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
- Activates the widget.*

 - unsigned int `active` () const

Returns whether the widget is active.
- int `active_r` () const

Returns whether the widget and all of its parents are active.
- `FL_Align align` () const

Gets the label alignment.
- void `align` (`FL_Align alignment`)

Sets the label alignment.
- long `argument` () const

Gets the current user data (long) argument that is passed to the callback function.
- void `argument` (long v)

Sets the current user data (long) argument that is passed to the callback function.
- virtual class `FL_Gl_Window * as_gl_window` ()

Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.
- virtual class `FL_Gl_Window` const * `as_gl_window` () const
- virtual `FL_Window * as_window` ()

Returns an `FL_Window` pointer if this widget is an `FL_Window`.
- virtual `FL_Window` const * `as_window` () const
- void `bind_deimage` (`FL_Image *img`)

Sets the image to use as part of the widget label when in the inactive state.
- void `bind_deimage` (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void `bind_image` (`FL_Image *img`)

Sets the image to use as part of the widget label when in the active state.
- void `bind_image` (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- `FL_Boxtype box` () const

Gets the box type of the widget.
- void `box` (`FL_Boxtype new_box`)

Sets the box type for the widget.
- `FL_Callback_p callback` () const

- Gets the current callback function for the widget.*
- void `callback` (`FI_Callback` *cb)
- Sets the current callback function for the widget.*
- void `callback` (`FI_Callback` *cb, `FI_Callback_User_Data` *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*
- void `callback` (`FI_Callback` *cb, void *p)
- Sets the current callback function and data for the widget.*
- void `callback` (`FI_Callback0` *cb)
- Sets the current callback function for the widget.*
- void `callback` (`FI_Callback1` *cb, long p=0)
- Sets the current callback function for the widget.*
- unsigned int `changed` () const
- Checks if the widget value changed since the last callback.*
- void `clear_active` ()
- Marks the widget as inactive without sending events or changing focus.*
- void `clear_changed` ()
- Marks the value of the widget as unchanged.*
- void `clear_damage` (`uchar` c=0)
- Clears or sets the damage flags.*
- void `clear_output` ()
- Sets a widget to accept input.*
- void `clear_visible` ()
- Hides the widget.*
- void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*
- `FI_Color` `color` () const
- Gets the background color of the widget.*
- void `color` (`FI_Color` bg)
- Sets the background color of the widget.*
- void `color` (`FI_Color` bg, `FI_Color` sel)
- Sets the background and selection color of the widget.*
- `FI_Color` `color2` () const
- For back compatibility only.*
- void `color2` (unsigned a)
- For back compatibility only.*
- int `contains` (const `FI_Widget` *w) const
- Checks if w is a child of this widget.*
- void `copy_label` (const char *new_label)
- Sets the current label.*
- void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*
- `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*
- void `damage` (`uchar` c)
- Sets the damage bits for the widget.*
- void `damage` (`uchar` c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int `damage_resize` (int, int, int, int)
- Internal use only.*
- void `deactivate` ()
- Deactivates the widget.*

- `FL_Image * deimage ()`
Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`
Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`
Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`
Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`
Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FL_Align) const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`
Gets the widget height.
- `virtual void hide ()`
Makes a widget invisible.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FL_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FL_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FL_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FL_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FL_Color labelcolor () const`

- Gets the label color.*

 - void [labelcolor](#) ([Fl_Color](#) c)
- Sets the label color.*

 - [Fl_Font](#) [labelfont](#) () const
- Gets the font to use.*

 - void [labelfont](#) ([Fl_Font](#) f)
- Sets the font to use.*

 - [Fl_Fontsize](#) [labelsize](#) () const
- Gets the font size in pixels.*

 - void [labelsize](#) ([Fl_Fontsize](#) pix)
- Sets the font size in pixels.*

 - [Fl_Labeltype](#) [labeltype](#) () const
- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const
- Sets width ww and height hh accordingly with the label size.*

 - bool [needs_keyboard](#) () const
- Returns whether this widget needs a keyboard.*

 - void [needs_keyboard](#) (bool needs)
- Sets whether this widget needs a keyboard.*

 - unsigned int [output](#) () const
- Returns if a widget is used for output only.*

 - [Fl_Group](#) * [parent](#) () const
- Returns a pointer to the parent widget.*

 - void [parent](#) ([Fl_Group](#) *p)
- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
- Repositions the window or widget.*

 - void [redraw](#) ()
- Schedules the drawing of the widget.*

 - void [redraw_label](#) ()
- Schedules the drawing of the label.*

 - [Fl_Color](#) [selection_color](#) () const
- Gets the selection color.*

 - void [selection_color](#) ([Fl_Color](#) a)
- Sets the selection color.*

 - void [set_active](#) ()
- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

 - void [set_output](#) ()
- Sets a widget to output only.*

 - void [set_visible](#) ()
- Makes the widget visible.*

 - void [set_visible_focus](#) ()
- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void [shortcut_label](#) (int value)
- Sets whether the widget's label uses '&' to indicate shortcuts.*

- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const

- Gets the widget position in its window.*
 • `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Public Attributes

- `Fl_Scrollbar hscrollbar`
Horizontal scrollbar.
- `Fl_Scrollbar scrollbar`
Vertical scrollbar.

Protected Member Functions

- `void bbox (int &X, int &Y, int &W, int &H) const`
Returns the bounding box for the interior of the list's display window, inside the scrollbars.
- `void deleting (void *item)`
This method should be used when `item` is being deleted from the list.
- `int displayed (void *item) const`
Returns non-zero if `item` has been scrolled to a position where it is being displayed.
- `void draw () FL_OVERRIDE`
Draws the list within the normal widget bounding box.
- `void * find_item (int ypos)`
This method returns the item under mouse y position `ypos`.
- `Fl_Browser_ (int X, int Y, int W, int H, const char *L=0)`
The constructor makes an empty browser.
- `virtual int full_height () const`
This method may be provided by the subclass to indicate the full height of the item list, in pixels.
- `virtual int full_width () const`
This method may be provided by the subclass to indicate the full width of the item list, in pixels.
- `virtual int incr_height () const`
This method may be provided to return the average height of all items to be used for scrolling.
- `void inserting (void *a, void *b)`
This method should be used when an item is in the process of being inserted into the list.
- `virtual void * item_at (int index) const`
This method must be provided by the subclass to return the item for the specified `index`.
- `virtual void item_draw (void *item, int X, int Y, int W, int H) const =0`
This method must be provided by the subclass to draw the `item` in the area indicated by `X, Y, W, H`.
- `virtual void * item_first () const =0`
This method must be provided by the subclass to return the first item in the list.
- `virtual int item_height (void *item) const =0`
This method must be provided by the subclass to return the height of `item` in pixels.
- `virtual void * item_last () const`
This method must be provided by the subclass to return the last item in the list.
- `virtual void * item_next (void *item) const =0`
This method must be provided by the subclass to return the item in the list after `item`.
- `virtual void * item_prev (void *item) const =0`
This method must be provided by the subclass to return the item in the list before `item`.
- `virtual int item_quick_height (void *item) const`
This method may be provided by the subclass to return the height of the `item`, in pixels.
- `virtual void item_select (void *item, int val=1)`

This method must be implemented by the subclass if it supports multiple selections; sets the selection state to `val` for the `item`.

- virtual int `item_selected` (void *item) const

This method must be implemented by the subclass if it supports multiple selections; returns the selection state for `item`.

- virtual void `item_swap` (void *a, void *b)

This optional method should be provided by the subclass to efficiently swap browser items `a` and `b`, such as for sorting.

- virtual const char * `item_text` (void *item) const

This optional method returns a string (label) that may be used for sorting.

- virtual int `item_width` (void *item) const =0

This method must be provided by the subclass to return the width of the `item` in pixels.

- int `leftedge` () const

This method returns the X position of the left edge of the list area after adjusting for the scrollbar and border, if any.

- void `new_list` ()

This method should be called when the list data is completely replaced or cleared.

- void `redraw_line` (void *item)

This method should be called when the contents of `item` has changed, but not its height.

- void `redraw_lines` ()

This method will cause the entire list to be redrawn.

- void `replacing` (void *a, void *b)

This method should be used when item `a` is being replaced by item `b`.

- void * `selection` () const

Returns the item currently selected, or NULL if there is no selection.

- void `swapping` (void *a, void *b)

This method should be used when two items `a` and `b` are being swapped.

- void * `top` () const

Returns the item that appears at the top of the list.

Protected Member Functions inherited from `FI_Group`

- `FI_Rect` * `bounds` ()

Returns the internal array of widget sizes and positions.

- void `draw` () `FL_OVERRIDE`

Draws the widget.

- void `draw_child` (`FI_Widget` &widget) const

Forces a child to redraw.

- void `draw_children` ()

Draws all children of the group.

- void `draw_outside_label` (const `FI_Widget` &widget) const

Parents normally call this to draw outside labels of child widgets.

- virtual int `on_insert` (`FI_Widget` *, int)

Allow derived groups to act when a widget is added as a child.

- virtual int `on_move` (int, int)

Allow derived groups to act when a widget is moved within the group.

- virtual void `on_remove` (int)

Allow derived groups to act when a child widget is removed from the group.

- int * `sizes` ()

Returns the internal array of widget sizes and positions.

- void `update_child` (`FI_Widget` &widget) const

Draws a child only if it needs it.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Group

- static FI_Group * **current** ()
Returns the currently active group.
- static void **current** (FI_Group *g)
Sets the current group.

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

```
• enum {
    INACTIVE = 1<<0 , INVISIBLE = 1<<1 , OUTPUT = 1<<2 , NOBORDER = 1<<3 ,
    FORCE_POSITION = 1<<4 , NON_MODAL = 1<<5 , SHORTCUT_LABEL = 1<<6 , CHANGED = 1<<7
,
    OVERRIDE = 1<<8 , VISIBLE_FOCUS = 1<<9 , COPIED_LABEL = 1<<10 , CLIP_CHILDREN = 1<<11
,
    MENU_WINDOW = 1<<12 , TOOLTIP_WINDOW = 1<<13 , MODAL = 1<<14 , NO_OVERLAY = 1<<15
,
    GROUP_RELATIVE = 1<<16 , COPIED_TOOLTIP = 1<<17 , FULLSCREEN = 1<<18 , MAC_USE_ACCENTS_MENU
= 1<<19 ,
    NEEDS_KEYBOARD = 1<<20 , IMAGE_BOUND = 1<<21 , DEIMAGE_BOUND = 1<<22 ,
    AUTO_DELETE_USER_DATA = 1<<23 ,
    MAXIMIZED = 1<<24 , POPUP = 1<<25 , USERFLAG3 = 1<<29 , USERFLAG2 = 1<<30 ,
    USERFLAG1 = 1<<31 }
```

flags possible values enumeration.

11.14.1 Detailed Description

This is the base class for browsers.

To be useful it must be subclassed and several virtual functions defined. The Forms-compatible browser and the file chooser's browser are subclassed off of this.

This has been designed so that the subclass has complete control over the storage of the data, although because `next()` and `prev()` functions are used to index, it works best as a linked list or as a large block of characters in which the line breaks must be searched for.

A great deal of work has been done so that the "height" of a data object does not need to be determined until it is drawn. This is useful if actually figuring out the size of an object requires accessing image data or doing `stat()` on a file or doing some other slow operation.

Callbacks are called when the value changes with `FL_REASON_CHANGED`. If `FL_WHEN_RELEASE` is set, callbacks are called when the mouse button is released with `FL_REASON_CHANGED` or `FL_REASON_RESELECTED` if the selection did not change. If `FL_WHEN_ENTER_KEY` is set, callbacks are also called when key presses or double clicks change the selection.

Keyboard navigation of browser items

The keyboard navigation of browser items is only possible if `visible_focus()` is enabled. If disabled, the widget rejects keyboard focus; Tab and Shift-Tab focus navigation will skip the widget.

In 'Select' and 'Normal' mode, the widget rejects keyboard focus; no navigation keys are supported (other than scrollbar positioning).

In 'Hold' mode, the widget accepts keyboard focus, and Up/Down arrow keys can navigate the selected item.

In 'Multi' mode, the widget accepts keyboard focus, and Up/Down arrow keys navigate the focus box; Space toggles the current item's selection, Enter selects only the current item (deselects all others). If Shift (or Ctrl) is combined with Up/Down arrow keys, the current item's selection state is extended to the next item. In this way one can extend a selection or de-selection.

11.14.2 Member Enumeration Documentation

11.14.2.1 anonymous enum

anonymous enum

Values for `has_scrollbar()`.

Anonymous enum bit flags for `has_scrollbar()`.

- bit 0: horizontal
- bit 1: vertical
- bit 2: 'always' (to be combined with bits 0 and 1)
- bit 3-31: reserved for future use

Enumerator

HORIZONTAL	Only show horizontal scrollbar.
VERTICAL	Only show vertical scrollbar.
BOTH	Show both scrollbars. (default)
ALWAYS_ON	Specified scrollbar(s) should 'always' be shown (to be used with HORIZONTAL/VERTICAL)
HORIZONTAL_ALWAYS	Horizontal scrollbar always on.
VERTICAL_ALWAYS	Vertical scrollbar always on.
BOTH_ALWAYS	Both scrollbars always on.

11.14.3 Constructor & Destructor Documentation

11.14.3.1 Fl_Browser_()

```
Fl_Browser_::Fl_Browser_ (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0) [protected]
```

The constructor makes an empty browser.

Parameters

in	<i>X,Y,W,H</i>	position and size.
in	<i>L</i>	The label string, may be NULL.

11.14.4 Member Function Documentation

11.14.4.1 bbox()

```
void Fl_Browser_::bbox (
    int & X,
    int & Y,
    int & W,
    int & H) const [protected]
```

Returns the bounding box for the interior of the list's display window, inside the scrollbars.

Parameters

out	<i>X,Y,W,H</i>	The returned bounding box. (The original contents of these parameters are overwritten)

11.14.4.2 deleting()

```
void Fl_Browser_::deleting (
    void * item) [protected]
```

This method should be used when *item* is being deleted from the list.

It allows the [Fl_Browser_](#) to discard any cached data it has on the item. This method does not actually delete the item, but handles the follow up bookkeeping after the item has just been deleted.

Parameters

in	<i>item</i>	The item being deleted.

11.14.4.3 deselect()

```
int Fl_Browser_::deselect (
    int docallbacks = 0)
```

Deselects all items in the list and returns 1 if the state changed or 0 if it did not.

If the optional `docallbacks` parameter is non-zero, `deselect` tries to call the callback function for the widget.

Parameters

in	<i>docallbacks</i>	If non-zero, invokes widget callback if item changed. If 0, doesn't do callback (default).

11.14.4.4 display()

```
void Fl_Browser_::display (
    void * item)
```

Displays the `item`, scrolling the list as necessary.

Parameters

in	<i>item</i>	The item to be displayed.

See also

[display\(\)](#), [displayed\(\)](#)

11.14.4.5 displayed()

```
int Fl_Browser_::displayed (
    void * item) const [protected]
```

Returns non-zero if `item` has been scrolled to a position where it is being displayed.

Checks to see if the item's vertical position is within the top and bottom edges of the display window. This does NOT take into account the [hide\(\)/show\(\)](#) status of the widget or item.

Parameters

in	<i>item</i>	The item to check

Returns

1 if visible, 0 if not visible.

See also

[display\(\)](#), [displayed\(\)](#)

11.14.4.6 draw()

```
void Fl_Browser_::draw (
    void ) [protected], [virtual]
```

Draws the list within the normal widget bounding box.

Implements [Fl_Widget](#).

11.14.4.7 find_item()

```
void * Fl_Browser_::find_item (
    int ypos) [protected]
```

This method returns the item under mouse y position `ypos`.

NULL is returned if no item is displayed at that position.

Parameters

in	ypos	The y position (eg. Fl::event_y()) to find an item under.

Returns

The item, or NULL if not found

11.14.4.8 full_height()

```
int Fl_Browser_::full_height () const [protected], [virtual]
```

This method may be provided by the subclass to indicate the full height of the item list, in pixels.

The default implementation computes the full height from the item heights. Includes the items that are scrolled off screen.

Returns

The height of the entire list, in pixels.

Reimplemented in [Fl_Browser](#).

11.14.4.9 full_width()

```
int Fl_Browser_::full_width () const [protected], [virtual]
```

This method may be provided by the subclass to indicate the full width of the item list, in pixels.

The default implementation computes the full width from the item widths.

Returns

The maximum width of all the items, in pixels.

11.14.4.10 handle()

```
int Fl_Browser_::handle (
    int event) [virtual]
```

Handles the `event` within the normal widget bounding box.

Parameters

in	event	The event to process.

Returns

1 if event was processed, 0 if not.

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Check_Browser](#).

11.14.4.11 has_scrollbar()

```
void Fl_Browser_::has_scrollbar (
    uchar mode) [inline]
```

Sets whether the widget should have scrollbars or not (default [Fl_Browser_::BOTH](#)).

By default you can scroll in both directions, and the scrollbars disappear if the data will fit in the widget.

[has_scrollbar\(\)](#) changes this based on the value of `mode`:

- 0 - No scrollbars.
- [Fl_Browser_::HORIZONTAL](#) - Only a horizontal scrollbar.

- [Fl_Browser_::VERTICAL](#) - Only a vertical scrollbar.
- [Fl_Browser_::BOTH](#) - The default is both scrollbars.
- [Fl_Browser_::HORIZONTAL_ALWAYS](#) - Horizontal scrollbar always on, vertical always off.
- [Fl_Browser_::VERTICAL_ALWAYS](#) - Vertical scrollbar always on, horizontal always off.
- [Fl_Browser_::BOTH_ALWAYS](#) - Both always on.

11.14.4.12 `hposition()` [1/2]

```
int Fl_Browser_::hposition () const [inline]
```

Gets the horizontal scroll position of the list as a pixel position `pos`.

The position returned is how many pixels of the list are scrolled off the left edge of the screen. Example: A position of '18' indicates the left 18 pixels of the list are scrolled off the left edge of the screen.

See also

[position\(\)](#), [hposition\(\)](#)

11.14.4.13 `hposition()` [2/2]

```
void Fl_Browser_::hposition (
    int pos)
```

Sets the horizontal scroll position of the list to pixel position `pos`.

The position is how many pixels of the list are scrolled off the left edge of the screen. Example: A position of '18' scrolls the left 18 pixels of the list off the left edge of the screen.

Parameters

<code>in</code>	<code>pos</code>	The horizontal position (in pixels) to scroll the browser to.

See also

[vposition\(\)](#), [hposition\(\)](#)

11.14.4.14 `incr_height()`

```
int Fl_Browser_::incr_height () const [protected], [virtual]
```

This method may be provided to return the average height of all items to be used for scrolling.

The default implementation uses the height of the first item.

Returns

The average height of items, in pixels.

Reimplemented in [Fl_Browser](#).

11.14.4.15 `inserting()`

```
void Fl_Browser_::inserting (
    void * a,
    void * b) [protected]
```

This method should be used when an item is in the process of being inserted into the list.

It allows the [Fl_Browser_](#) to update its cache data as needed, scheduling a redraw for the affected lines. This method does not actually insert items, but handles the follow up bookkeeping after items have been inserted.

Parameters

in	<i>a</i>	The starting item position
in	<i>b</i>	The new item being inserted

11.14.4.16 item_at()

```
virtual void * Fl_Browser_::item_at (
    int index) const [inline], [protected], [virtual]
```

This method must be provided by the subclass to return the item for the specified *index*.

Parameters

in	<i>index</i>	The index of the item to be returned

Returns

The item at the specified *index*.

Reimplemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.17 item_draw()

```
virtual void Fl_Browser_::item_draw (
    void * item,
    int X,
    int Y,
    int W,
    int H) const [protected], [pure virtual]
```

This method must be provided by the subclass to draw the *item* in the area indicated by X, Y, W, H.

Implemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.18 item_first()

```
virtual void * Fl_Browser_::item_first () const [protected], [pure virtual]
```

This method must be provided by the subclass to return the first item in the list.

See also

[item_first\(\)](#), [item_next\(\)](#), [item_last\(\)](#), [item_prev\(\)](#)

Implemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.19 item_height()

```
virtual int Fl_Browser_::item_height (
    void * item) const [protected], [pure virtual]
```

This method must be provided by the subclass to return the height of *item* in pixels.

Allow for two additional pixels for the list selection box.

Parameters

in	<i>item</i>	The item whose height is returned.

Returns

The height of the specified *item* in pixels.

See also

[item_height\(\)](#), [item_width\(\)](#), [item_quick_height\(\)](#)

Implemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.20 item_last()

```
virtual void * Fl_Browser_::item_last () const [inline], [protected], [virtual]
```

This method must be provided by the subclass to return the last item in the list.

See also

[item_first\(\)](#), [item_next\(\)](#), [item_last\(\)](#), [item_prev\(\)](#)

Reimplemented in [Fl_Browser](#).

11.14.4.21 item_next()

```
virtual void * Fl_Browser_::item_next (
    void * item) const [protected], [pure virtual]
```

This method must be provided by the subclass to return the item in the list after *item*.

See also

[item_first\(\)](#), [item_next\(\)](#), [item_last\(\)](#), [item_prev\(\)](#)

Implemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.22 item_prev()

```
virtual void * Fl_Browser_::item_prev (
    void * item) const [protected], [pure virtual]
```

This method must be provided by the subclass to return the item in the list before *item*.

See also

[item_first\(\)](#), [item_next\(\)](#), [item_last\(\)](#), [item_prev\(\)](#)

Implemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.23 item_quick_height()

```
int Fl_Browser_::item_quick_height (
    void * item) const [protected], [virtual]
```

This method may be provided by the subclass to return the height of the *item*, in pixels.

Allow for two additional pixels for the list selection box. This method differs from *item_height* in that it is only called for selection and scrolling operations. The default implementation calls *item_height*.

Parameters

<i>in</i>	<i>item</i>	The item whose height to return.

Returns

The height, in pixels.

11.14.4.24 item_select()

```
void Fl_Browser_::item_select (
    void * item,
    int val = 1) [protected], [virtual]
```

This method must be implemented by the subclass if it supports multiple selections; sets the selection state to *val* for the *item*.

Sets the selection state for *item*, where optional *val* is 1 (select, the default) or 0 (de-select).

Parameters

in	<i>item</i>	The item to be selected
in	<i>val</i>	The optional selection state; 1=select, 0=de-select. The default is to select the item (1).

Reimplemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.25 item_selected()

```
int Fl_Browser_::item_selected (
    void * item) const [protected], [virtual]
```

This method must be implemented by the subclass if it supports multiple selections; returns the selection state for *item*.

The method should return 1 if *item* is selected, or 0 otherwise.

Parameters

in	<i>item</i>	The item to test.

Reimplemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.26 item_swap()

```
virtual void Fl_Browser_::item_swap (
    void * a,
    void * b) [inline], [protected], [virtual]
```

This optional method should be provided by the subclass to efficiently swap browser items *a* and *b*, such as for sorting.

Parameters

in	<i>a,b</i>	The two items to be swapped.

Reimplemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.27 item_text()

```
virtual const char * Fl_Browser_::item_text (
    void * item) const [inline], [protected], [virtual]
```

This optional method returns a string (label) that may be used for sorting.

Parameters

in	<i>item</i>	The item whose label text is returned.

Returns

The item's text label. (Can be NULL if blank)

Reimplemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.28 item_width()

```
virtual int Fl_Browser_::item_width (
    void * item) const [protected], [pure virtual]
```

This method must be provided by the subclass to return the width of the *item* in pixels.

Allow for two additional pixels for the list selection box.

Parameters

<i>in</i>	<i>item</i>	The item whose width is returned.

Returns

The width of the item in pixels.

Implemented in [Fl_Browser](#), and [Fl_Check_Browser](#).

11.14.4.29 leftedge()

```
int Fl_Browser_::leftedge () const [protected]
```

This method returns the X position of the left edge of the list area after adjusting for the scrollbar and border, if any.

Returns

The X position of the left edge of the list, in pixels.

See also

[Fl_Browser_::bbox\(\)](#)

11.14.4.30 linespacing() [1/2]

```
int Fl_Browser_::linespacing () const [inline]
```

Return the height of additional spacing between browser lines.

Returns

spacing height in pixel units.

11.14.4.31 linespacing() [2/2]

```
void Fl_Browser_::linespacing (
    int pixels) [inline]
```

Add some space between browser lines.

Parameters

<i>in</i>	<i>pixels</i>	number of additional pixels between lines.

11.14.4.32 new_list()

```
void Fl_Browser_::new_list () [protected]
```

This method should be called when the list data is completely replaced or cleared.

It informs the [Fl_Browser_](#) widget that any cached information it has concerning the items is invalid. This method does not clear the list, it just handles the follow up bookkeeping after the list has been cleared.

11.14.4.33 position() [1/2]

```
int Fl_Browser_::position () const [inline]
```

Deprecated "since 1.4.0 - use vposition() instead"

11.14.4.34 position() [2/2]

```
void Fl_Browser_::position (
    int pos) [inline]
```

Deprecated "since 1.4.0 - use `vposition(pos)` instead"

11.14.4.35 redraw_line()

```
void Fl_Browser_::redraw_line (
    void * item) [protected]
```

This method should be called when the contents of `item` has changed, but not its height.

Parameters

in	<i>item</i>	The item that needs to be redrawn.

See also

[redraw_lines\(\)](#), [redraw_line\(\)](#)

11.14.4.36 redraw_lines()

```
void Fl_Browser_::redraw_lines () [inline], [protected]
```

This method will cause the entire list to be redrawn.

See also

[redraw_lines\(\)](#), [redraw_line\(\)](#)

11.14.4.37 replacing()

```
void Fl_Browser_::replacing (
    void * a,
    void * b) [protected]
```

This method should be used when item `a` is being replaced by item `b`.

It allows the `Fl_Browser_` to update its cache data as needed, schedules a redraw for the item being changed, and tries to maintain the selection. This method does not actually replace the item, but handles the follow up bookkeeping after the item has just been replaced.

Parameters

in	<i>a</i>	Item being replaced
in	<i>b</i>	Item to replace 'a'

11.14.4.38 resize()

```
void Fl_Browser_::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Repositions and/or resizes the browser.

Parameters

in	<i>X,Y,W,H</i>	The new position and size for the browser, in pixels.

Reimplemented from [Fl_Widget](#).

11.14.4.39 scrollbar_left()

```
void Fl_Browser_::scrollbar_left () [inline]
```

Moves the vertical scrollbar to the lefthand side of the list.

For back compatibility.

11.14.4.40 scrollbar_right()

```
void Fl_Browser_::scrollbar_right () [inline]
```

Moves the vertical scrollbar to the righthand side of the list.

For back compatibility.

11.14.4.41 scrollbar_size() [1/2]

```
int Fl_Browser_::scrollbar_size () const [inline]
```

Gets the current size of the scrollbars' troughs, in pixels.

If this value is zero (default), this widget will use the [Fl::scrollbar_size\(\)](#) value as the scrollbar's width.

Returns

Scrollbar size in pixels, or 0 if the global [Fl::scrollbar_size\(\)](#) is being used.

See also

[Fl::scrollbar_size\(int\)](#)

11.14.4.42 scrollbar_size() [2/2]

```
void Fl_Browser_::scrollbar_size (
    int newSize) [inline]
```

Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.

Normally you should not need this method, and should use [Fl::scrollbar_size\(int\)](#) instead to manage the size of ALL your widgets' scrollbars. This ensures your application has a consistent UI, is the default behavior, and is normally what you want.

Only use THIS method if you really need to override the global scrollbar size. The need for this should be rare.

Setting `newSize` to the special value of 0 causes the widget to track the global [Fl::scrollbar_size\(\)](#), which is the default.

Parameters

<code>in</code>	<code>newSize</code>	Sets the scrollbar size in pixels. If 0 (default), scrollbar size tracks the global Fl::scrollbar_size()

See also

[Fl::scrollbar_size\(\)](#)

11.14.4.43 scrollbar_width() [1/2]

```
int Fl_Browser_::scrollbar_width () const [inline]
```

Returns the global value [Fl::scrollbar_size\(\)](#).

Deprecated Use [scrollbar_size\(\)](#) instead.

Todo This method should eventually be removed in 1.4+

11.14.4.44 scrollbar_width() [2/2]

```
void Fl_Browser_::scrollbar_width (
    int width) [inline]
```

Sets the global [Fl::scrollbar_size\(\)](#), and forces this instance of the widget to use it.

Deprecated Use [scrollbar_size\(\)](#) instead.

Todo This method should eventually be removed in 1.4+

11.14.4.45 select()

```
int Fl_Browser_::select (
    void * item,
    int val = 1,
    int docallbacks = 0)
```

Sets the selection state of *item* to *val*, and returns 1 if the state changed or 0 if it did not.

If *docallbacks* is non-zero, *select* tries to call the callback function for the widget.

Parameters

in	<i>item</i>	The item whose selection state is to be changed
in	<i>val</i>	The new selection state (1=select, 0=de-select)
in	<i>docallbacks</i>	If non-zero, invokes widget callback if item changed. If 0, doesn't do callback (default).

Returns

1 if state was changed, 0 if not.

11.14.4.46 select_only()

```
int Fl_Browser_::select_only (
    void * item,
    int docallbacks = 0)
```

Selects *item* and returns 1 if the state changed or 0 if it did not.

Any other items in the list are deselected.

Parameters

in	<i>item</i>	The <i>item</i> to select.
in	<i>docallbacks</i>	If non-zero, invokes widget callback if item changed. If 0, doesn't do callback (default).

11.14.4.47 selection()

```
void * Fl_Browser_::selection () const [inline], [protected]
```

Returns the item currently selected, or NULL if there is no selection.

For multiple selection browsers this call returns the currently focused item, even if it is not selected. To find all selected items, call [Fl_Multi_Browser::selected\(\)](#) for every item in question.

11.14.4.48 sort()

```
void Fl_Browser_::sort (
    int flags = 0)
```

Sort the items in the browser based on *flags*.

[item_swap\(void*, void*\)](#) and [item_text\(void*\)](#) must be implemented for this call.

Parameters

in	<i>flags</i>	FL_SORT_ASCENDING – sort in ascending order FL_SORT_DESCENDING – sort in descending order FL_SORT_CASEINSENSITIVE – add this to sort case-insensitively Values other than the above will cause undefined behavior Other flags may appear in the future.

11.14.4.49 swapping()

```
void Fl_Browser_::swapping (
    void * a,
    void * b) [protected]
```

This method should be used when two items *a* and *b* are being swapped.

It allows the [Fl_Browser_](#) to update its cache data as needed, schedules a redraw for the two items, and tries to maintain the current selection. This method does not actually swap items, but handles the follow up bookkeeping after items have been swapped.

Parameters

in	<i>a,b</i>	Items being swapped.

11.14.4.50 textfont()

```
Fl_Font Fl_Browser_::textfont () const [inline]
```

Gets the default text font for the lines in the browser.

See also

[textfont\(\)](#), [textsize\(\)](#), [textcolor\(\)](#)

11.14.4.51 vposition() [1/2]

```
int Fl_Browser_::vposition () const [inline]
```

Gets the vertical scroll position of the list as a pixel position *pos*.

The position returned is how many pixels of the list are scrolled off the top edge of the screen. Example: A position of '3' indicates the top 3 pixels of the list are scrolled off the top edge of the screen.

See also

[position\(\)](#), [hposition\(\)](#)

11.14.4.52 vposition() [2/2]

```
void Fl_Browser_::vposition (
    int pos)
```

Sets the vertical scroll position of the list to pixel position *pos*.

The position is how many pixels of the list are scrolled off the top edge of the screen. Example: A position of '3' scrolls the top three pixels of the list off the top edge of the screen.

Parameters

in	<i>pos</i>	The vertical position (in pixels) to scroll the browser to.

See also

[vposition\(\)](#), [hposition\(\)](#)

11.14.5 Member Data Documentation

11.14.5.1 hscrollbar

`Fl_Scrollbar Fl_Browser_::hscrollbar`

Horizontal scrollbar.

Public, so that it can be accessed directly.

11.14.5.2 scrollbar

`Fl_Scrollbar Fl_Browser_::scrollbar`

Vertical scrollbar.

Public, so that it can be accessed directly.

Use `scrollbar_left()` or `scrollbar_right()` to change what side the vertical scrollbar is drawn on.

Use `scrollbar.align(int)` (see `Fl_Widget::align(Fl_Align)`) to change what side either of the scrollbars is drawn on.

If the `FL_ALIGN_LEFT` bit is on, the vertical scrollbar is on the left. If the `FL_ALIGN_TOP` bit is on, the horizontal scrollbar is on the top. Note that only the alignment flags in scrollbar are considered. The flags in hscrollbar however are ignored.

The documentation for this class was generated from the following files:

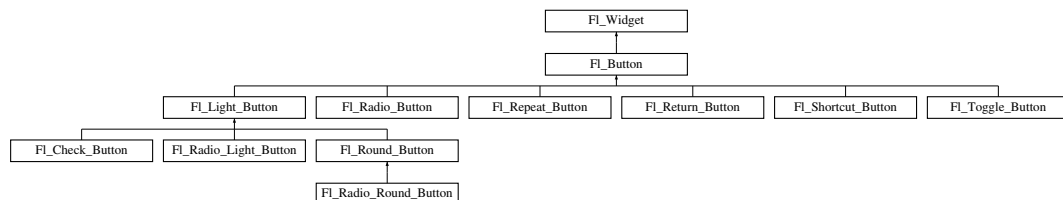
- `Fl_Browser_.H`
- `Fl_Browser_.cxx`

11.15 Fl_Button Class Reference

Buttons generate callbacks when they are clicked by the user.

```
#include <Fl_Button.H>
```

Inheritance diagram for `Fl_Button`:



Public Member Functions

- `int clear()`
Same as `value(0)`.
- `uchar compact()`
Return true if buttons are rendered as compact buttons.
- `void compact(uchar v)`
Decide if buttons should be rendered in compact mode.
- `Fl_Boxtype down_box() const`
Returns the current down box type, which is drawn when `value()` is non-zero.
- `void down_box(Fl_Boxtype b)`
Sets the down box type.
- `Fl_Color down_color() const`
(for backwards compatibility)
- `void down_color(unsigned c)`
(for backwards compatibility)
- `Fl_Button(int X, int Y, int W, int H, const char *L=0)`
The constructor creates the button using the given position, size, and label.

- int **handle** (int) **FL_OVERRIDE**
Handles the specified event.
- int **set** ()
Same as `value (1)`.
- void **setonly** ()
Turns on this button and turns off all other radio buttons in the group (calling `value (1)` or `set ()` does not do this).
- int **shortcut** () const
Returns the current shortcut key for the button.
- void **shortcut** (const char *s)
(for backwards compatibility)
- void **shortcut** (int s)
Sets the shortcut key to `s`.
- char **value** () const
Returns the current value of the button (0 or 1).
- int **value** (int v)
Sets the current value of the button.

Public Member Functions inherited from **FL_Widget**

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FL_Align** **align** () const
Gets the label alignment.
- void **align** (**FL_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FL_Gl_Window** * **as_gl_window** ()
*Returns an **FL_Gl_Window** pointer if this widget is an **FL_Gl_Window**.*
- virtual class **FL_Gl_Window** const * **as_gl_window** () const
- virtual **FL_Group** * **as_group** ()
*Returns an **FL_Group** pointer if this widget is an **FL_Group**.*
- virtual **FL_Group** const * **as_group** () const
- virtual **FL_Window** * **as_window** ()
*Returns an **FL_Window** pointer if this widget is an **FL_Window**.*
- virtual **FL_Window** const * **as_window** () const
- void **bind_deimage** (**FL_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (**FL_Image** *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)

- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- [FI_Boxtype box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) ([uchar](#) c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.

- void **damage** (uchar c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int **damage_resize** (int, int, int, int)
Internal use only.
- void **deactivate** ()
Deactivates the widget.
- **FL_Image** * **deimage** ()
Gets the image that is used as part of the widget label when in the inactive state.
- const **FL_Image** * **deimage** () const
Gets the image that is used as part of the widget label when in the inactive state.
- void **deimage** (**FL_Image** &img)
Sets the image to use as part of the widget label when in the inactive state.
- void **deimage** (**FL_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- int **deimage_bound** () const
Returns whether the inactive image is managed by the widget.
- void **do_callback** (**FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)
Calls the widget callback function with default arguments.
- void **do_callback** (**FL_Widget** *widget, long arg, **FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)
Calls the widget callback function with arbitrary arguments.
- void **do_callback** (**FL_Widget** *widget, void *arg=0, **FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)
Calls the widget callback function with arbitrary arguments.
- void **draw_label** (int, int, int, int, **FL_Align**) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int **h** () const
Gets the widget height.
- virtual void **hide** ()
Makes a widget invisible.
- int **horizontal_label_margin** ()
Get the spacing between the label and the horizontal edge of the widget.
- void **horizontal_label_margin** (int px)
Set the spacing between the label and the horizontal edge of the widget.
- **FL_Image** * **image** ()
Gets the image that is used as part of the widget label when in the active state.
- const **FL_Image** * **image** () const
Gets the image that is used as part of the widget label when in the active state.
- void **image** (**FL_Image** &img)
Sets the image to use as part of the widget label when in the active state.
- void **image** (**FL_Image** *img)
Sets the image to use as part of the widget label when in the active state.
- int **image_bound** () const
Returns whether the image is managed by the widget.
- int **inside** (const **FL_Widget** *wgt) const
Checks if this widget is a child of wgt.
- int **is_label_copied** () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * **label** () const
Gets the current label text.
- void **label** (const char *text)
Sets the current label pointer.
- void **label** (**FL_Labeltype** a, const char *b)

- Shortcut to set the label text and type in one call.*

 - int [label_image_spacing](#) ()

Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)

Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const

Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)

Sets the label color.
- [FL_Font](#) [labelfont](#) () const

Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)

Sets the font to use.
- [FL_Fontsize](#) [labelsize](#) () const

Gets the font size in pixels.
- void [labelsize](#) ([FL_Fontsize](#) pix)

Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const

Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)

Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.

- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const

- Gets the widget width.*
- `FL_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `FL_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~FL_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.
- `void simulate_key_action ()`

Protected Member Functions inherited from FL_Widget

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const`
Draws a focus box for the widget at the given position and size.
- `void draw_label () const`
Draws the widget's label at the defined label position.
- `void draw_label (int, int, int, int) const`
Draws the label in an arbitrary bounding box.
- `FL_Widget (int x, int y, int w, int h, const char *label=0L)`
Creates a widget at the given position and size.
- `unsigned int flags () const`
Gets the widget flags mask.
- `void h (int v)`
Internal use only.
- `void set_flag (unsigned int c)`
Sets a flag in the flags mask.
- `void w (int v)`
Internal use only.

- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Static Protected Member Functions

- static void `key_release_timeout` (void *)

Static Protected Attributes

- static `Fl_Widget_Tracker` * `key_release_tracker` = 0

Additional Inherited Members

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget` *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `Fl_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

11.15.1 Detailed Description

Buttons generate callbacks when they are clicked by the user.

You control exactly when and how by changing the values for `type(uchar)` and `when(uchar)`. Buttons can also generate callbacks in response to `FL_SHORTCUT` events. The button can either have an explicit `shortcut(int s)` value or a letter shortcut can be indicated in the `label()` with an '&' character before it. For the label shortcut it does not matter if *Alt* is held down, but if you have an input field in the same window, the user will have to hold down the *Alt* key so that the input field does not eat the event first as an `FL_KEYBOARD` event.

See also

`Fl_Widget::shortcut_label(int)`

For an `Fl_Button` object, the `type()` call returns one of:

- `FL_NORMAL_BUTTON` (0): `value()` remains unchanged after button press.

- `FL_TOGGLE_BUTTON`: `value()` is inverted after button press.
- `FL_RADIO_BUTTON`: `value()` is set to 1 after button press, and all other buttons in the current group with `type() == FL_RADIO_BUTTON` are set to zero.

For an `Fl_Button` object, the following `when()` values are useful, the default being `FL_WHEN_RELEASE`:

- 0: The callback is not done, instead `changed()` is turned on.
- `FL_WHEN_RELEASE`: The callback is done after the user successfully clicks the button, or when a shortcut is typed. The reason is `FL_REASON_RELEASED`.
- `FL_WHEN_CHANGED`: The callback is done each time the `value()` changes (when the user pushes and releases the button, and as the mouse is dragged around in and out of the button). The reason is set to `FL_REASON_CHANGED`.
- `FL_WHEN_NOT_CHANGED`: The callback is done when the mouse button is released, but the value did not changed. The reason is set to `FL_REASON_SELECTED`.

11.15.2 Constructor & Destructor Documentation

11.15.2.1 Fl_Button()

```
Fl_Button::Fl_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

The constructor creates the button using the given position, size, and label.

The default box type is `box(FL_UP_BOX)`.

You can control how the button is drawn when ON by setting `down_box()`. The default is `FL_NO_BOX` (0) which will select an appropriate box type using the normal (OFF) box type by using `fl_down(box())`.

Derived classes may handle this differently.

A button may request callbacks with `when()` `FL_WHEN_CHANGED`, `FL_WHEN_NOT_CHANGED`, and `FL_WHEN_RELEASE`, triggering the callback reasons `FL_REASON_CHANGED`, `FL_REASON_SELECTED`, and `FL_REASON_DESELECTED`.

Parameters

in	<code>X,Y,W,H</code>	position and size of the widget
in	<code>L</code>	widget label, default is no label

11.15.3 Member Function Documentation

11.15.3.1 clear()

```
int Fl_Button::clear () [inline]
Same as value(0).
```

See also

[value\(int v\)](#)

11.15.3.2 compact() [1/2]

```
uchar Fl_Button::compact () [inline]
```

Return true if buttons are rendered as compact buttons.

Returns

0 if compact mode is off, 1 if it is on

See also

`compact(bool)`

11.15.3.3 compact() [2/2]

```
void Fl_Button::compact (
    uchar v)
```

Decide if buttons should be rendered in compact mode.



Figure 11.4 compact button keypad using GTK+ Scheme



Figure 11.5 compact buttons in Gleam

In compact mode, the button's surrounding border is altered to visually signal that multiple buttons are functionally linked together. To ensure the correct rendering of buttons in compact mode, all buttons must be part of the same group, positioned close to each other, and aligned with the edges of the group. Any button outlines not in contact with the parent group's outline will be displayed as separators.

Parameters

in	v	switch compact mode on (1) or off (0)

11.15.3.4 down_box() [1/2]

```
Fl_Boxtype Fl_Button::down_box () const [inline]
```

Returns the current down box type, which is drawn when `value()` is non-zero.

Return values

<code>Fl_Boxtype</code>	

11.15.3.5 down_box() [2/2]

```
void Fl_Button::down_box (
    Fl_Boxtype b) [inline]
```

Sets the down box type.

The default value of 0 causes FLTK to figure out the correct matching down version of [box\(\)](#).

Some derived classes (e.g. [Fl_Round_Button](#) and [Fl_Light_Button](#) use [down_box\(\)](#) for special purposes. See docs of these classes.

Parameters

in	<code>b</code>	down box type

11.15.3.6 draw()

```
void Fl_Button::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                  // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

Reimplemented in [Fl_Light_Button](#), [Fl_Return_Button](#), and [Fl_Shortcut_Button](#).

11.15.3.7 handle()

```
int Fl_Button::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<code>event</code>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Light_Button](#), [Fl_Repeat_Button](#), [Fl_Return_Button](#), and [Fl_Shortcut_Button](#).

11.15.3.8 set()

```
int Fl_Button::set () [inline]
```

Same as `value(1)`.

See also

[value\(int v\)](#)

11.15.3.9 shortcut() [1/2]

```
int Fl_Button::shortcut () const [inline]
```

Returns the current shortcut key for the button.

Return values

<i>int</i>	

11.15.3.10 shortcut() [2/2]

```
void Fl_Button::shortcut (
    int s) [inline]
```

Sets the shortcut key to `s`.

Setting this overrides the use of '&' in the [label\(\)](#). The value is a bitwise OR of a key and a set of shift flags, for example: `FL_ALT | 'a'`, or `FL_ALT | (FL_F + 10)`, or just `'a'`. A value of 0 disables the shortcut.

The key can be any value returned by [Fl::event_key\(\)](#), but will usually be an ASCII letter. Use a lower-case letter unless you require the shift key to be held down.

The shift flags can be any set of values accepted by [Fl::event_state\(\)](#). If the bit is on, that shift key must be pushed. Meta, Alt, Ctrl, and Shift must be off if they are not in the shift flags (zero for the other bits indicates a "don't care" setting).

Parameters

<i>in</i>	<i>s</i>	bitwise OR of key and shift flags

11.15.3.11 value()

```
int Fl_Button::value (
    int v)
```

Sets the current value of the button.

A non-zero value sets the button to 1 (ON), and zero sets it to 0 (OFF).

Parameters

<i>in</i>	<i>v</i>	button value.

See also

[set\(\)](#), [clear\(\)](#)

The documentation for this class was generated from the following files:

- [Fl_Button.H](#)
- [Fl_Button.cxx](#)

11.16 Fl_Cairo_State Class Reference

Contains all the necessary info on the current cairo context.

```
#include <Fl_Cairo.H>
```

Public Member Functions

- bool **autolink** () const
Gets the autolink option. See [Fl::cairo_autolink_context\(bool\)](#)
- void **autolink** (bool b)
Sets the autolink option, only available with `--enable-cairoext`.
- cairo_t * **cc** () const
Gets the current cairo context.
- void **cc** (cairo_t *c, bool own=true)
Sets the current cairo context.
- void * **gc** () const
Gets the last gc attached to a cc.
- void **gc** (void *c)
Sets the gc c to keep track on.
- void * **window** () const
Gets the last window attached to a cc.
- void **window** (void *w)
Sets the window w to keep track on.

11.16.1 Detailed Description

Contains all the necessary info on the current cairo context.

A private internal & unique corresponding object is created to permit cairo context state handling while keeping it opaque. For internal use only.

Note

Only available when configure has the `--enable-cairo` or `--enable-cairoext` option or one or both of the CMake options `FLTK_OPTION_CAIRO_WINDOW` or `FLTK_OPTION_CAIRO_EXT` is set (ON)

11.16.2 Member Function Documentation

11.16.2.1 cc()

```
void Fl_Cairo_State::cc (
    cairo_t * c,
    bool own = true) [inline]
```

Sets the current cairo context.

`own == true` (the default) indicates that the cairo context `c` will be deleted by FLTK internally when another `cc` is set later.

`own == false` indicates `cc` deletion is handled externally by the user program.

The documentation for this class was generated from the following files:

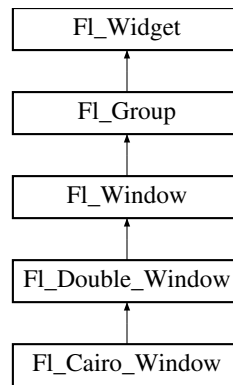
- [Fl_Cairo.H](#)
- [Fl_Cairo.cxx](#)

11.17 Fl_Cairo_Window Class Reference

This defines an FLTK window with Cairo support.

```
#include <Fl_Cairo_Window.H>
```

Inheritance diagram for `Fl_Cairo_Window`:



Public Types

- typedef void(*) **cairo_draw_cb**([Fl_Cairo_Window](#) *self, cairo_t *def)
The Cairo draw callback prototype you need to implement.

Public Types inherited from [Fl_Window](#)

- typedef struct HICON__ * **HICON**

Public Member Functions

- **Fl_Cairo_Window** (int W, int H, const char *L=0)
- **Fl_Cairo_Window** (int X, int Y, int W, int H, const char *L=0)
- void **set_draw_cb** ([cairo_draw_cb](#) cb)
You must provide a draw callback that implements your Cairo rendering.

Public Member Functions inherited from [Fl_Double_Window](#)

- [Fl_Double_Window](#) * **as_double_window** () **FL_OVERRIDE**
Return non-null if this is an [Fl_Double_Window](#) object.
- **Fl_Double_Window** (int W, int H, const char *l=0)
Creates a new [Fl_Double_Window](#) widget using the given position, size, and label (title) string.
- **Fl_Double_Window** (int X, int Y, int W, int H, const char *l=0)
*See [Fl_Double_Window::Fl_Double_Window](#)(int w, int h, const char *label = 0)*
- void **flush** () **FL_OVERRIDE**
Forces the window to be drawn, this window is also made current and calls [draw\(\)](#).
- void **hide** () **FL_OVERRIDE**
Makes a widget invisible.
- void **resize** (int, int, int, int) **FL_OVERRIDE**
Changes the size or position of the widget.
- void **show** () **FL_OVERRIDE**
Makes a widget visible.
- void **show** (int a, char **b)
*Same as [Fl_Window::show](#)(int a, char **b)*
- **~Fl_Double_Window** ()
The destructor also deletes all the children.

Public Member Functions inherited from FL_Window

- void [allow_expand_outside_parent](#) ()
Allow this subwindow to expand outside the area of its parent window.
- virtual class [FL_Overlay_Window](#) * [as_overlay_window](#) ()
Return non-null if this is an [FL_Overlay_Window](#) object.
- [FL_Window](#) const * [as_window](#) () const [FL_OVERRIDE](#)
- [FL_Window](#) * [as_window](#) () [FL_OVERRIDE](#)
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).
- unsigned int [border](#) () const
Returns whether the window possesses a border.
- void [border](#) (int b)
Sets whether or not the window manager border is around the window.
- void [clear_border](#) ()
Fast inline function to turn the window manager border off.
- void [clear_modal_states](#) ()
Clears the "modal" flags and converts a "modal" or "non-modal" window back into a "normal" window.
- void [copy_label](#) (const char *a)
Sets the window titlebar label to a copy of a character string.
- void [cursor](#) (const [FL_RGB_Image](#) *, int, int)
Changes the cursor for this window using the provided image as cursor's shape.
- void [cursor](#) ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void [cursor](#) ([FL_Cursor](#))
Changes the cursor for this window.
- int [decorated_h](#) () const
Returns the window height including any window title bar and any frame added by the window manager.
- int [decorated_w](#) () const
Returns the window width including any frame added by the window manager.
- void [default_cursor](#) ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void [default_cursor](#) ([FL_Cursor](#))
Sets the default window cursor.
- void [draw_backdrop](#) ()
Draw the background image if one is set and is aligned inside.
- [FL_Window](#) (int w, int h, const char *title=0)
Creates a window from the given width w, height h, and title.
- [FL_Window](#) (int x, int y, int w, int h, const char *title=0)
Creates a window from the given position (x, y), size (w, h) and title.
- void [free_position](#) ()
Undoes the effect of a previous [resize\(\)](#) or [show\(\)](#) so that the next time [show\(\)](#) is called the window manager is free to position the window.
- void [fullscreen](#) ()
Makes the window completely fill one or more screens, without any window manager border visible.
- unsigned int [fullscreen_active](#) () const
Returns non zero if [FULLSCREEN](#) flag is set, 0 otherwise.
- void [fullscreen_off](#) ()
Turns off any side effects of [fullscreen\(\)](#)
- void [fullscreen_off](#) (int X, int Y, int W, int H)
Turns off any side effects of [fullscreen\(\)](#) and does [resize\(x,y,w,h\)](#).
- void [fullscreen_screens](#) (int top, int bottom, int left, int right)

Sets which screens should be used when this window is in fullscreen mode.

- `uchar get_size_range` (int *minw, int *minh, int *maxw=NULL, int *maxh=NULL, int *dw=NULL, int *dh=NULL, int *aspect=NULL)

Gets the allowable range to which the user can resize this window.

- int `handle` (int) `FL_OVERRIDE`

Handles the specified event.

- void `hide` () `FL_OVERRIDE`

Removes the window from the screen.

- void `hotspot` (const `FL_Widget` &p, int offscreen=0)

See void `FL_Window::hotspot`(int x, int y, int offscreen = 0)

- void `hotspot` (const `FL_Widget` *, int offscreen=0)

See void `FL_Window::hotspot`(int x, int y, int offscreen = 0)

- void `hotspot` (int x, int y, int offscreen=0)

Positions the window so that the mouse is pointing at the given position, or at the center of the given widget, which may be the window itself.

- const void * `icon` () const

Gets the current icon window target dependent data.

- void `icon` (const `FL_RGB_Image` *)

Sets or resets a single window icon.

- void `icon` (const void *ic)

Platform-specific method to set the window icon usable on Windows and X11 only.

- void `iconize` ()

Iconifies the window.

- const char * `iconlabel` () const

See void `FL_Window::iconlabel`(const char)*

- void `iconlabel` (const char *)

Sets the icon label.

- void `icons` (const `FL_RGB_Image` *[], int)

Sets the window icons.

- void `icons` (HICON big_icon, HICON small_icon)

Sets the window icons using HICON handles (Windows platform only).

- const char * `label` () const

See void `FL_Window::label`(const char)*

- void `label` (const char *)

Sets the window title bar label.

- void `label` (const char *label, const char *iconlabel)

Sets the icon label.

- void `make_current` ()

Sets things up so that the drawing functions in `<FL/fl_draw.H>` will go into this window.

- void `maximize` ()

Maximizes a top-level window to its current screen.

- unsigned int `maximize_active` () const

Returns whether the window is currently maximized.

- unsigned int `menu_window` () const

Returns true if this window is a menu window.

- unsigned int `modal` () const

Returns true if this window is modal.

- unsigned int `non_modal` () const

Returns true if this window is modal or non-modal.

- `fl_uintptr_t os_id ()`
Returns a platform-specific identification of a shown window, or 0 if not shown.
- `unsigned int override () const`
Returns non zero if OVERRIDE flag is set, 0 otherwise.
- `void resize (int X, int Y, int W, int H) FL_OVERRIDE`
Changes the size and position of the window.
- `int screen_num ()`
The number of the screen containing the mapped window.
- `void screen_num (int screen_num)`
Set the number of the screen where to map the window.
- `void set_menu_window ()`
Marks the window as a menu window.
- `void set_modal ()`
A "modal" window, when `shown()`, will prevent any events from being delivered to other windows in the same program, and will also remain on top of the other windows (if the X window manager supports the "transient for" property).
- `void set_non_modal ()`
A "non-modal" window (terminology borrowed from Microsoft Windows) acts like a `modal()` one in that it remains on top, but it has no effect on event delivery.
- `void set_override ()`
Activates the flags NOBORDER|OVERRIDE.
- `void set_tooltip_window ()`
Marks the window as a tooltip window.
- `const FL_Image * shape ()`
Returns the image controlling the window shape or NULL.
- `void shape (const FL_Image &b)`
Set the window's shape with an `FL_Image`.
- `void shape (const FL_Image *img)`
Assigns a non-rectangular shape to the window.
- `void show () FL_OVERRIDE`
Puts the window on the screen.
- `void show (int argc, char **argv)`
Puts the window on the screen with `show()` and parses command-line arguments.
- `int shown ()`
Returns non-zero if `show()` has been called (but not `hide()`).
- `void size_range (int minw, int minh, int maxw=0, int maxh=0, int dw=0, int dh=0, int aspect=0)`
Sets the allowable range to which the user can resize this window.
- `unsigned int tooltip_window () const`
Returns true if this window is a tooltip window.
- `void un_maximize ()`
Returns a previously maximized top-level window to its previous size.
- `void wait_for_expose ()`
Waits for the window to be displayed after calling `show()`.
- `int x_root () const`
Gets the x position of the window on the screen.
- `const char * xclass () const`
Returns the xclass for this window, or a default.
- `void xclass (const char *c)`
Sets the xclass for this window.
- `int y_root () const`
Gets the y position of the window on the screen.
- `virtual ~FL_Window ()`
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *[_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void [add](#) ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void [add](#) ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void [add_resizable](#) ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * [array](#) () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * [as_group](#) () const [FL_OVERRIDE](#)
- [FL_Group](#) * [as_group](#) () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void [begin](#) ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * [child](#) (int n) const
Returns the n'th child.
- int [children](#) () const
Returns how many child widgets the group has.
- void [clear](#) ()
Deletes all child widgets from memory recursively.
- unsigned int [clip_children](#) ()
Returns the current clipping mode.
- void [clip_children](#) (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int [delete_child](#) (int n)
*Removes the widget at *index* from the group and deletes it.*
- void [end](#) ()
*Exactly the same as *current(this->parent())*.*
- int [find](#) (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int [find](#) (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char *==0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void [focus](#) ([FL_Widget](#) *W)
- void [forms_end](#) ()
This is for forms compatibility only.
- void [init_sizes](#) ()
Resets the internal array of widget sizes and positions.
- void [insert](#) ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void [insert](#) ([FL_Widget](#) &o, [FL_Widget](#) *before)
*This does *insert(w, find(before))*.*
- void [remove](#) ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void [remove](#) ([FL_Widget](#) *o)
Removes the widget o from the group.
- void [remove](#) (int index)

- Removes the widget at `index` from the group but does not delete it.*

 - `FI_Widget * resizable () const`
 - Returns the group's resizable widget.*
 - `void resizable (FI_Widget &o)`
 - Sets the group's resizable widget.*
 - `void resizable (FI_Widget *o)`
 - The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
 - `virtual ~FI_Group ()`
 - The destructor also deletes all the children.*

Public Member Functions inherited from FI_Widget

- `void _clear_fullscreen ()`
- `void _set_fullscreen ()`
- `void activate ()`
 - Activates the widget.*
- `unsigned int active () const`
 - Returns whether the widget is active.*
- `int active_r () const`
 - Returns whether the widget and all of its parents are active.*
- `FI_Align align () const`
 - Gets the label alignment.*
- `void align (FI_Align alignment)`
 - Sets the label alignment.*
- `long argument () const`
 - Gets the current user data (long) argument that is passed to the callback function.*
- `void argument (long v)`
 - Sets the current user data (long) argument that is passed to the callback function.*
- `virtual class FI_Gl_Window * as_gl_window ()`
 - Returns an `FI_Gl_Window` pointer if this widget is an `FI_Gl_Window`.*
- `virtual class FI_Gl_Window const * as_gl_window () const`
- `void bind_deimage (FI_Image *img)`
 - Sets the image to use as part of the widget label when in the inactive state.*
- `void bind_deimage (int f)`
 - Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- `void bind_image (FI_Image *img)`
 - Sets the image to use as part of the widget label when in the active state.*
- `void bind_image (int f)`
 - Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- `FI_Boxtype box () const`
 - Gets the box type of the widget.*
- `void box (FI_Boxtype new_box)`
 - Sets the box type for the widget.*
- `FI_Callback_p callback () const`
 - Gets the current callback function for the widget.*
- `void callback (FI_Callback *cb)`
 - Sets the current callback function for the widget.*
- `void callback (FI_Callback *cb, FI_Callback_User_Data *p, bool auto_free)`
 - Sets the current callback function and managed user data for the widget.*
- `void callback (FI_Callback *cb, void *p)`
 - Sets the current callback function and data for the widget.*

- void `callback` (`FI_Callback0` *cb)
Sets the current callback function for the widget.
- void `callback` (`FI_Callback1` *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int `changed` () const
Checks if the widget value changed since the last callback.
- void `clear_active` ()
Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()
Marks the value of the widget as unchanged.
- void `clear_damage` (`uchar` c=0)
Clears or sets the damage flags.
- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FI_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FI_Color` bg)
Sets the background color of the widget.
- void `color` (`FI_Color` bg, `FI_Color` sel)
Sets the background and selection color of the widget.
- `FI_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FI_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int **`damage_resize`** (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FI_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FI_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FI_Image` *img)

- Sets the image to use as part of the widget label when in the inactive state.*

 - int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FI_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.
- void `do_callback` (FI_Widget *widget, long arg, FI_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with arbitrary arguments.
- void `do_callback` (FI_Widget *widget, void *arg=0, FI_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, FI_Align) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const

Gets the widget height.
- int `horizontal_label_margin` ()

Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)

Set the spacing between the label and the horizontal edge of the widget.
- FI_Image * `image` ()

Gets the image that is used as part of the widget label when in the active state.
- const FI_Image * `image` () const

Gets the image that is used as part of the widget label when in the active state.
- void `image` (FI_Image &img)

Sets the image to use as part of the widget label when in the active state.
- void `image` (FI_Image *img)

Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const

Returns whether the image is managed by the widget.
- int `inside` (const FI_Widget *wgt) const

Checks if this widget is a child of wgt.
- int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const

Gets the current label text.
- void `label` (const char *text)

Sets the current label pointer.
- void `label` (FI_Labeltype a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- FI_Color `labelcolor` () const

Gets the label color.
- void `labelcolor` (FI_Color c)

Sets the label color.
- FI_Font `labelfont` () const

Gets the font to use.
- void `labelfont` (FI_Font f)

Sets the font to use.
- FI_Fontsize `labelsize` () const

Gets the font size in pixels.

- void [labelsize](#) ([Fl_Fontsize](#) pix)
Sets the font size in pixels.
- [Fl_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([Fl_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [Fl_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([Fl_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [Fl_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

- Gets the current tooltip text.*
- void **tooltip** (const char *text)
 - Sets the current tooltip text.*
- **Fl_Window * top_window** () const
 - Returns a pointer to the top-level window for the widget.*
- **Fl_Window * top_window_offset** (int &xoff, int &yoff) const
 - Finds the x/y offset of the current widget relative to the top-level window.*
- **uchar type** () const
 - Gets the widget type.*
- void **type** (uchar t)
 - Sets the widget type.*
- int **use_accents_menu** ()
 - Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*
- void * **user_data** () const
 - Gets the user data for this widget.*
- void **user_data** (Fl_Callback_User_Data *v, bool auto_free)
 - Sets the user data for this widget.*
- void **user_data** (void *v)
 - Sets the user data for this widget.*
- int **vertical_label_margin** ()
 - Get the spacing between the label and the vertical edge of the widget.*
- void **vertical_label_margin** (int px)
 - Set the spacing between the label and the vertical edge of the widget.*
- unsigned int **visible** () const
 - Returns whether a widget is visible.*
- unsigned int **visible_focus** () const
 - Checks whether this widget has a visible focus.*
- void **visible_focus** (int v)
 - Modifies keyboard focus navigation.*
- int **visible_r** () const
 - Returns whether a widget and all its parents are visible.*
- int **w** () const
 - Gets the widget width.*
- **Fl_When when** () const
 - Returns the conditions under which the callback is called.*
- void **when** (uchar i)
 - Sets the flags used to decide when a callback is called.*
- **Fl_Window * window** () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
- int **x** () const
 - Gets the widget position in its window.*
- int **y** () const
 - Gets the widget position in its window.*
- virtual **~Fl_Widget** ()
 - Destroys the widget.*

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
 - Overloaded to provide Cairo callback support.*

Protected Member Functions inherited from [FL_Window](#)

- void [default_size_range](#) ()
Protected method to calculate the default size range of a window.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- int [force_position](#) () const
Returns the internal state of the window's FORCE_POSITION flag.
- void [force_position](#) (int force)
Sets an internal flag that tells FLTK and the window manager to honor position requests.
- void [free_icons](#) ()
Deletes all icons previously attached to the window.
- int [is_resizable](#) ()
Protected method to determine whether a window is resizable.

Protected Member Functions inherited from [FL_Group](#)

- [FL_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw_child](#) ([FL_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [FL_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) ([FL_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int [on_move](#) (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void [on_remove](#) (int)
Allow derived groups to act when a child widget is removed from the group.
- int * [sizes](#) ()
Returns the internal array of widget sizes and positions.
- void [update_child](#) ([FL_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FL_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FL_Boxtype](#) t, [FL_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FL_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.

- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Window

- static FI_Window * **current** ()
Returns the last window that was made current.
- static void **default_callback** (FI_Window *, void *v)
Back compatibility: Sets the default callback v for win to call on close event.
- static void **default_icon** (const FI_RGB_Image *)
Sets a single default window icon.
- static void **default_icons** (const FI_RGB_Image *[], int)
Sets the default window icons.
- static void **default_icons** (HICON big_icon, HICON small_icon)
Sets the default window icons (Windows platform only).
- static const char * **default_xclass** ()
Returns the default xclass.
- static void **default_xclass** (const char *)
Sets the default window xclass.
- static bool **is_a_rescale** ()
Returns true when a window is being rescaled.
- static char **show_next_window_iconic** ()
Returns the static flag whether the next window should be opened iconified.
- static void **show_next_window_iconic** (char stat)
Sets a static flag whether the next window should be opened iconified.

Static Public Member Functions inherited from FI_Group

- static FI_Group * **current** ()
Returns the currently active group.
- static void **current** (FI_Group *g)
Sets the current group.

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0, [INVISIBLE](#) = 1<<1, [OUTPUT](#) = 1<<2, [NOBORDER](#) = 1<<3,
[FORCE_POSITION](#) = 1<<4, [NON_MODAL](#) = 1<<5, [SHORTCUT_LABEL](#) = 1<<6, [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8, [VISIBLE_FOCUS](#) = 1<<9, [COPIED_LABEL](#) = 1<<10, [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12, [TOOLTIP_WINDOW](#) = 1<<13, [MODAL](#) = 1<<14, [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16, [COPIED_TOOLTIP](#) = 1<<17, [FULLSCREEN](#) = 1<<18, [MAC_USE_ACCENTS_MENU](#)
= 1<<19,
[NEEDS_KEYBOARD](#) = 1<<20, [IMAGE_BOUND](#) = 1<<21, [DEIMAGE_BOUND](#) = 1<<22,
[AUTO_DELETE_USER_DATA](#) = 1<<23,
[MAXIMIZED](#) = 1<<24, [POPUP](#) = 1<<25, [USERFLAG3](#) = 1<<29, [USERFLAG2](#) = 1<<30,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Static Protected Attributes inherited from [Fl_Window](#)

- static [Fl_Window](#) * [current_](#)
Stores the last window that was made current.

11.17.1 Detailed Description

This defines an FLTK window with Cairo support.

This class overloads the virtual [draw\(\)](#) method for you, so that the only thing you have to do is to provide your Cairo code. All Cairo context handling is achieved transparently.

The default coordinate system for Cairo drawing commands within [Fl_Cairo_Window](#) is FLTK's coordinate system, where the x, y, w, h values are relative to the top/left corner of the [Fl_Cairo_Window](#), as one would expect with regular FLTK drawing commands, e.g.: (0 x w-1), (0 y h-1). **Example:**

```
static void my_cairo_draw_cb(Fl\_Cairo\_Window *window, cairo_t *cr) {
    // Draw an "X"
    const double xmax = (window->w() - 1);
    const double ymax = (window->h() - 1);
    cairo_set_line_width(cr, 1.00); // line width for drawing
    cairo_set_source_rgb(cr, 1.0, 0.5, 0.0); // orange
    cairo_move_to(cr, 0.0, 0.0); cairo_line_to(cr, xmax, ymax); // draw diagonal "\"
    cairo_move_to(cr, 0.0, ymax); cairo_line_to(cr, xmax, 0.0); // draw diagonal "/"
    cairo_stroke(cr); // stroke the lines
}
```

The FLTK coordinate system differs from the default native Cairo coordinate system which uses normalized (0.0 ... 1.0) values for x and y, e.g.: (0 x 1.0), (0 y 1.0). So beware of this when copy/pasting Cairo example programs that assume normalized values. If need be, you can revert to the Cairo coordinate system by simply calling [cairo_scale\(\)](#) with the widget's [w\(\)](#) and [h\(\)](#) values. **Example:**

```
static void my_cairo_draw_cb(Fl\_Cairo\_Window *window, cairo_t *cr) {
    cairo_scale(cr, window->w(), window->h()); // use Cairo's default coordinate system
    [...use 0.0 to 1.0 values from here on..]
}
```

See also

examples/cairo-draw-x.cxx
test/cairo_test.cxx

Note

Class [Fl_Cairo_Window](#) requires the FLTK library to have been built with CMake option `FLTK_OPTION_CAIRO_WINDOW` or configure `--enable-cairo`.

You can alternatively define your custom Cairo FLTK window, and thus at least override the [draw\(\)](#) method to provide custom Cairo support. In this case you will probably use [Fl::cairo_make_current\(Fl_Window*\)](#) to attach a context to your window. You should do this only when your window is the current window.

See also

[Fl_Window::current\(\)](#)

11.17.2 Member Function Documentation**11.17.2.1 draw()**

```
void Fl_Cairo_Window::draw (
    void ) [inline], [protected], [virtual]
```

Overloaded to provide Cairo callback support.

Implements [Fl_Widget](#).

11.17.2.2 set_draw_cb()

```
void Fl_Cairo_Window::set_draw_cb (
    cairo_draw_cb cb) [inline]
```

You must provide a draw callback that implements your Cairo rendering.

This method permits you to set your Cairo callback to `cb`.

The documentation for this class was generated from the following file:

- [Fl_Cairo_Window.H](#)

11.18 FI_Callback_User_Data Class Reference

A class prototype that allows for additional data in callbacks.

```
#include <Fl_Widget.H>
```

Public Member Functions

- virtual [~FI_Callback_User_Data \(\)](#)
Destructor.

Protected Member Functions

- [FI_Callback_User_Data \(\)](#)
Protected constructor.

11.18.1 Detailed Description

A class prototype that allows for additional data in callbacks.

Users can derive this class and pass objects of such derived classes to widget callbacks. Widgets can take ownership of the callback data, deleting the data when the widget itself is deleted.

The destructor of this class is virtual, allowing for additional code to deallocate resources when the user data is deleted.

See also

[FL_FUNCTION_CALLBACK_3](#), [FL_METHOD_CALLBACK_1](#), [FL_INLINE_CALLBACK_2](#)

[Fl_Widget::callback\(Fl_Callback*, Fl_Callback_User_Data*, bool\)](#)

[Fl_Widget::user_data\(Fl_Callback_User_Data*, bool\)](#)

The documentation for this class was generated from the following file:

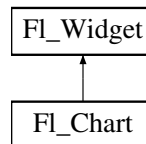
- [Fl_Widget.H](#)

11.19 Fl_Chart Class Reference

[Fl_Chart](#) displays simple charts.

```
#include <Fl_Chart.H>
```

Inheritance diagram for [Fl_Chart](#):



Public Member Functions

- void [add](#) (double val, const char *str=0, unsigned col=0)
Adds the data value `val` with optional label `str` and color `col` to the chart.
- [uchar autosize](#) () const
Gets whether the chart will automatically adjust the bounds of the chart.
- void [autosize](#) (uchar n)
Sets whether the chart will automatically adjust the bounds of the chart.
- void [bounds](#) (double *a, double *b) const
Gets the lower and upper bounds of the chart values.
- void [bounds](#) (double a, double b)
Sets the lower and upper bounds of the chart values.
- void [clear](#) ()
Removes all values from the chart.
- [Fl_Chart](#) (int X, int Y, int W, int H, const char *L=0)
Create a new [Fl_Chart](#) widget using the given position, size and label string.
- void [insert](#) (int ind, double val, const char *str=0, unsigned col=0)
Inserts a data value `val` at the given position `ind`.
- int [maxsize](#) () const
Gets the maximum number of data values for a chart.
- void [maxsize](#) (int m)
Sets the maximum number of data values for a chart.
- void [replace](#) (int ind, double val, const char *str=0, unsigned col=0)
Replaces a data value `val` at the given position `ind`.
- int [size](#) () const
Returns the number of data values in the chart.
- void [size](#) (int W, int H)
Sets the widget size (width, height).
- [Fl_Color textcolor](#) () const
Gets the chart's text color.
- void [textcolor](#) ([Fl_Color](#) n)
Sets the chart's text color to `n`.
- [Fl_Font textfont](#) () const
Gets the chart's text font.
- void [textfont](#) ([Fl_Font](#) s)
Sets the chart's text font to `s`.
- [Fl_Fontsize textsize](#) () const
Gets the chart's text size.
- void [textsize](#) ([Fl_Fontsize](#) s)
Sets the chart's text size to `s`.
- ~[Fl_Chart](#) ()
Destroys the [Fl_Chart](#) widget and all of its data.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (**FI_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FI_Gl_Window** * **as_gl_window** ()
Returns an FI_Gl_Window pointer if this widget is an FI_Gl_Window.
- virtual class **FI_Gl_Window** const * **as_gl_window** () const
- virtual **FI_Group** * **as_group** ()
Returns an FI_Group pointer if this widget is an FI_Group.
- virtual **FI_Group** const * **as_group** () const
- virtual **FI_Window** * **as_window** ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual **FI_Window** const * **as_window** () const
- void **bind_deimage** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- **FI_Boxtype box** () const
Gets the box type of the widget.
- void **box** (**FI_Boxtype** new_box)
Sets the box type for the widget.
- **FI_Callback_p callback** () const
Gets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb, **FI_Callback_User_Data** *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void **callback** (**FI_Callback** *cb, void *p)
Sets the current callback function and data for the widget.
- void **callback** (**FI_Callback0** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback1** *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int **changed** () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FL_Color` bg)

Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)

Sets the background and selection color of the widget.
- `FL_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.

- void `do_callback` (`FL_Widget *widget`, long `arg`, `FL_Callback_Reason` `reason=FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget *widget`, void `*arg=0`, `FL_Callback_Reason` `reason=FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual int `handle` (int `event`)
Handles the specified event.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int `px`)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image * image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image &img`)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image *img`)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget *wgt`) const
Checks if this widget is a child of `wgt`.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char `* label` () const
Gets the current label text.
- void `label` (const char `*text`)
Sets the current label pointer.
- void `label` (`FL_Labeltype` `a`, const char `*b`)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int `gap`)
Set the gap between the label and the image in pixels.
- `FL_Color labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` `c`)
Sets the label color.
- `FL_Font labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` `f`)
Sets the font to use.
- `FL_Fonsize labelsiz` () const
Gets the font size in pixels.
- void `labelsiz` (`FL_Fonsize` `pix`)

- Sets the font size in pixels.*

 - `FI_Labeltype labeltype () const`

Gets the label type.

 - `void labeltype (FI_Labeltype a)`

Sets the label type.

 - `void measure_label (int &ww, int &hh) const`

Sets width ww and height hh accordingly with the label size.

 - `bool needs_keyboard () const`

Returns whether this widget needs a keyboard.

 - `void needs_keyboard (bool needs)`

Sets whether this widget needs a keyboard.

 - `unsigned int output () const`

Returns if a widget is used for output only.

 - `FI_Group * parent () const`

Returns a pointer to the parent widget.

 - `void parent (FI_Group *p)`

Internal use only - "for hacks only".

 - `void position (int X, int Y)`

Repositions the window or widget.

 - `void redraw ()`

Schedules the drawing of the widget.

 - `void redraw_label ()`

Schedules the drawing of the label.

 - `virtual void resize (int x, int y, int w, int h)`

Changes the size or position of the widget.

 - `FI_Color selection_color () const`

Gets the selection color.

 - `void selection_color (FI_Color a)`

Sets the selection color.

 - `void set_active ()`

Marks the widget as active without sending events or changing focus.

 - `void set_changed ()`

Marks the value of the widget as changed.

 - `void set_output ()`

Sets a widget to output only.

 - `void set_visible ()`

Makes the widget visible.

 - `void set_visible_focus ()`

Enables keyboard focus navigation with this widget.

 - `int shortcut_label () const`

Returns whether the widget's label uses '&' to indicate shortcuts.

 - `void shortcut_label (int value)`

Sets whether the widget's label uses '&' to indicate shortcuts.

 - `virtual void show ()`

Makes a widget visible.

 - `void size (int W, int H)`

Changes the size of the widget.

 - `int take_focus ()`

Gives the widget the keyboard focus.

 - `unsigned int takeevents () const`

Returns if the widget is able to take events.

- int `test_shortcut` ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const
Gets the current tooltip text.
- void `tooltip` (const char *text)
Sets the current tooltip text.
- Fl_Window * `top_window` () const
Returns a pointer to the top-level window for the widget.
- Fl_Window * `top_window_offset` (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- uchar `type` () const
Gets the widget type.
- void `type` (uchar t)
Sets the widget type.
- int `use_accents_menu` ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * `user_data` () const
Gets the user data for this widget.
- void `user_data` (Fl_Callback_User_Data *v, bool auto_free)
Sets the user data for this widget.
- void `user_data` (void *v)
Sets the user data for this widget.
- int `vertical_label_margin` ()
Get the spacing between the label and the vertical edge of the widget.
- void `vertical_label_margin` (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int `visible` () const
Returns whether a widget is visible.
- unsigned int `visible_focus` () const
Checks whether this widget has a visible focus.
- void `visible_focus` (int v)
Modifies keyboard focus navigation.
- int `visible_r` () const
Returns whether a widget and all its parents are visible.
- int `w` () const
Gets the widget width.
- Fl_When `when` () const
Returns the conditions under which the callback is called.
- void `when` (uchar i)
Sets the flags used to decide when a callback is called.
- Fl_Window * `window` () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual ~`Fl_Widget` ()
Destroys the widget.

Protected Member Functions

- void **draw** () [FL_OVERRIDE](#)

Draws the [FL_Chart](#) widget.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If [FL_ALIGN_IMAGE_BACKDROP](#) is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Static Protected Member Functions

- static void **draw_barchart** (int x, int y, int w, int h, int numb, [FL_CHART_ENTRY](#) entries[], double min, double max, int [autosize](#), int maxnumb, [FI_Color](#) textcolor)
Draws a bar chart.
- static void **draw_horbarchart** (int x, int y, int w, int h, int numb, [FL_CHART_ENTRY](#) entries[], double min, double max, int [autosize](#), int maxnumb, [FI_Color](#) textcolor)
Draws a horizontal bar chart.
- static void **draw_linechart** (int type, int x, int y, int w, int h, int numb, [FL_CHART_ENTRY](#) entries[], double min, double max, int [autosize](#), int maxnumb, [FI_Color](#) textcolor)
Draws a line chart.

- static void `draw_piechart` (int `x`, int `y`, int `w`, int `h`, int `numb`, `FL_CHART_ENTRY` `entries[]`, int `special`, `FI_Color` `textcolor`)

Draws a pie chart.

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *`widget`, void *`data`)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *`t`)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool `require_alt=false`)
Returns true if the given text `t` contains the entered '&x' shortcut.

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

11.19.1 Detailed Description

`FI_Chart` displays simple charts.

It is provided for Forms compatibility.

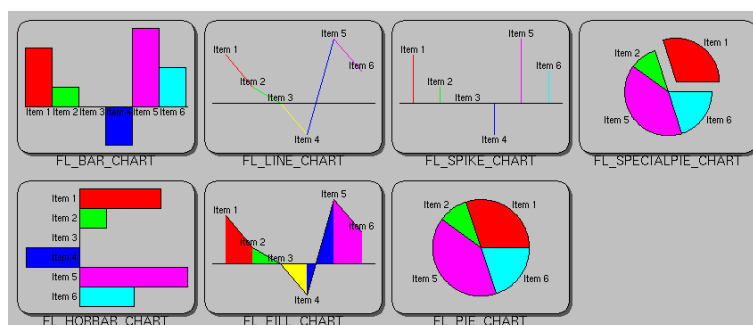


Figure 11.6 `FI_Chart`

Todo Refactor `FI_Chart::type()` information.

The type of an `FI_Chart` object can be set using `type(uchar t)` to:

Chart Type	Description
<code>FL_BAR_CHART</code>	Each sample value is drawn as a vertical bar.

Chart Type	Description
FL_FILLED_CHART	The chart is filled from the bottom of the graph to the sample values.
FL_HORBAR_CHART	Each sample value is drawn as a horizontal bar.
FL_LINE_CHART	The chart is drawn as a polyline with vertices at each sample value.
FL_PIE_CHART	A pie chart is drawn with each sample value being drawn as a proportionate slice in the circle.
FL_SPECIALPIE_CHART	Like FL_PIE_CHART, but the first slice is separated from the pie.
FL_SPIKE_CHART	Each sample value is drawn as a vertical line.

11.19.2 Constructor & Destructor Documentation

11.19.2.1 Fl_Chart()

```
Fl_Chart::Fl_Chart (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Create a new [Fl_Chart](#) widget using the given position, size and label string. The default boxstyle is FL_NO_BOX.

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

11.19.3 Member Function Documentation

11.19.3.1 add()

```
void Fl_Chart::add (
    double val,
    const char * str = 0,
    unsigned col = 0)
```

Adds the data value *val* with optional label *str* and color *col* to the chart.

Parameters

in	<i>val</i>	data value
in	<i>str</i>	optional data label
in	<i>col</i>	optional data color

11.19.3.2 autosize() [1/2]

```
uchar Fl_Chart::autosize () const [inline]
```

Gets whether the chart will automatically adjust the bounds of the chart.

Returns

non-zero if auto-sizing is enabled and zero if disabled.

11.19.3.3 autosize() [2/2]

```
void Fl_Chart::autosize (  
    uchar n) [inline]
```

Sets whether the chart will automatically adjust the bounds of the chart.

Parameters

in	<i>n</i>	non-zero to enable automatic resizing, zero to disable.

11.19.3.4 bounds() [1/2]

```
void Fl_Chart::bounds (
    double * a,
    double * b) const [inline]
```

Gets the lower and upper bounds of the chart values.

Parameters

out	<i>a,b</i>	are set to lower, upper

11.19.3.5 bounds() [2/2]

```
void Fl_Chart::bounds (
    double a,
    double b)
```

Sets the lower and upper bounds of the chart values.

Parameters

in	<i>a,b</i>	are used to set lower, upper

11.19.3.6 draw()

```
void Fl_Chart::draw (
    void ) [protected], [virtual]
```

Draws the [Fl_Chart](#) widget.

Implements [Fl_Widget](#).

11.19.3.7 draw_barchart()

```
void Fl_Chart::draw_barchart (
    int x,
    int y,
    int w,
    int h,
    int numb,
    FL_CHART_ENTRY entries[],
    double min,
    double max,
    int autosize,
    int maxnumb,
    Fl_Color textcolor) [static], [protected]
```

Draws a bar chart.

x, *y*, *w*, *h* is the bounding box, *entries* the array of *numb* entries, and *min* and *max* the boundaries.

Parameters

in	<i>x,y,w,h</i>	Widget position and size
in	<i>numb</i>	Number of values
in	<i>entries</i>	Array of values

Parameters

in	<i>min</i>	Lower boundary
in	<i>max</i>	Upper boundary
in	<i>autosize</i>	Whether the chart autosizes
in	<i>maxnumb</i>	Maximal number of entries
in	<i>textcolor</i>	Text color

11.19.3.8 draw_horbarchart()

```
void Fl_Chart::draw_horbarchart (
    int x,
    int y,
    int w,
    int h,
    int numb,
    FL_CHART_ENTRY entries[],
    double min,
    double max,
    int autosize,
    int maxnumb,
    Fl_Color textcolor) [static], [protected]
```

Draws a horizontal bar chart.

x, *y*, *w*, *h* is the bounding box, *entries* the array of *numb* entries, and *min* and *max* the boundaries.

Parameters

in	<i>x,y,w,h</i>	Widget position and size
in	<i>numb</i>	Number of values
in	<i>entries</i>	Array of values
in	<i>min</i>	Lower boundary
in	<i>max</i>	Upper boundary
in	<i>autosize</i>	Whether the chart autosizes
in	<i>maxnumb</i>	Maximal number of entries
in	<i>textcolor</i>	Text color

11.19.3.9 draw_linechart()

```
void Fl_Chart::draw_linechart (
    int type,
    int x,
    int y,
    int w,
    int h,
    int numb,
    FL_CHART_ENTRY entries[],
    double min,
    double max,
    int autosize,
    int maxnumb,
    Fl_Color textcolor) [static], [protected]
```

Draws a line chart.

x, *y*, *w*, *h* is the bounding box, *entries* the array of *numb* entries, and *min* and *max* the boundaries.

Parameters

in	<i>type</i>	Chart type
in	<i>x,y,w,h</i>	Widget position and size
in	<i>numb</i>	Number of values
in	<i>entries</i>	Array of values
in	<i>min</i>	Lower boundary
in	<i>max</i>	Upper boundary
in	<i>autosize</i>	Whether the chart autosizes
in	<i>maxnumb</i>	Maximal number of entries
in	<i>textcolor</i>	Text color

11.19.3.10 draw_piechart()

```
void Fl_Chart::draw_piechart (
    int x,
    int y,
    int w,
    int h,
    int numb,
    FL_CHART_ENTRY entries[],
    int special,
    Fl_Color textcolor) [static], [protected]
```

Draws a pie chart.

Parameters

in	<i>x,y,w,h</i>	bounding box
in	<i>numb</i>	number of chart entries
in	<i>entries</i>	array of chart entries
in	<i>special</i>	special (?)
in	<i>textcolor</i>	text color

11.19.3.11 insert()

```
void Fl_Chart::insert (
    int ind,
    double val,
    const char * str = 0,
    unsigned col = 0)
```

Inserts a data value *val* at the given position *ind*.
Position 1 is the first data value.

Parameters

in	<i>ind</i>	insertion position
in	<i>val</i>	data value
in	<i>str</i>	optional data label
in	<i>col</i>	optional data color

11.19.3.12 maxsize()

```
void Fl_Chart::maxsize (
    int m)
```

Sets the maximum number of data values for a chart.

If you do not call this method then the chart will be allowed to grow to any size depending on available memory.

Parameters

in	<i>m</i>	maximum number of data values allowed.

11.19.3.13 replace()

```
void Fl_Chart::replace (
    int ind,
    double val,
    const char * str = 0,
    unsigned col = 0)
```

Replaces a data value *val* at the given position *ind*.

Position 1 is the first data value.

Parameters

in	<i>ind</i>	insertion position
in	<i>val</i>	data value
in	<i>str</i>	optional data label
in	<i>col</i>	optional data color

11.19.3.14 size()

```
void Fl_Chart::size (
    int W,
    int H) [inline]
```

Sets the widget size (width, height).

This is the same as calling [FL_Widget::size\(int W, int H\)](#);

Parameters

in	<i>W,H</i>	new width and height of the widget

The documentation for this class was generated from the following files:

- [FL_Chart.H](#)
- [FL_Chart.cxx](#)

11.20 FL_CHART_ENTRY Struct Reference

For internal use only.

```
#include <Fl_Chart.H>
```

Public Attributes

- unsigned **col**
For internal use only.
- char **str** [[FL_CHART_LABEL_MAX](#)+1]

For internal use only.

- float **val**

For internal use only.

11.20.1 Detailed Description

For internal use only.

The documentation for this struct was generated from the following file:

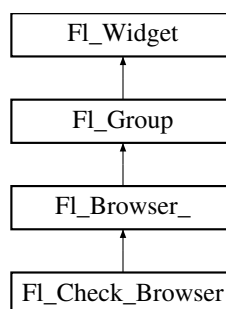
- [Fl_Chart.H](#)

11.21 Fl_Check_Browser Class Reference

The [Fl_Check_Browser](#) widget displays a scrolling list of text lines that may be selected and/or checked by the user.

```
#include <Fl_Check_Browser.H>
```

Inheritance diagram for Fl_Check_Browser:



Public Member Functions

- int [add](#) (char *s)
Add a new unchecked line to the end of the browser.
- int [add](#) (char *s, int b)
Add a new line to the end of the browser.
- int **add** (const char *s)
*See int [Fl_Check_Browser::add\(char *s\)](#)*
- int **add** (const char *s, int b)
*See int [Fl_Check_Browser::add\(char *s\)](#)*
- void **check_all** ()
Sets all the items checked.
- void **check_none** ()
Sets all the items unchecked.
- int **checked** (int item) const
Gets the current status of item item.
- void **checked** (int item, int b)
Sets the check status of item item to b.
- void **clear** ()
Remove every item from the browser.
- **Fl_Check_Browser** (int x, int y, int w, int h, const char *l=0)
The constructor makes an empty browser.
- void * [item_at](#) (int index) const [FL_OVERRIDE](#)
This method must be provided by the subclass to return the item for the specified index.
- void **item_swap** (int ia, int ib)
- void [item_swap](#) (void *a, void *b) [FL_OVERRIDE](#)

*This optional method should be provided by the subclass to efficiently swap browser items *a* and *b*, such as for sorting.*

- int **nchecked** () const
Returns how many items are currently checked.
- int **nitems** () const
Returns how many lines are in the browser.
- int **remove** (int item)
*Remove line *n* and make the browser one line shorter.*
- void **set_checked** (int item)
Equivalent to `FI_Check_Browser::checked(item, 1)`.
- char * **text** (int item) const
*Return a pointer to an internal buffer holding item *item*'s text.*
- int **value** () const
Returns the index of the currently selected item.
- ~**FI_Check_Browser** ()
The destructor deletes all list items and destroys the browser.

Public Member Functions inherited from **FI_Browser_**

- int **deselect** (int docallbacks=0)
Deselects all items in the list and returns 1 if the state changed or 0 if it did not.
- void **display** (void *item)
*Displays the *item*, scrolling the list as necessary.*
- uchar **has_scrollbar** () const
Returns the current scrollbar mode, see `FI_Browser_::has_scrollbar(uchar)`
- void **has_scrollbar** (uchar mode)
Sets whether the widget should have scrollbars or not (default `FI_Browser_::BOTH`).
- int **hposition** () const
*Gets the horizontal scroll position of the list as a pixel position *pos*.*
- void **hposition** (int)
*Sets the horizontal scroll position of the list to pixel position *pos*.*
- int **linespacing** () const
Return the height of additional spacing between browser lines.
- void **linespacing** (int pixels)
Add some space between browser lines.
- int **position** () const
- void **position** (int pos)
- void **position** (int *x*, int *y*)
- void **resize** (int *X*, int *Y*, int *W*, int *H*) **FL_OVERRIDE**
Repositions and/or resizes the browser.
- void **scrollbar_left** ()
Moves the vertical scrollbar to the lefthand side of the list.
- void **scrollbar_right** ()
Moves the vertical scrollbar to the righthand side of the list.
- int **scrollbar_size** () const
Gets the current size of the scrollbars' troughs, in pixels.
- void **scrollbar_size** (int newSize)
*Sets the pixel size of the scrollbars' troughs to *newSize*, in pixels.*
- int **scrollbar_width** () const
Returns the global value `FI::scrollbar_size()`.
- void **scrollbar_width** (int width)

- Sets the global `Fl::scrollbar_size()`, and forces this instance of the widget to use it.
- `int select` (`void *item`, `int val=1`, `int docallbacks=0`)

Sets the selection state of `item` to `val`, and returns 1 if the state changed or 0 if it did not.
- `int select_only` (`void *item`, `int docallbacks=0`)

Selects `item` and returns 1 if the state changed or 0 if it did not.
- `void sort` (`int flags=0`)

Sort the items in the browser based on `flags`.
- `Fl_Color textcolor` () `const`

Gets the default text color for the lines in the browser.
- `void textcolor` (`Fl_Color col`)

Sets the default text color for the lines in the browser to color `col`.
- `Fl_Font textfont` () `const`

Gets the default text font for the lines in the browser.
- `void textfont` (`Fl_Font font`)

Sets the default text font for the lines in the browser to `font`.
- `Fl_Fonsize textsize` () `const`

Gets the default text size (in pixels) for the lines in the browser.
- `void textsize` (`Fl_Fonsize newSize`)

Sets the default text size (in pixels) for the lines in the browser to `size`.
- `int vposition` () `const`

Gets the vertical scroll position of the list as a pixel position `pos`.
- `void vposition` (`int pos`)

Sets the vertical scroll position of the list to pixel position `pos`.

Public Member Functions inherited from `Fl_Group`

- `Fl_Widget *& _ddfdesign_kludge` ()

This is for forms compatibility only.
- `void add` (`Fl_Widget &`)

The widget is removed from its current group (if any) and then added to the end of this group.
- `void add` (`Fl_Widget *o`)

See `void Fl_Group::add(Fl_Widget &w)`
- `void add_resizable` (`Fl_Widget &o`)

Adds a widget to the group and makes it the resizable widget.
- `Fl_Widget *const * array` () `const`

Returns a pointer to the internal array of children.
- `Fl_Group const * as_group` () `const` `FL_OVERRIDE`
- `Fl_Group * as_group` () `FL_OVERRIDE`

Returns an `Fl_Group` pointer if this widget is an `Fl_Group`.
- `void begin` ()

Sets the current group so you can build the widget tree by just constructing the widgets.
- `Fl_Widget * child` (`int n`) `const`

Returns the `n`'th child.
- `int children` () `const`

Returns how many child widgets the group has.
- `void clear` ()

Deletes all child widgets from memory recursively.
- `unsigned int clip_children` ()

Returns the current clipping mode.
- `void clip_children` (`int c`)

Controls whether the group widget clips the drawing of child widgets to its bounding box.

- virtual int `delete_child` (int n)
Removes the widget at `index` from the group and deletes it.
- void `end` ()
Exactly the same as `current(this->parent())`.
- int `find` (const `FL_Widget` &o) const
See int `FL_Group::find(const FL_Widget *w)` const.
- int `find` (const `FL_Widget` *) const
Searches the children for the widget and returns the index.
- `FL_Group` (int, int, int, int, const char *s=0)
Creates a new `FL_Group` widget using the given position, size, and label string.
- void `focus` (`FL_Widget` *W)
- void `forms_end` ()
This is for forms compatibility only.
- int `handle` (int) `FL_OVERRIDE`
Handles the specified event.
- void `init_sizes` ()
Resets the internal array of widget sizes and positions.
- void `insert` (`FL_Widget` &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void `insert` (`FL_Widget` &o, `FL_Widget` *before)
This does `insert(w, find(before))`.
- void `remove` (`FL_Widget` &)
Removes a widget from the group but does not delete it.
- void `remove` (`FL_Widget` *o)
Removes the widget o from the group.
- void `remove` (int index)
Removes the widget at `index` from the group but does not delete it.
- `FL_Widget` * `resizable` () const
Returns the group's resizable widget.
- void `resizable` (`FL_Widget` &o)
Sets the group's resizable widget.
- void `resizable` (`FL_Widget` *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- void `resize` (int, int, int, int) `FL_OVERRIDE`
Resizes the `FL_Group` widget and all of its children.
- virtual `~FL_Group` ()
The destructor also deletes all the children.

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
Activates the widget.
- unsigned int `active` () const
Returns whether the widget is active.
- int `active_r` () const
Returns whether the widget and all of its parents are active.
- `FL_Align` `align` () const
Gets the label alignment.
- void `align` (`FL_Align` alignment)

- Sets the label alignment.*

 - long [argument](#) () const

Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)

Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()

Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Window](#) * [as_window](#) ()

Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const

Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)

Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const

Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()

Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.
- void [clear_output](#) ()

Sets a widget to accept input.
- void [clear_visible](#) ()

Hides the widget.
- void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const

Gets the background color of the widget.

- void `color` (`FL_Color` bg)
Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)
Sets the background and selection color of the widget.
- `FL_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FL_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FL_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()

- Gets the image that is used as part of the widget label when in the active state.*

 - `const FI_Image * image () const`
- Gets the image that is used as part of the widget label when in the active state.*

 - `void image (FI_Image &img)`
- Sets the image to use as part of the widget label when in the active state.*

 - `void image (FI_Image *img)`
- Sets the image to use as part of the widget label when in the active state.*

 - `int image_bound () const`
- Returns whether the image is managed by the widget.*

 - `int inside (const FI_Widget *wgt) const`
- Checks if this widget is a child of wgt.*

 - `int is_label_copied () const`
- Returns whether the current label was assigned with copy_label().*

 - `const char * label () const`
- Gets the current label text.*

 - `void label (const char *text)`
- Sets the current label pointer.*

 - `void label (FI_Labeltype a, const char *b)`
- Shortcut to set the label text and type in one call.*

 - `int label_image_spacing ()`
- Return the gap size between the label and the image.*

 - `void label_image_spacing (int gap)`
- Set the gap between the label and the image in pixels.*

 - `FI_Color labelcolor () const`
- Gets the label color.*

 - `void labelcolor (FI_Color c)`
- Sets the label color.*

 - `FI_Font labelfont () const`
- Gets the font to use.*

 - `void labelfont (FI_Font f)`
- Sets the font to use.*

 - `FI_Fonsize labelsiz (int) const`
- Gets the font size in pixels.*

 - `void labelsiz (FI_Fonsize pix)`
- Sets the font size in pixels.*

 - `FI_Labeltype labeltype () const`
- Gets the label type.*

 - `void labeltype (FI_Labeltype a)`
- Sets the label type.*

 - `void measure_label (int &ww, int &hh) const`
- Sets width ww and height hh accordingly with the label size.*

 - `bool needs_keyboard () const`
- Returns whether this widget needs a keyboard.*

 - `void needs_keyboard (bool needs)`
- Sets whether this widget needs a keyboard.*

 - `unsigned int output () const`
- Returns if a widget is used for output only.*

 - `FI_Group * parent () const`
- Returns a pointer to the parent widget.*

 - `void parent (FI_Group *p)`
- Internal use only - "for hacks only".*

- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [Fl_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)

- Sets the user data for this widget.*

 - void **user_data** (void *v)

Sets the user data for this widget.
- int **vertical_label_margin** ()

Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const

Returns whether a widget is visible.
- unsigned int **visible_focus** () const

Checks whether this widget has a visible focus.
- void **visible_focus** (int v)

Modifies keyboard focus navigation.
- int **visible_r** () const

Returns whether a widget and all its parents are visible.
- int **w** () const

Gets the widget width.
- **FL_When** when () const

Returns the conditions under which the callback is called.
- void **when** (uchar i)

Sets the flags used to decide when a callback is called.
- **FL_Window** * **window** () const

Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const

Gets the widget position in its window.
- int **y** () const

Gets the widget position in its window.
- virtual **~FL_Widget** ()

Destroys the widget.

Protected Member Functions

- int **handle** (int) **FL_OVERRIDE**

Handles the event within the normal widget bounding box.
- void **item_draw** (void *, int, int, int, int) const **FL_OVERRIDE**

This method must be provided by the subclass to draw the item in the area indicated by X, Y, W, H.
- void * **item_first** () const **FL_OVERRIDE**

This method must be provided by the subclass to return the first item in the list.
- int **item_height** (void *) const **FL_OVERRIDE**

This method must be provided by the subclass to return the height of item in pixels.
- void * **item_next** (void *) const **FL_OVERRIDE**

This method must be provided by the subclass to return the item in the list after item.
- void * **item_prev** (void *) const **FL_OVERRIDE**

This method must be provided by the subclass to return the item in the list before item.
- void **item_select** (void *, int) **FL_OVERRIDE**

This method must be implemented by the subclass if it supports multiple selections; sets the selection state to val for the item.
- int **item_selected** (void *) const **FL_OVERRIDE**

This method must be implemented by the subclass if it supports multiple selections; returns the selection state for item.
- const char * **item_text** (void *item) const **FL_OVERRIDE**

This optional method returns a string (label) that may be used for sorting.
- int **item_width** (void *) const **FL_OVERRIDE**

This method must be provided by the subclass to return the width of the item in pixels.

Protected Member Functions inherited from FI_Browser_

- void **bbox** (int &X, int &Y, int &W, int &H) const
Returns the bounding box for the interior of the list's display window, inside the scrollbars.
- void **deleting** (void *item)
*This method should be used when *item* is being deleted from the list.*
- int **displayed** (void *item) const
*Returns non-zero if *item* has been scrolled to a position where it is being displayed.*
- void **draw** () **FL_OVERRIDE**
Draws the list within the normal widget bounding box.
- void * **find_item** (int ypos)
*This method returns the item under mouse y position *ypos*.*
- **FI_Browser_** (int X, int Y, int W, int H, const char *L=0)
The constructor makes an empty browser.
- virtual int **full_height** () const
This method may be provided by the subclass to indicate the full height of the item list, in pixels.
- virtual int **full_width** () const
This method may be provided by the subclass to indicate the full width of the item list, in pixels.
- virtual int **incr_height** () const
This method may be provided to return the average height of all items to be used for scrolling.
- void **inserting** (void *a, void *b)
This method should be used when an item is in the process of being inserted into the list.
- virtual void * **item_last** () const
This method must be provided by the subclass to return the last item in the list.
- virtual int **item_quick_height** (void *item) const
*This method may be provided by the subclass to return the height of the *item*, in pixels.*
- int **leftedge** () const
This method returns the X position of the left edge of the list area after adjusting for the scrollbar and border, if any.
- void **new_list** ()
This method should be called when the list data is completely replaced or cleared.
- void **redraw_line** (void *item)
*This method should be called when the contents of *item* has changed, but not its height.*
- void **redraw_lines** ()
This method will cause the entire list to be redrawn.
- void **replacing** (void *a, void *b)
*This method should be used when item *a* is being replaced by item *b*.*
- void * **selection** () const
Returns the item currently selected, or NULL if there is no selection.
- void **swapping** (void *a, void *b)
*This method should be used when two items *a* and *b* are being swapped.*
- void * **top** () const
Returns the item that appears at the top of the list.

Protected Member Functions inherited from FI_Group

- **FI_Rect** * **bounds** ()
Returns the internal array of widget sizes and positions.
- void **draw** () **FL_OVERRIDE**
Draws the widget.
- void **draw_child** (**FI_Widget** &widget) const
Forces a child to redraw.
- void **draw_children** ()

- Draws all children of the group.*
- void **draw_outside_label** (const [FL_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int **on_insert** ([FL_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int **on_move** (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void **on_remove** (int)
Allow derived groups to act when a child widget is removed from the group.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** ([FL_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FL_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FL_Boxtype](#) t, [FL_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FL_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FL_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Public Types inherited from FI_Browser_

- enum {
[HORIZONTAL](#) = 1 , [VERTICAL](#) = 2 , [BOTH](#) = 3 , [ALWAYS_ON](#) = 4 ,
[HORIZONTAL_ALWAYS](#) = 5 , [VERTICAL_ALWAYS](#) = 6 , [BOTH_ALWAYS](#) = 7 }
Values for [has_scrollbar\(\)](#).

Static Public Member Functions inherited from FI_Group

- static [FI_Group](#) * [current](#) ()
Returns the currently active group.
- static void [current](#) ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from FI_Widget

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Public Attributes inherited from FI_Browser_

- [FI_Scrollbar](#) [hscrollbar](#)
Horizontal scrollbar.
- [FI_Scrollbar](#) [scrollbar](#)
Vertical scrollbar.

Protected Types inherited from FI_Widget

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.21.1 Detailed Description

The [FI_Check_Browser](#) widget displays a scrolling list of text lines that may be selected and/or checked by the user.

11.21.2 Member Function Documentation

11.21.2.1 `add()` [1/2]

```
int Fl_Check_Browser::add (
    char * s)
```

Add a new unchecked line to the end of the browser.

See also

[add\(char *s, int b\)](#)

11.21.2.2 `add()` [2/2]

```
int Fl_Check_Browser::add (
    char * s,
    int b)
```

Add a new line to the end of the browser.

The text is copied using the `strdup()` function. It may also be `NULL` to make a blank line. It can set the item checked if `b` is not 0.

11.21.2.3 `handle()`

```
int Fl_Check_Browser::handle (
    int event) [protected], [virtual]
```

Handles the `event` within the normal widget bounding box.

Parameters

in	<i>event</i>	The event to process.

Returns

1 if event was processed, 0 if not.

Reimplemented from [Fl_Browser_](#).

11.21.2.4 `item_at()`

```
void * Fl_Check_Browser::item_at (
    int index) const [virtual]
```

This method must be provided by the subclass to return the item for the specified `index`.

Parameters

in	<i>index</i>	The <code>index</code> of the item to be returned

Returns

The item at the specified `index`.

Reimplemented from [Fl_Browser_](#).

11.21.2.5 `item_draw()`

```
void Fl_Check_Browser::item_draw (
    void * item,
    int X,
    int Y,
    int W,
    int H) const [protected], [virtual]
```

This method must be provided by the subclass to draw the `item` in the area indicated by `X`, `Y`, `W`, `H`.

Implements [Fl_Browser_](#).

11.21.2.6 item_first()

```
void * Fl_Check_Browser::item_first () const [protected], [virtual]
```

This method must be provided by the subclass to return the first item in the list.

See also

[item_first\(\)](#), [item_next\(\)](#), [item_last\(\)](#), [item_prev\(\)](#)

Implements [Fl_Browser_](#).

11.21.2.7 item_height()

```
int Fl_Check_Browser::item_height (
    void * item) const [protected], [virtual]
```

This method must be provided by the subclass to return the height of *item* in pixels. Allow for two additional pixels for the list selection box.

Parameters

<i>in</i>	<i>item</i>	The item whose height is returned.

Returns

The height of the specified *item* in pixels.

See also

[item_height\(\)](#), [item_width\(\)](#), [item_quick_height\(\)](#)

Implements [Fl_Browser_](#).

11.21.2.8 item_next()

```
void * Fl_Check_Browser::item_next (
    void * item) const [protected], [virtual]
```

This method must be provided by the subclass to return the item in the list after *item*.

See also

[item_first\(\)](#), [item_next\(\)](#), [item_last\(\)](#), [item_prev\(\)](#)

Implements [Fl_Browser_](#).

11.21.2.9 item_prev()

```
void * Fl_Check_Browser::item_prev (
    void * item) const [protected], [virtual]
```

This method must be provided by the subclass to return the item in the list before *item*.

See also

[item_first\(\)](#), [item_next\(\)](#), [item_last\(\)](#), [item_prev\(\)](#)

Implements [Fl_Browser_](#).

11.21.2.10 item_select()

```
void Fl_Check_Browser::item_select (
    void * item,
    int val) [protected], [virtual]
```

This method must be implemented by the subclass if it supports multiple selections; sets the selection state to *val* for the *item*.

Sets the selection state for *item*, where optional *val* is 1 (select, the default) or 0 (de-select).

Parameters

in	<i>item</i>	The item to be selected
in	<i>val</i>	The optional selection state; 1=select, 0=de-select. The default is to select the item (1).

Reimplemented from [Fl_Browser_](#).

11.21.2.11 item_selected()

```
int Fl_Check_Browser::item_selected (
    void * item) const [protected], [virtual]
```

This method must be implemented by the subclass if it supports multiple selections; returns the selection state for *item*.

The method should return 1 if *item* is selected, or 0 otherwise.

Parameters

in	<i>item</i>	The item to test.

Reimplemented from [Fl_Browser_](#).

11.21.2.12 item_swap()

```
void Fl_Check_Browser::item_swap (
    void * a,
    void * b) [virtual]
```

This optional method should be provided by the subclass to efficiently swap browser items *a* and *b*, such as for sorting.

Parameters

in	<i>a,b</i>	The two items to be swapped.

Reimplemented from [Fl_Browser_](#).

11.21.2.13 item_text()

```
const char * Fl_Check_Browser::item_text (
    void * item) const [protected], [virtual]
```

This optional method returns a string (label) that may be used for sorting.

Parameters

in	<i>item</i>	The item whose label text is returned.

Returns

The item's text label. (Can be NULL if blank)

Reimplemented from [Fl_Browser_](#).

11.21.2.14 item_width()

```
int Fl_Check_Browser::item_width (
    void * item) const [protected], [virtual]
```

This method must be provided by the subclass to return the width of the *item* in pixels.

Allow for two additional pixels for the list selection box.

Parameters

<code>in</code>	<code>item</code>	The item whose width is returned.

Returns

The width of the item in pixels.

Implements [Fl_Browser_](#).

11.21.2.15 nitems()

```
int Fl_Check_Browser::nitems () const [inline]
```

Returns how many lines are in the browser.

The last line number is equal to this.

11.21.2.16 remove()

```
int Fl_Check_Browser::remove (
    int item)
```

Remove line n and make the browser one line shorter.

Returns the number of lines left in the browser.

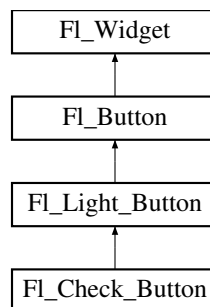
The documentation for this class was generated from the following files:

- [Fl_Check_Browser.H](#)
- [Fl_Check_Browser.cxx](#)

11.22 Fl_Check_Button Class Reference

A button with a "checkmark" to show its status.

Inheritance diagram for [Fl_Check_Button](#):

**Public Member Functions**

- [Fl_Check_Button](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Check_Button](#) widget using the given position, size, and label string.

Public Member Functions inherited from [Fl_Light_Button](#)

- [Fl_Light_Button](#) (int x, int y, int w, int h, const char *l=0)
Creates a new [Fl_Light_Button](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [Fl_Button](#)

- `int clear ()`
Same as `value(0)`.
- `uchar compact ()`
Return true if buttons are rendered as compact buttons.
- `void compact (uchar v)`
Decide if buttons should be rendered in compact mode.
- `Fl_Boxtype down_box () const`
Returns the current down box type, which is drawn when `value()` is non-zero.
- `void down_box (Fl_Boxtype b)`
Sets the down box type.
- `Fl_Color down_color () const`
(for backwards compatibility)
- `void down_color (unsigned c)`
(for backwards compatibility)
- `Fl_Button (int X, int Y, int W, int H, const char *L=0)`
The constructor creates the button using the given position, size, and label.
- `int set ()`
Same as `value(1)`.
- `void setonly ()`
Turns on this button and turns off all other radio buttons in the group (calling `value(1)` or `set()` does not do this).
- `int shortcut () const`
Returns the current shortcut key for the button.
- `void shortcut (const char *s)`
(for backwards compatibility)
- `void shortcut (int s)`
Sets the shortcut key to `s`.
- `char value () const`
Returns the current value of the button (0 or 1).
- `int value (int v)`
Sets the current value of the button.

Public Member Functions inherited from [Fl_Widget](#)

- `void _clear_fullscreen ()`
- `void _set_fullscreen ()`
- `void activate ()`
Activates the widget.
- `unsigned int active () const`
Returns whether the widget is active.
- `int active_r () const`
Returns whether the widget and all of its parents are active.
- `Fl_Align align () const`
Gets the label alignment.
- `void align (Fl_Align alignment)`
Sets the label alignment.
- `long argument () const`
Gets the current user data (long) argument that is passed to the callback function.
- `void argument (long v)`
Sets the current user data (long) argument that is passed to the callback function.
- virtual class `Fl_Gl_Window * as_gl_window ()`

- Returns an [FI_GI_Window](#) pointer if this widget is an [FI_GI_Window](#).*
- virtual class [FI_GI_Window](#) const * **as_gi_window** () const
- virtual [FI_Group](#) * **as_group** ()
- Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).*
- virtual [FI_Group](#) const * **as_group** () const
- virtual [FI_Window](#) * **as_window** ()
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*
- virtual [FI_Window](#) const * **as_window** () const
- void **bind_deimage** ([FI_Image](#) *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void **bind_deimage** (int f)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- void **bind_image** ([FI_Image](#) *img)
- Sets the image to use as part of the widget label when in the active state.*
- void **bind_image** (int f)
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- [FI_Boxtype](#) **box** () const
- Gets the box type of the widget.*
- void **box** ([FI_Boxtype](#) new_box)
- Sets the box type for the widget.*
- [FI_Callback_p](#) **callback** () const
- Gets the current callback function for the widget.*
- void **callback** ([FI_Callback](#) *cb)
- Sets the current callback function for the widget.*
- void **callback** ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*
- void **callback** ([FI_Callback](#) *cb, void *p)
- Sets the current callback function and data for the widget.*
- void **callback** ([FI_Callback0](#) *cb)
- Sets the current callback function for the widget.*
- void **callback** ([FI_Callback1](#) *cb, long p=0)
- Sets the current callback function for the widget.*
- unsigned int **changed** () const
- Checks if the widget value changed since the last callback.*
- void **clear_active** ()
- Marks the widget as inactive without sending events or changing focus.*
- void **clear_changed** ()
- Marks the value of the widget as unchanged.*
- void **clear_damage** (uchar c=0)
- Clears or sets the damage flags.*
- void **clear_output** ()
- Sets a widget to accept input.*
- void **clear_visible** ()
- Hides the widget.*
- void **clear_visible_focus** ()
- Disables keyboard focus navigation with this widget.*
- [FI_Color](#) **color** () const
- Gets the background color of the widget.*
- void **color** ([FI_Color](#) bg)
- Sets the background color of the widget.*
- void **color** ([FI_Color](#) bg, [FI_Color](#) sel)

- Sets the background and selection color of the widget.*

 - `FL_Color color2 () const`

For back compatibility only.
- `void color2 (unsigned a)`

For back compatibility only.
- `int contains (const FL_Widget *w) const`

Checks if w is a child of this widget.
- `void copy_label (const char *new_label)`

Sets the current label.
- `void copy_tooltip (const char *text)`

Sets the current tooltip text.
- `uchar damage () const`

Returns non-zero if `draw()` needs to be called.
- `void damage (uchar c)`

Sets the damage bits for the widget.
- `void damage (uchar c, int x, int y, int w, int h)`

Sets the damage bits for an area inside the widget.
- `int damage_resize (int, int, int, int)`

Internal use only.
- `void deactivate ()`

Deactivates the widget.
- `FL_Image * deimage ()`

Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`

Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`

Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`

Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`

Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FL_Align) const`

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`

Gets the widget height.
- `virtual void hide ()`

Makes a widget invisible.
- `int horizontal_label_margin ()`

Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`

Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`

Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`

Gets the image that is used as part of the widget label when in the active state.

- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()

- Schedules the drawing of the widget.*

 - void `redraw_label` ()

Schedules the drawing of the label.
- virtual void `resize` (int `x`, int `y`, int `w`, int `h`)

Changes the size or position of the widget.
- `Fl_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`Fl_Color` `a`)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.
- void `set_visible` ()

Makes the widget visible.
- void `set_visible_focus` ()

Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int `value`)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()

Makes a widget visible.
- void `size` (int `W`, int `H`)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const

Gets the current tooltip text.
- void `tooltip` (const char *`text`)

Sets the current tooltip text.
- `Fl_Window` * `top_window` () const

Returns a pointer to the top-level window for the widget.
- `Fl_Window` * `top_window_offset` (int &`xoff`, int &`yoff`) const

Finds the x/y offset of the current widget relative to the top-level window.
- `uchar` `type` () const

Gets the widget type.
- void `type` (`uchar` `t`)

Sets the widget type.
- int `use_accents_menu` ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * `user_data` () const

Gets the user data for this widget.
- void `user_data` (`Fl_Callback_User_Data` *`v`, bool `auto_free`)

Sets the user data for this widget.

- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **Fl_When** when () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~Fl_Widget** ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from **Fl_Widget**

- static void **default_callback** (**Fl_Widget** *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from **Fl_Widget**

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,

`MAXIMIZED = 1<<24 , POPUP = 1<<25 , USERFLAG3 = 1<<29 , USERFLAG2 = 1<<30 ,
USERFLAG1 = 1<<31 }`

flags possible values enumeration.

Protected Member Functions inherited from `Fl_Light_Button`

- void `draw ()` `FL_OVERRIDE`

Draws the widget.

Protected Member Functions inherited from `Fl_Button`

- void `simulate_key_action ()`

Protected Member Functions inherited from `Fl_Widget`

- void `clear_flag` (unsigned int c)
Clears a flag in the flags mask.
- void `draw_backdrop ()` const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void `draw_box ()` const
Draws the widget box according its box style.
- void `draw_box` (`Fl_Boxtype` t, `Fl_Color` c) const
Draws a box of type t, of color c at the widget's position and size.
- void `draw_box` (`Fl_Boxtype` t, int x, int y, int w, int h, `Fl_Color` c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void `draw_focus ()` const
Draws a focus rectangle around the widget.
- void `draw_focus` (`Fl_Boxtype` t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void `draw_focus` (`Fl_Boxtype` t, int x, int y, int w, int h, `Fl_Color` bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label ()` const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `Fl_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags ()` const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Static Protected Member Functions inherited from `Fl_Button`

- static void `key_release_timeout` (void *)

Static Protected Attributes inherited from Fl_Button

- static `Fl_Widget_Tracker * key_release_tracker = 0`

11.22.1 Detailed Description

A button with a "checkmark" to show its status.

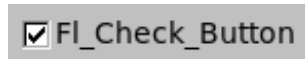


Figure 11.7 Fl_Check_Button

Buttons generate callbacks when they are clicked by the user. You control exactly when and how by changing the values for `type()` and `when()`.

The `Fl_Check_Button` subclass displays its "ON" state by showing a "checkmark" rather than drawing itself pushed in.

11.22.2 Constructor & Destructor Documentation

11.22.2.1 Fl_Check_Button()

```
Fl_Check_Button::Fl_Check_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new `Fl_Check_Button` widget using the given position, size, and label string.

The default box type is `FL_NO_BOX`, which draws the label w/o a box right of the checkmark.

The `selection_color()` sets the color of the checkmark. Default is `FL_FOREGROUND_COLOR` (usually black).

You can use `down_box()` to change the box type of the checkmark. Default is `FL_DOWN_BOX`.

Parameters

in	<code>X,Y,W,H</code>	position and size of the widget
in	<code>L</code>	widget label, default is no label

The documentation for this class was generated from the following files:

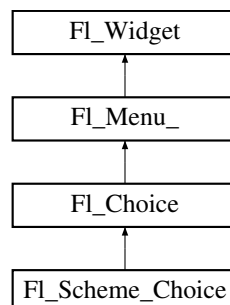
- `Fl_Check_Button.H`
- `Fl_Check_Button.cxx`

11.23 Fl_Choice Class Reference

A button that is used to pop up a menu.

```
#include <Fl_Choice.H>
```

Inheritance diagram for `Fl_Choice`:



Public Member Functions

- [FL_Choice](#) (int X, int Y, int W, int H, const char *L=0)
Create a new [FL_Choice](#) widget using the given position, size and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- int [value](#) () const
Gets the index of the last item chosen by the user.
- int [value](#) (const [FL_Menu_Item](#) *v)
Sets the currently selected value using a pointer to menu item.
- int [value](#) (int v)
Sets the currently selected value using the index into the menu item array.

Public Member Functions inherited from [FL_Menu_](#)

- int [add](#) (const char *)
This is a Forms (and SGI GL library) compatible add function, it adds many menu items, with '|' separating the menu items, and tab separating the menu item names from an optional shortcut string.
- int [add](#) (const char *, int [shortcut](#), [FL_Callback](#) *, void *=0, int=0)
Adds a new menu item.
- int [add](#) (const char *a, const char *b, [FL_Callback](#) *c, void *d=0, int e=0)
See int [FL_Menu_::add](#)(const char label, int shortcut, [FL_Callback](#)*, void *user_data=0, int flags=0)*
- void [clear](#) ()
Same as menu(NULL), set the array pointer to null, indicating a zero-length menu.
- int [clear_submenu](#) (int index)
*Clears the specified submenu pointed to by *index* of all menu items.*
- void [copy](#) (const [FL_Menu_Item](#) *m, void *user_data=0)
Sets the menu array pointer with a copy of m that will be automatically deleted.
- [FL_Boxtype](#) [down_box](#) () const
This box type is used to surround the currently-selected items in the menus.
- void [down_box](#) ([FL_Boxtype](#) b)
Sets the box type used to surround the currently-selected items in the menus.
- [FL_Color](#) [down_color](#) () const
For back compatibility, same as [selection_color\(\)](#)
- void [down_color](#) (unsigned c)
For back compatibility, same as [selection_color\(\)](#)
- int [find_index](#) (const char *name) const
*Find the menu item index for a given menu *pathname*, such as "Edit/Copy".*
- int [find_index](#) (const [FL_Menu_Item](#) *item) const
*Find the index into the menu array for a given *item*.*
- int [find_index](#) ([FL_Callback](#) *cb) const
*Find the index into the menu array for a given callback *cb*.*
- const [FL_Menu_Item](#) * [find_item](#) (const char *name)
*Find the menu item for a given menu *pathname*, such as "Edit/Copy".*
- const [FL_Menu_Item](#) * [find_item](#) ([FL_Callback](#) *)
*Find the menu item for the given callback *cb*.*
- const [FL_Menu_Item](#) * [find_item_with_argument](#) (long)
*Find the menu item for the given user argument *v*.*
- const [FL_Menu_Item](#) * [find_item_with_user_data](#) (void *)
*Find the menu item for the given user data *v*.*
- [FL_Menu_](#) (int, int, int, int, const char *=0)
Creates a new [FL_Menu_](#) widget using the given position, size, and label string.

- void **global** ()
Make the shortcuts for this menu work no matter what window has the focus when you type it.
- int **insert** (int index, const char *, int shortcut, FL_Callback *, void *=0, int=0)
Inserts a new menu item at the specified `index` position.
- int **insert** (int index, const char *a, const char *b, FL_Callback *c, void *d=0, int e=0)
See int `FL_Menu::insert`(const char label, int shortcut, FL_Callback*, void *user_data=0, int flags=0)*
- int **item_pathname** (char *name, int namelen, const FL_Menu_Item *finditem=0) const
Get the menu 'pathname' for the specified menuitem.
- const FL_Menu_Item * **menu** () const
Returns a pointer to the array of FL_Menu_Items.
- void **menu** (const FL_Menu_Item *m)
Sets the menu array pointer directly.
- FL_Boxtype **menu_box** () const
Get the box type for the menu popup windows.
- void **menu_box** (FL_Boxtype b)
Set the box type for the menu popup windows.
- const FL_Menu_Item * **menu_end** ()
Finishes menu modifications and returns `menu()`.
- int **mode** (int i) const
Get the flags of item `i`.
- void **mode** (int i, int fl)
Set the flags of item `i`.
- const FL_Menu_Item * **mvalue** () const
Return a pointer to the last menu item that was picked.
- const FL_Menu_Item * **picked** (const FL_Menu_Item *)
When user picks a menu item, call this.
- const FL_Menu_Item * **prev_mvalue** () const
Return a pointer to the menu item that was picked before the current one was picked.
- void **remove** (int)
Deletes item `i` from the menu.
- void **replace** (int, const char *)
Changes the text of item `i`.
- void **setonly** (FL_Menu_Item *item)
Turns the radio item "on" for the menu item and turns "off" adjacent radio items of the same group.
- void **shortcut** (int i, int s)
Change the shortcut of item `i` to `s`.
- int **size** () const
This returns the number of FL_Menu_Item structures that make up the menu, correctly counting submenus.
- void **size** (int W, int H)
- const FL_Menu_Item * **test_shortcut** ()
Returns the menu item with the entered shortcut (key value).
- const char * **text** () const
Returns the title of the last item chosen.
- const char * **text** (int i) const
Returns the title of item `i`.
- FL_Color **textcolor** () const
Get the current color of menu item labels.
- void **textcolor** (FL_Color c)
Sets the current color of menu item labels.
- FL_Font **textfont** () const
Gets the current font of menu item labels.

- void `textfont` (`FL_Font` c)
Sets the current font of menu item labels.
- `FL_Fontsize` `textsize` () const
Gets the font size of menu item labels.
- void `textsize` (`FL_Fontsize` c)
Sets the font size of menu item labels.
- int `value` () const
Return the index into the `menu()` of the last item chosen by the user.
- int `value` (const `FL_Menu_Item` *)
Set the value of a menu to the menu item `m`.
- int `value` (int i)
Set the value of the menu to index `i`.

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
Activates the widget.
- unsigned int `active` () const
Returns whether the widget is active.
- int `active_r` () const
Returns whether the widget and all of its parents are active.
- `FL_Align` `align` () const
Gets the label alignment.
- void `align` (`FL_Align` alignment)
Sets the label alignment.
- long `argument` () const
Gets the current user data (long) argument that is passed to the callback function.
- void `argument` (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class `FL_Gl_Window` * `as_gl_window` ()
Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.
- virtual class `FL_Gl_Window` const * `as_gl_window` () const
- virtual `FL_Group` * `as_group` ()
Returns an `FL_Group` pointer if this widget is an `FL_Group`.
- virtual `FL_Group` const * `as_group` () const
- virtual `FL_Window` * `as_window` ()
Returns an `FL_Window` pointer if this widget is an `FL_Window`.
- virtual `FL_Window` const * `as_window` () const
- void `bind_deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- void `bind_deimage` (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void `bind_image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- void `bind_image` (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- `FL_Boxtype` `box` () const
Gets the box type of the widget.
- void `box` (`FL_Boxtype` new_box)

- Sets the box type for the widget.*

 - `FI_Callback_p callback () const`
- Gets the current callback function for the widget.*

 - `void callback (FI_Callback *cb)`
- Sets the current callback function for the widget.*

 - `void callback (FI_Callback *cb, FI_Callback_User_Data *p, bool auto_free)`
- Sets the current callback function and managed user data for the widget.*

 - `void callback (FI_Callback *cb, void *p)`
- Sets the current callback function and data for the widget.*

 - `void callback (FI_Callback0 *cb)`
- Sets the current callback function for the widget.*

 - `void callback (FI_Callback1 *cb, long p=0)`
- Sets the current callback function for the widget.*

 - `unsigned int changed () const`
- Checks if the widget value changed since the last callback.*

 - `void clear_active ()`
- Marks the widget as inactive without sending events or changing focus.*

 - `void clear_changed ()`
- Marks the value of the widget as unchanged.*

 - `void clear_damage (uchar c=0)`
- Clears or sets the damage flags.*

 - `void clear_output ()`
- Sets a widget to accept input.*

 - `void clear_visible ()`
- Hides the widget.*

 - `void clear_visible_focus ()`
- Disables keyboard focus navigation with this widget.*

 - `FI_Color color () const`
- Gets the background color of the widget.*

 - `void color (FI_Color bg)`
- Sets the background color of the widget.*

 - `void color (FI_Color bg, FI_Color sel)`
- Sets the background and selection color of the widget.*

 - `FI_Color color2 () const`
- For back compatibility only.*

 - `void color2 (unsigned a)`
- For back compatibility only.*

 - `int contains (const FI_Widget *w) const`
- Checks if w is a child of this widget.*

 - `void copy_label (const char *new_label)`
- Sets the current label.*

 - `void copy_tooltip (const char *text)`
- Sets the current tooltip text.*

 - `uchar damage () const`
- Returns non-zero if `draw()` needs to be called.*

 - `void damage (uchar c)`
- Sets the damage bits for the widget.*

 - `void damage (uchar c, int x, int y, int w, int h)`
- Sets the damage bits for an area inside the widget.*

 - `int damage_resize (int, int, int, int)`
- Internal use only.*

- void `deactivate` ()
Deactivates the widget.
- `FL_Image * deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image &img`)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image *img`)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget *widget`, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget *widget`, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image * image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image &img`)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image *img`)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget *wgt`) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

- Set the gap between the label and the image in pixels.*

 - [FI_Color labelcolor](#) () const
 - Gets the label color.*
 - void [labelcolor](#) ([FI_Color](#) c)
 - Sets the label color.*
 - [FI_Font labelfont](#) () const
 - Gets the font to use.*
 - void [labelfont](#) ([FI_Font](#) f)
 - Sets the font to use.*
 - [FI_Fonsize labelsiz](#)e () const
 - Gets the font size in pixels.*
 - void [labelsiz](#)e ([FI_Fonsize](#) pix)
 - Sets the font size in pixels.*
 - [FI_Labeltype labeltyp](#)e () const
 - Gets the label type.*
 - void [labeltyp](#)e ([FI_Labeltype](#) a)
 - Sets the label type.*
 - void [measure_label](#) (int &ww, int &hh) const
 - Sets width ww and height hh accordingly with the label size.*
 - bool [needs_keyboard](#) () const
 - Returns whether this widget needs a keyboard.*
 - void [needs_keyboard](#) (bool needs)
 - Sets whether this widget needs a keyboard.*
 - unsigned int [output](#) () const
 - Returns if a widget is used for output only.*
 - [FI_Group * parent](#) () const
 - Returns a pointer to the parent widget.*
 - void [parent](#) ([FI_Group](#) *p)
 - Internal use only - "for hacks only".*
 - void [position](#) (int X, int Y)
 - Repositions the window or widget.*
 - void [redraw](#) ()
 - Schedules the drawing of the widget.*
 - void [redraw_label](#) ()
 - Schedules the drawing of the label.*
 - virtual void [resize](#) (int x, int y, int w, int h)
 - Changes the size or position of the widget.*
 - [FI_Color selection_color](#) () const
 - Gets the selection color.*
 - void [selection_color](#) ([FI_Color](#) a)
 - Sets the selection color.*
 - void [set_active](#) ()
 - Marks the widget as active without sending events or changing focus.*
 - void [set_changed](#) ()
 - Marks the value of the widget as changed.*
 - void [set_output](#) ()
 - Sets a widget to output only.*
 - void [set_visible](#) ()
 - Makes the widget visible.*
 - void [set_visible_focus](#) ()
 - Enables keyboard focus navigation with this widget.*

- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)

Sets the flags used to decide when a callback is called.

- `FL_Window * window () const`

Returns a pointer to the nearest parent window up the widget hierarchy.

- `int x () const`

Gets the widget position in its window.

- `int y () const`

Gets the widget position in its window.

- `virtual ~FL_Widget ()`

Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`

Draws the widget.

Protected Member Functions inherited from `FL_Menu_`

- `int item_pathname_ (char *name, int namelen, const FL_Menu_Item *finditem, const FL_Menu_Item *menu=0) const`

Protected Member Functions inherited from `FL_Widget`

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const`
Draws a focus box for the widget at the given position and size.
- `void draw_label () const`
Draws the widget's label at the defined label position.
- `void draw_label (int, int, int, int) const`
Draws the label in an arbitrary bounding box.
- `FL_Widget (int x, int y, int w, int h, const char *label=0L)`
Creates a widget at the given position and size.
- `unsigned int flags () const`
Gets the widget flags mask.
- `void h (int v)`
Internal use only.
- `void set_flag (unsigned int c)`
Sets a flag in the flags mask.
- `void w (int v)`
Internal use only.

- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Attributes inherited from [Fl_Menu_](#)

- [uchar](#) **alloc**
- [uchar](#) **down_box_**
- [Fl_Boxtype](#) **menu_box_**
- [Fl_Color](#) **textcolor_**
- [Fl_Font](#) **textfont_**
- [Fl_Fonsize](#) **textsize_**

11.23.1 Detailed Description

A button that is used to pop up a menu.

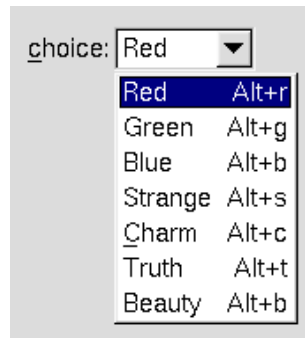


Figure 11.8 Fl_Choice

This is a button that, when pushed, pops up a menu (or hierarchy of menus) defined by an array of [Fl_Menu_Item](#) objects. Motif calls this an `OptionButton`.

The only difference between this and a [Fl_Menu_Button](#) is that the name of the most recent chosen menu item is displayed inside the box, while the label is displayed outside the box. However, since the use of this is most often to control a single variable rather than do individual callbacks, some of the [Fl_Menu_Button](#) methods are redescribed here in those terms.

When the user clicks a menu item, `value()` is set to that item and then:

- The item's callback is done if one has been set; the `Fl_Choice` is passed as the `Fl_Widget*` argument, along with any userdata configured for the callback.
- If the item does not have a callback, the `Fl_Choice` widget's callback is done instead, along with any userdata configured for it. The callback can determine which item was picked using `value()`, `mvalue()`, `item_pathname()`, etc.

All three mouse buttons pop up the menu. The Forms behavior of the first two buttons to increment/decrement the choice is not implemented. This could be added with a subclass, however.

The menu will also pop up in response to shortcuts indicated by putting a `'&'` character in the `label()`. See [Fl_Button::shortcut\(int s\)](#) for a description of this.

Typing the `shortcut()` of any of the items will do exactly the same as when you pick the item with the mouse. The `'&'` character in item names are only looked at when the menu is popped up, however.

The inherited [Fl_Widget::changed\(\)](#) and related methods can be used as follows:

- `int Fl_Widget::changed() const` This value is true when the user picks a different value. *It is turned off by `value()` and just before doing a callback (the callback can turn it back on if desired).*
- `void Fl_Widget::set_changed()` This method sets the `changed()` flag.
- `void Fl_Widget::clear_changed()` This method clears the `changed()` flag.

The inherited [Fl_Menu::down_box\(\)](#) methods can be used as follows:

- `Fl_Boxtype Fl_Menu::down_box() const` Gets the current down box, which is used when the menu is popped up. The default down box type is `FL_DOWN_BOX`.
- `void Fl_Menu::down_box(Fl_Boxtype b)` Sets the current down box type to `b`.

Simple example:

```
#include <FL/Fl.H>
#include <FL/Fl_Window.H>
#include <FL/Fl_Choice.H>
int main() {
    Fl_Window *win = new Fl_Window(300,200);
    Fl_Choice *choice = new Fl_Choice(100,10,100,25,"Choice:");
    choice->add("Zero");
    choice->add("One");
    choice->add("Two");
    choice->add("Three");
    choice->value(2); // make "Two" selected by default (zero based!)
    win->end();
    win->show();
    return Fl::run();
}
```

11.23.2 Constructor & Destructor Documentation

11.23.2.1 Fl_Choice()

```
Fl_Choice::Fl_Choice (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Create a new [Fl_Choice](#) widget using the given position, size and label string.

The default boxtype is `FL_UP_BOX`.

The constructor sets [menu\(\)](#) to NULL. See [Fl_Menu_](#) for the methods to set or change the menu.

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

11.23.3 Member Function Documentation

11.23.3.1 draw()

```
void Fl_Choice::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.23.3.2 handle()

```
int Fl_Choice::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

<i>0</i>	if the event was not used or understood

1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Scheme_Choice](#).

11.23.3.3 value() [1/3]

```
int Fl_Choice::value () const [inline]
```

Gets the index of the last item chosen by the user.
The index is -1 initially.

11.23.3.4 value() [2/3]

```
int Fl_Choice::value (
    const Fl_Menu_Item * v)
```

Sets the currently selected value using a pointer to menu item.
Changing the selected value causes a [redraw\(\)](#).

Parameters

in	v	pointer to menu item in the menu item array.

Returns

non-zero if the new value is different to the old one.

11.23.3.5 value() [3/3]

```
int Fl_Choice::value (
    int v)
```

Sets the currently selected value using the index into the menu item array.
Changing the selected value causes a [redraw\(\)](#).

Parameters

in	v	index of value in the menu item array.

Returns

non-zero if the new value is different to the old one.

The documentation for this class was generated from the following files:

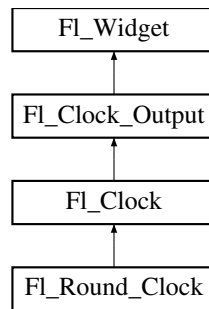
- [Fl_Choice.H](#)
- [Fl_Choice.cxx](#)

11.24 Fl_Clock Class Reference

This widget provides a round analog clock display.

```
#include <Fl_Clock.H>
```

Inheritance diagram for Fl_Clock:



Public Member Functions

- [Fl_Clock](#) (int X, int Y, int W, int H, const char *L=0)
Create an [Fl_Clock](#) widget using the given position, size, and label string.
- [Fl_Clock](#) (uchar t, int X, int Y, int W, int H, const char *L)
Create an [Fl_Clock](#) widget using the given clock type t, position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- ~[Fl_Clock](#) ()
The destructor removes the clock.

Public Member Functions inherited from [Fl_Clock_Output](#)

- [Fl_Clock_Output](#) (int X, int Y, int W, int H, const char *L=0)
Create a new [Fl_Clock_Output](#) widget with the given position, size and label.
- int [hour](#) () const
Returns the displayed hour (0 to 23).
- int [minute](#) () const
Returns the displayed minute (0 to 59).
- int [second](#) () const
Returns the displayed second (0 to 60, 60=leap second).
- int [shadow](#) () const
Returns the shadow drawing mode of the hands.
- void [shadow](#) (int mode)
Sets the shadow drawing mode of the hands.
- [ulong value](#) () const
Returns the displayed time.
- void [value](#) (int H, int m, int s)
Set the displayed time.
- void [value](#) (ulong v)
Set the displayed time.

Public Member Functions inherited from [Fl_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.

- [FL_Align align](#) () const
Gets the label alignment.
- void [align](#) ([FL_Align alignment](#))
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FL_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).
- virtual class [FL_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FL_Group](#) * [as_group](#) ()
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- virtual [FL_Group](#) const * [as_group](#) () const
- virtual [FL_Window](#) * [as_window](#) ()
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).
- virtual [FL_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FL_Boxtype box](#) () const
Gets the box type of the widget.
- void [box](#) ([FL_Boxtype new_box](#))
Sets the box type for the widget.
- [FL_Callback_p callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FL_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FL_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.

- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FL_Color` bg)
Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)
Sets the background and selection color of the widget.
- `FL_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FL_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FL_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()

- Makes a widget invisible.*

 - int [horizontal_label_margin](#) ()

Get the spacing between the label and the horizontal edge of the widget.

- void [horizontal_label_margin](#) (int px)

Set the spacing between the label and the horizontal edge of the widget.

- [FL_Image](#) * [image](#) ()

Gets the image that is used as part of the widget label when in the active state.

- const [FL_Image](#) * [image](#) () const

Gets the image that is used as part of the widget label when in the active state.

- void [image](#) ([FL_Image](#) &img)

Sets the image to use as part of the widget label when in the active state.

- void [image](#) ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.

- int [image_bound](#) () const

Returns whether the image is managed by the widget.

- int [inside](#) (const [FL_Widget](#) *wgt) const

Checks if this widget is a child of wgt.

- int [is_label_copied](#) () const

Returns whether the current label was assigned with [copy_label\(\)](#).

- const char * [label](#) () const

Gets the current label text.

- void [label](#) (const char *text)

Sets the current label pointer.

- void [label](#) ([FL_Labeltype](#) a, const char *b)

Shortcut to set the label text and type in one call.

- int [label_image_spacing](#) ()

Return the gap size between the label and the image.

- void [label_image_spacing](#) (int gap)

Set the gap between the label and the image in pixels.

- [FL_Color](#) [labelcolor](#) () const

Gets the label color.

- void [labelcolor](#) ([FL_Color](#) c)

Sets the label color.

- [FL_Font](#) [labelfont](#) () const

Gets the font to use.

- void [labelfont](#) ([FL_Font](#) f)

Sets the font to use.

- [FL_Fonsize](#) [labelsize](#) () const

Gets the font size in pixels.

- void [labelsize](#) ([FL_Fonsize](#) pix)

Sets the font size in pixels.

- [FL_Labeltype](#) [labeltype](#) () const

Gets the label type.

- void [labeltype](#) ([FL_Labeltype](#) a)

Sets the label type.

- void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.

- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.

- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.

- unsigned int `output` () const
Returns if a widget is used for output only.
- `Fl_Group * parent` () const
Returns a pointer to the parent widget.
- void `parent` (`Fl_Group *p`)
Internal use only - "for hacks only".
- void `position` (int X, int Y)
Repositions the window or widget.
- void `redraw` ()
Schedules the drawing of the widget.
- void `redraw_label` ()
Schedules the drawing of the label.
- virtual void `resize` (int x, int y, int w, int h)
Changes the size or position of the widget.
- `Fl_Color selection_color` () const
Gets the selection color.
- void `selection_color` (`Fl_Color a`)
Sets the selection color.
- void `set_active` ()
Marks the widget as active without sending events or changing focus.
- void `set_changed` ()
Marks the value of the widget as changed.
- void `set_output` ()
Sets a widget to output only.
- void `set_visible` ()
Makes the widget visible.
- void `set_visible_focus` ()
Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()
Makes a widget visible.
- void `size` (int W, int H)
Changes the size of the widget.
- int `take_focus` ()
Gives the widget the keyboard focus.
- unsigned int `takeevents` () const
Returns if the widget is able to take events.
- int `test_shortcut` ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const
Gets the current tooltip text.
- void `tooltip` (const char *text)
Sets the current tooltip text.
- `Fl_Window * top_window` () const
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset` (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- uchar `type` () const

- Gets the widget type.*
- void **type** (uchar t)
 - Sets the widget type.*
- int **use_accents_menu** ()
 - Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*
- void * **user_data** () const
 - Gets the user data for this widget.*
- void **user_data** (FL_Callback_User_Data *v, bool auto_free)
 - Sets the user data for this widget.*
- void **user_data** (void *v)
 - Sets the user data for this widget.*
- int **vertical_label_margin** ()
 - Get the spacing between the label and the vertical edge of the widget.*
- void **vertical_label_margin** (int px)
 - Set the spacing between the label and the vertical edge of the widget.*
- unsigned int **visible** () const
 - Returns whether a widget is visible.*
- unsigned int **visible_focus** () const
 - Checks whether this widget has a visible focus.*
- void **visible_focus** (int v)
 - Modifies keyboard focus navigation.*
- int **visible_r** () const
 - Returns whether a widget and all its parents are visible.*
- int **w** () const
 - Gets the widget width.*
- **FL_When** when () const
 - Returns the conditions under which the callback is called.*
- void **when** (uchar i)
 - Sets the flags used to decide when a callback is called.*
- **FL_Window** * **window** () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
- int **x** () const
 - Gets the widget position in its window.*
- int **y** () const
 - Gets the widget position in its window.*
- virtual ~**FL_Widget** ()
 - Destroys the widget.*

Additional Inherited Members

Static Public Member Functions inherited from FL_Widget

- static void **default_callback** (FL_Widget *widget, void *data)
 - The default callback for all widgets that don't set a callback.*
- static unsigned int **label_shortcut** (const char *t)
 - Returns the Unicode value of the '&x' shortcut in a given text.*
- static int **test_shortcut** (const char *, const bool require_alt=false)
 - Returns true if the given text t contains the entered '&x' shortcut.*

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from [FI_Clock_Output](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draw clock with current position and size.
- void [draw](#) (int X, int Y, int W, int H)
Draw clock with the given position and size.

Protected Member Functions inherited from [FI_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.
- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int [flags](#) () const
Gets the widget flags mask.
- void [h](#) (int v)
Internal use only.
- void [set_flag](#) (unsigned int c)

- Sets a flag in the flags mask.*
- void [w](#) (int v)
Internal use only.
- void [x](#) (int v)
Internal use only.
- void [y](#) (int v)
Internal use only.

11.24.1 Detailed Description

This widget provides a round analog clock display.

[FL_Clock](#) is provided for Forms compatibility. It installs a 1-second timeout callback using [Fl::add_timeout\(\)](#). You can choose the rounded or square type of the clock with [type\(\)](#). Please see [FL_Clock_Output](#) widget for applicable values.

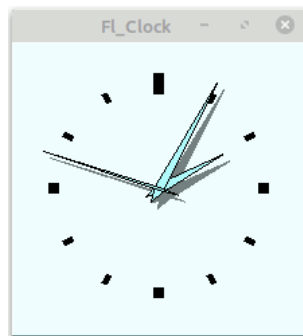


Figure 11.9 FL_SQUARE_CLOCK type

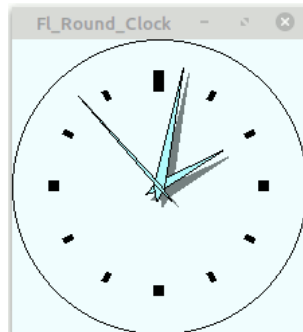


Figure 11.10 FL_ROUND_CLOCK type

See also

class [FL_Clock_Output](#)

11.24.2 Constructor & Destructor Documentation

11.24.2.1 FL_Clock() [1/2]

```
Fl_Clock::Fl_Clock (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Create an [FL_Clock](#) widget using the given position, size, and label string.

The default clock type is FL_SQUARE_CLOCK and the default boxtype is FL_UP_BOX.

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

11.24.2.2 FL_Clock() [2/2]

```

Fl_Clock::Fl_Clock (
    uchar t,
    int X,
    int Y,
    int W,
    int H,
    const char * L)

```

Create an [FL_Clock](#) widget using the given clock type *t*, position, size, and label string.

The default clock type *t* is `FL_SQUARE_CLOCK`. You can set the clock type to `FL_ROUND_CLOCK` or any other valid clock type. See [FL_Clock_Output](#) widget for applicable values.

The default boxtype is `FL_UP_BOX` for `FL_SQUARE_CLOCK` and `FL_NO_BOX` for `FL_ROUND_CLOCK`, if set by the constructor. If you change the clock type with [type\(\)](#) later you should also set the boxtype with [box\(\)](#).

Parameters

in	<i>t</i>	type of clock: <code>FL_ROUND_CLOCK</code> or <code>FL_SQUARE_CLOCK</code> (0)
in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

See also

class [FL_Clock_Output](#)

11.24.3 Member Function Documentation**11.24.3.1 handle()**

```

int Fl_Clock::handle (
    int event) [virtual]

```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood

1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

The documentation for this class was generated from the following files:

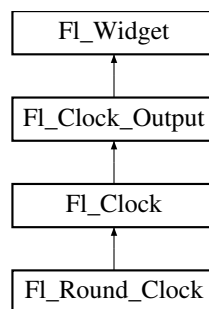
- [Fl_Clock.H](#)
- [Fl_Clock.cxx](#)

11.25 Fl_Clock_Output Class Reference

This widget can be used to display a program-supplied time.

```
#include <Fl_Clock.H>
```

Inheritance diagram for [Fl_Clock_Output](#):



Public Member Functions

- [Fl_Clock_Output](#) (int X, int Y, int W, int H, const char *L=0)
Create a new [Fl_Clock_Output](#) widget with the given position, size and label.
- int [hour](#) () const
Returns the displayed hour (0 to 23).
- int [minute](#) () const
Returns the displayed minute (0 to 59).
- int [second](#) () const
Returns the displayed second (0 to 60, 60=leap second).
- int [shadow](#) () const
Returns the shadow drawing mode of the hands.
- void [shadow](#) (int mode)
Sets the shadow drawing mode of the hands.
- [ulong value](#) () const
Returns the displayed time.
- void [value](#) (int H, int m, int s)
Set the displayed time.
- void [value](#) ([ulong](#) v)
Set the displayed time.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color color` () const

Gets the background color of the widget.
- void `color` (FL_Color bg)

Sets the background color of the widget.
- void `color` (FL_Color bg, FL_Color sel)

Sets the background and selection color of the widget.
- `FL_Color color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const FL_Widget *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.

- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual int `handle` (int event)
Handles the specified event.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)

- Sets the font size in pixels.*

 - `FI_Labeltype labeltype () const`
- Gets the label type.*

 - `void labeltype (FI_Labeltype a)`
- Sets the label type.*

 - `void measure_label (int &ww, int &hh) const`
- Sets width ww and height hh accordingly with the label size.*

 - `bool needs_keyboard () const`
- Returns whether this widget needs a keyboard.*

 - `void needs_keyboard (bool needs)`
- Sets whether this widget needs a keyboard.*

 - `unsigned int output () const`
- Returns if a widget is used for output only.*

 - `FI_Group * parent () const`
- Returns a pointer to the parent widget.*

 - `void parent (FI_Group *p)`
- Internal use only - "for hacks only".*

 - `void position (int X, int Y)`
- Repositions the window or widget.*

 - `void redraw ()`
- Schedules the drawing of the widget.*

 - `void redraw_label ()`
- Schedules the drawing of the label.*

 - `virtual void resize (int x, int y, int w, int h)`
- Changes the size or position of the widget.*

 - `FI_Color selection_color () const`
- Gets the selection color.*

 - `void selection_color (FI_Color a)`
- Sets the selection color.*

 - `void set_active ()`
- Marks the widget as active without sending events or changing focus.*

 - `void set_changed ()`
- Marks the value of the widget as changed.*

 - `void set_output ()`
- Sets a widget to output only.*

 - `void set_visible ()`
- Makes the widget visible.*

 - `void set_visible_focus ()`
- Enables keyboard focus navigation with this widget.*

 - `int shortcut_label () const`
- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - `void shortcut_label (int value)`
- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - `virtual void show ()`
- Makes a widget visible.*

 - `void size (int W, int H)`
- Changes the size of the widget.*

 - `int take_focus ()`
- Gives the widget the keyboard focus.*

 - `unsigned int takeevents () const`
- Returns if the widget is able to take events.*

- `int test_shortcut ()`
Returns true if the widget's label contains the entered '&x' shortcut.
- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- void **draw** () [FL_OVERRIDE](#)
Draw clock with current position and size.
- void **draw** (int X, int Y, int W, int H)
Draw clock with the given position and size.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

```
enum {
    INACTIVE = 1<<0, INVISIBLE = 1<<1, OUTPUT = 1<<2, NOBORDER = 1<<3,
    FORCE_POSITION = 1<<4, NON_MODAL = 1<<5, SHORTCUT_LABEL = 1<<6, CHANGED = 1<<7,
    ,
    OVERRIDE = 1<<8, VISIBLE_FOCUS = 1<<9, COPIED_LABEL = 1<<10, CLIP_CHILDREN = 1<<11,
    ,
    MENU_WINDOW = 1<<12, TOOLTIP_WINDOW = 1<<13, MODAL = 1<<14, NO_OVERLAY = 1<<15,
    ,
    GROUP_RELATIVE = 1<<16, COPIED_TOOLTIP = 1<<17, FULLSCREEN = 1<<18, MAC_USE_ACCENTS_MENU
    = 1<<19,
    NEEDS_KEYBOARD = 1<<20, IMAGE_BOUND = 1<<21, DEIMAGE_BOUND = 1<<22,
    AUTO_DELETE_USER_DATA = 1<<23,
    MAXIMIZED = 1<<24, POPUP = 1<<25, USERFLAG3 = 1<<29, USERFLAG2 = 1<<30,
    USERFLAG1 = 1<<31 }
```

flags possible values enumeration.

11.25.1 Detailed Description

This widget can be used to display a program-supplied time.

The time shown on the clock is not updated. To display the current time, use [Fl_Clock](#) instead.

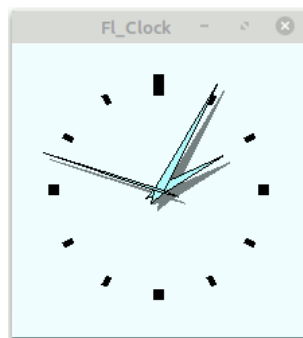


Figure 11.11 FL_SQUARE_CLOCK type

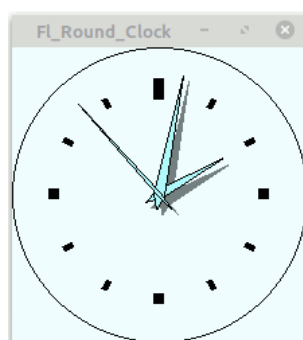


Figure 11.12 FL_ROUND_CLOCK type

Values for clock [type\(\)](#) (`#include <FL/Clock.H>`):

```
#define FL_SQUARE_CLOCK      0      // Square Clock variant
#define FL_ROUND_CLOCK      1      // Round Clock variant
#define FL_ANALOG_CLOCK FL_SQUARE_CLOCK // An analog clock is square
#define FL_DIGITAL_CLOCK FL_SQUARE_CLOCK // Not yet implemented
```


11.25.2 Constructor & Destructor Documentation

11.25.2.1 Fl_Clock_Output()

```
Fl_Clock_Output::Fl_Clock_Output (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Create a new [Fl_Clock_Output](#) widget with the given position, size and label.

The default clock type is `FL_SQUARE_CLOCK` and the default boxtype is `FL_UP_BOX`.

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

11.25.3 Member Function Documentation

11.25.3.1 draw() [1/2]

```
void Fl_Clock_Output::draw (
    void ) [protected], [virtual]
```

Draw clock with current position and size.

Implements [Fl_Widget](#).

11.25.3.2 draw() [2/2]

```
void Fl_Clock_Output::draw (
    int X,
    int Y,
    int W,
    int H) [protected]
```

Draw clock with the given position and size.

Parameters

in	<i>X,Y,W,H</i>	position and size

11.25.3.3 hour()

```
int Fl_Clock_Output::hour () const [inline]
```

Returns the displayed hour (0 to 23).

See also

[value\(\)](#), [minute\(\)](#), [second\(\)](#)

11.25.3.4 minute()

```
int Fl_Clock_Output::minute () const [inline]
```

Returns the displayed minute (0 to 59).

See also

[value\(\)](#), [hour\(\)](#), [second\(\)](#)

11.25.3.5 second()

```
int Fl_Clock_Output::second () const [inline]
```

Returns the displayed second (0 to 60, 60=leap second).

See also

[value\(\)](#), [hour\(\)](#), [minute\(\)](#)

11.25.3.6 shadow() [1/2]

```
int Fl_Clock_Output::shadow () const [inline]
```

Returns the shadow drawing mode of the hands.

Returns

shadow drawing mode of the hands

Return values

0	no shadows
1	draw shadows of hands (default)

11.25.3.7 shadow() [2/2]

```
void Fl_Clock_Output::shadow (
    int mode) [inline]
```

Sets the shadow drawing mode of the hands.

Enables (1) or disables (0) drawing the hands with shadows.

Values except 0 and 1 are reserved for future extensions and yield undefined behavior.

The default is to draw the shadows (1).

Parameters

in	mode	1 = shadows (default), 0 = no shadows

11.25.3.8 value() [1/3]

```
ulong Fl_Clock_Output::value () const [inline]
```

Returns the displayed time.

Returns the time in seconds since the UNIX epoch (January 1, 1970).

See also

[value\(ulong\)](#)

11.25.3.9 value() [2/3]

```
void Fl_Clock_Output::value (
    int H,
    int m,
    int s)
```

Set the displayed time.

Set the time in hours, minutes, and seconds.

Parameters

in	<i>H,m,s</i>	displayed time

See also

[hour\(\)](#), [minute\(\)](#), [second\(\)](#)

11.25.3.10 value() [3/3]

```
void Fl_Clock_Output::value (
    ulong v)
```

Set the displayed time.

Set the time in seconds since the UNIX epoch (January 1, 1970).

Parameters

in	<i>v</i>	seconds since epoch

See also

[value\(\)](#)

The documentation for this class was generated from the following files:

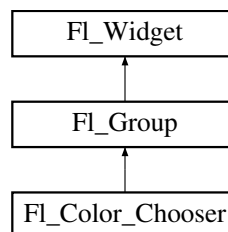
- `Fl_Clock.H`
- `Fl_Clock.cxx`

11.26 Fl_Color_Chooser Class Reference

The [Fl_Color_Chooser](#) widget provides a standard RGB color chooser.

```
#include <Fl_Color_Chooser.H>
```

Inheritance diagram for [Fl_Color_Chooser](#):

**Public Member Functions**

- `double b () const`
Returns the current blue value.
- `Fl\_Color\_Chooser (int X, int Y, int W, int H, const char *L=0)`
Creates a new [Fl_Color_Chooser](#) widget using the given position, size, and label string.
- `double g () const`
Returns the current green value.
- `int handle (int e) FL\_OVERRIDE`
Handles all events received by this widget.
- `int hsv (double H, double S, double V)`
Set the hsv values.
- `double hue () const`

- *Returns the current hue.*
- `int mode ()`
Returns which [FL_Color_Chooser](#) variant is currently active.
- `void mode (int newMode)`
Set which [FL_Color_Chooser](#) variant is currently active.
- `double r () const`
Returns the current red value.
- `int rgb (double R, double G, double B)`
Sets the current rgb color values.
- `double saturation () const`
Returns the saturation.
- `double value () const`
Returns the value/brightness.

Public Member Functions inherited from [FL_Group](#)

- `FL_Widget *& _ddfdesign_kludge ()`
This is for forms compatibility only.
- `void add (FL_Widget &)`
The widget is removed from its current group (if any) and then added to the end of this group.
- `void add (FL_Widget *o)`
See void [FL_Group::add\(FL_Widget &w\)](#)
- `void add_resizable (FL_Widget &o)`
Adds a widget to the group and makes it the resizable widget.
- `FL_Widget *const * array () const`
Returns a pointer to the internal array of children.
- `FL_Group const * as_group () const` `FL_OVERRIDE`
- `FL_Group * as_group ()` `FL_OVERRIDE`
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- `void begin ()`
Sets the current group so you can build the widget tree by just constructing the widgets.
- `FL_Widget * child (int n) const`
Returns the n'th child.
- `int children () const`
Returns how many child widgets the group has.
- `void clear ()`
Deletes all child widgets from memory recursively.
- `unsigned int clip_children ()`
Returns the current clipping mode.
- `void clip_children (int c)`
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- `virtual int delete_child (int n)`
*Removes the widget at *index* from the group and deletes it.*
- `void end ()`
Exactly the same as `current(this->parent())`.
- `int find (const FL_Widget &o) const`
*See int [FL_Group::find\(const FL_Widget *w\) const](#).*
- `int find (const FL_Widget *) const`
Searches the children for the widget and returns the index.
- `FL_Group (int, int, int, int, const char * = 0)`
Creates a new [FL_Group](#) widget using the given position, size, and label string.

- void **focus** (FL_Widget *W)
- void **forms_end** ()
This is for forms compatibility only.
- int **handle** (int) FL_OVERRIDE
Handles the specified event.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** (FL_Widget &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** (FL_Widget &o, FL_Widget *before)
This does insert(w, find(before)).
- void **remove** (FL_Widget &)
Removes a widget from the group but does not delete it.
- void **remove** (FL_Widget *o)
Removes the widget o from the group.
- void **remove** (int index)
Removes the widget at index from the group but does not delete it.
- FL_Widget * **resizable** () const
Returns the group's resizable widget.
- void **resizable** (FL_Widget &o)
Sets the group's resizable widget.
- void **resizable** (FL_Widget *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- void **resize** (int, int, int, int) FL_OVERRIDE
Resizes the FL_Group widget and all of its children.
- virtual ~FL_Group ()
The destructor also deletes all the children.

Public Member Functions inherited from FL_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- FL_Align **align** () const
Gets the label alignment.
- void **align** (FL_Align alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class FL_Gl_Window * **as_gl_window** ()
Returns an FL_Gl_Window pointer if this widget is an FL_Gl_Window.
- virtual class FL_Gl_Window const * **as_gl_window** () const
- virtual FL_Window * **as_window** ()
Returns an FL_Window pointer if this widget is an FL_Window.

- virtual [FI_Window](#) const * **as_window** () const
- void [bind_deimage](#) ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) **box** () const

Gets the box type of the widget.
- void **box** ([FI_Boxtype](#) new_box)

Sets the box type for the widget.
- [FI_Callback_p](#) **callback** () const

Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()

Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.
- void [clear_output](#) ()

Sets a widget to accept input.
- void [clear_visible](#) ()

Hides the widget.
- void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.
- [FI_Color](#) **color** () const

Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)

Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)

Sets the background and selection color of the widget.
- [FI_Color](#) **color2** () const

For back compatibility only.
- void [color2](#) (unsigned a)

For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const

Checks if w is a child of this widget.

- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const

- Checks if this widget is a child of wgt.*

 - int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const

Gets the current label text.
- void `label` (const char *text)

Sets the current label pointer.
- void `label` (FL_Labeltype a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- FL_Color `labelcolor` () const

Gets the label color.
- void `labelcolor` (FL_Color c)

Sets the label color.
- FL_Font `labelfont` () const

Gets the font to use.
- void `labelfont` (FL_Font f)

Sets the font to use.
- FL_Fonsize `labelsize` () const

Gets the font size in pixels.
- void `labelsize` (FL_Fonsize pix)

Sets the font size in pixels.
- FL_Labeltype `labeltype` () const

Gets the label type.
- void `labeltype` (FL_Labeltype a)

Sets the label type.
- void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- FL_Group * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (FL_Group *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- FL_Color `selection_color` () const

Gets the selection color.
- void `selection_color` (FL_Color a)

Sets the selection color.

- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const

- *Checks whether this widget has a visible focus.*
- void `visible_focus` (int v)
Modifies keyboard focus navigation.
- int `visible_r` () const
Returns whether a widget and all its parents are visible.
- int `w` () const
Gets the widget width.
- `Fl_When when` () const
Returns the conditions under which the callback is called.
- void `when` (uchar i)
Sets the flags used to decide when a callback is called.
- `Fl_Window * window` () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual `~Fl_Widget` ()
Destroys the widget.

Static Public Member Functions

- static void `hsv2rgb` (double H, double S, double V, double &R, double &G, double &B)
This static method converts HSV colors to RGB colorspace.
- static void `rgb2hsv` (double R, double G, double B, double &H, double &S, double &V)
This static method converts RGB colors to HSV colorspace.

Static Public Member Functions inherited from `Fl_Group`

- static `Fl_Group * current` ()
Returns the currently active group.
- static void `current` (`Fl_Group *g`)
Sets the current group.

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget *widget`, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Related Symbols

(Note that these are not member symbols.)

- int `fl_color_chooser` (const char *name, double &r, double &g, double &b, int cmode)
Pops up a window to let the user pick an arbitrary RGB color.
- int `fl_color_chooser` (const char *name, uchar &r, uchar &g, uchar &b, int cmode)
Pops up a window to let the user pick an arbitrary RGB color.

Additional Inherited Members

Protected Types inherited from FI_Widget

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
 ,
OVERRIDE = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
 ,
MENU_WINDOW = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
 ,
GROUP_RELATIVE = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
 = 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from FI_Group

- **FI_Rect** * **bounds** ()
Returns the internal array of widget sizes and positions.
- void **draw** () **FL_OVERRIDE**
Draws the widget.
- void **draw_child** (**FI_Widget** &widget) const
Forces a child to redraw.
- void **draw_children** ()
Draws all children of the group.
- void **draw_outside_label** (const **FI_Widget** &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int **on_insert** (**FI_Widget** *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int **on_move** (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void **on_remove** (int)
Allow derived groups to act when a child widget is removed from the group.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** (**FI_Widget** &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (**FI_Boxtype** t, **FI_Color** c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (**FI_Boxtype** t, int x, int y, int w, int h, **FI_Color** c) const

- Draws a box of type t, of color c at the position X,Y and size W,H.*
- void `draw_focus` () const
Draws a focus rectangle around the widget.
- void `draw_focus` (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void `draw_focus` (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `FI_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

11.26.1 Detailed Description

The `FI_Color_Chooser` widget provides a standard RGB color chooser.

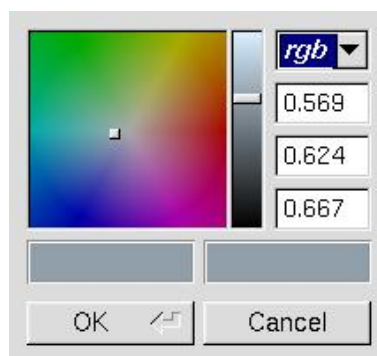


Figure 11.13 `fl_color_chooser()`

You can place any number of the widgets into a panel of your own design. The diagram shows the widget as part of a color chooser dialog created by the `fl_color_chooser()` function. The `FI_Color_Chooser` widget contains the hue box, value slider, and rgb input fields from the above diagram (it does not have the color chips or the Cancel or OK buttons). The callback is done every time the user changes the rgb value. It is not done if they move the hue control in a way that produces the *same* rgb value, such as when saturation or value is zero.

The `fl_color_chooser()` function pops up a window to let the user pick an arbitrary RGB color. They can pick the hue and saturation in the "hue box" on the left (hold down CTRL to just change the saturation), and the brightness using the vertical slider. Or they can type the 8-bit numbers into the RGB `FI_Value_Input` fields, or drag the mouse across them to adjust them. The pull-down menu lets the user set the input fields to show RGB, HSV, or 8-bit RGB (0 to 255).

The user can press CTRL-C to copy the currently selected color value as text in RGB hex format with leading zeroes to the clipboard, for instance `FL_GREEN` would be '00FF00' (since FLTK 1.4.0).

`fl_color_chooser()` returns non-zero if the user picks ok, and updates the RGB values. If the user picks cancel or closes the window this returns zero and leaves RGB unchanged.

If you use the color chooser on an 8-bit screen, it will allocate all the available colors, leaving you no space to exactly represent the color the user picks! You can however use `fl_rectf()` to fill a region with a simulated color using dithering.

Callback reasons can be `FL_REASON_DRAGGED`, `FL_REASON_CHANGED`, or `FL_REASON_RESELECTED`.

11.26.2 Constructor & Destructor Documentation

11.26.2.1 Fl_Color_Chooser()

```
Fl_Color_Chooser::Fl_Color_Chooser (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new `Fl_Color_Chooser` widget using the given position, size, and label string.

The recommended dimensions are 200x95. The color is initialized to black.

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

11.26.3 Member Function Documentation

11.26.3.1 b()

```
double Fl_Color_Chooser::b () const [inline]
```

Returns the current blue value.

$0 \leq b \leq 1$.

11.26.3.2 g()

```
double Fl_Color_Chooser::g () const [inline]
```

Returns the current green value.

$0 \leq g \leq 1$.

11.26.3.3 handle()

```
int Fl_Color_Chooser::handle (
    int e) [virtual]
```

Handles all events received by this widget.

This specific `handle()` method processes the standard 'copy' function as seen in other input widgets. It copies the current color value to the clipboard as a string in RGB format ('RRGGBB').

This format is independent of the `Fl_Color_Chooser` display format setting. No other formats are supplied.

The keyboard events handled are:

- ctrl-c
- ctrl-x
- ctrl-Insert

All other events are processed by the parent class `Fl_Group`.

This enables the **user** to choose a color value, press `ctrl-c` to copy the value to the clipboard and paste it into a color selection widget in another application window or any other text input (e.g. a preferences dialog or an editor).

Note

Keyboard event handling by the current focus widget has priority, hence moving the focus to one of the buttons or selecting text in one of the input widgets effectively disables this special method.

Parameters

in	<i>e</i>	current event

Returns

1 if event has been handled, 0 otherwise

See also

[Fl_Group::handle\(int\)](#)

Reimplemented from [Fl_Widget](#).

11.26.3.4 hsv()

```
int Fl_Color_Chooser::hsv (
    double H,
    double S,
    double V)
```

Set the hsv values.

The passed values are clamped (or for hue, modulus 6 is used) to get legal values. Does not do the callback.

Parameters

in	<i>H,S,V</i>	color components.

Returns

1 if a new hsv value was set, 0 if the hsv value was the previous one.

11.26.3.5 hsv2rgb()

```
void Fl_Color_Chooser::hsv2rgb (
    double H,
    double S,
    double V,
    double & R,
    double & G,
    double & B) [static]
```

This *static* method converts HSV colors to RGB colorspace.

Parameters

in	<i>H,S,V</i>	color components
out	<i>R,G,B</i>	color components

11.26.3.6 hue()

```
double Fl_Color_Chooser::hue () const [inline]
```

Returns the current hue.

0 ≤ hue < 6. Zero is red, one is yellow, two is green, etc. *This value is convenient for the internal calculations - some other systems consider hue to run from zero to one, or from 0 to 360.*

11.26.3.7 mode() [1/2]

```
int Fl_Color_Chooser::mode () [inline]
```

Returns which [Fl_Color_Chooser](#) variant is currently active.

Returns

color modes are rgb(0), byte(1), hex(2), or hsv(3)

11.26.3.8 mode() [2/2]

```
void Fl_Color_Chooser::mode (
    int newMode)
```

Set which [Fl_Color_Chooser](#) variant is currently active.

Parameters

in	<i>newMode</i>	color modes are rgb(0), byte(1), hex(2), or hsv(3)

11.26.3.9 r()

```
double Fl_Color_Chooser::r () const [inline]
```

Returns the current red value.

$0 \leq r \leq 1$.

11.26.3.10 rgb()

```
int Fl_Color_Chooser::rgb (
    double R,
    double G,
    double B)
```

Sets the current rgb color values.

Does not do the callback. Does not clamp (but out of range values will produce psychedelic effects in the hue selector).

Parameters

in	<i>R,G,B</i>	color components.

Returns

1 if a new rgb value was set, 0 if the rgb value was the previous one.

11.26.3.11 rgb2hsv()

```
void Fl_Color_Chooser::rgb2hsv (
    double R,
    double G,
    double B,
    double & H,
    double & S,
    double & V) [static]
```

This *static* method converts RGB colors to HSV colorspace.

Parameters

in	<i>R,G,B</i>	color components
out	<i>H,S,V</i>	color components

11.26.3.12 saturation()

```
double Fl_Color_Chooser::saturation () const [inline]
```

Returns the saturation.

0 <= saturation <= 1.

11.26.3.13 value()

```
double Fl_Color_Chooser::value () const [inline]
```

Returns the value/brightness.

0 <= value <= 1.

The documentation for this class was generated from the following files:

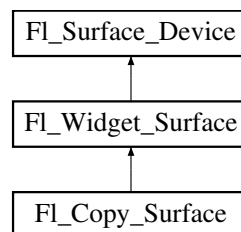
- [Fl_Color_Chooser.H](#)
- [Fl_Color_Chooser.cxx](#)

11.27 Fl_Copy_Surface Class Reference

Supports copying of graphical data to the clipboard.

```
#include <Fl_Copy_Surface.H>
```

Inheritance diagram for Fl_Copy_Surface:



Public Member Functions

- [Fl_Copy_Surface](#) (int **w**, int **h**)
the constructor
- int **h** ()
Returns the pixel height of the copy surface.
- bool [is_current](#) () [FL_OVERRIDE](#)
Is this surface the current drawing surface?
- void [origin](#) (int *x, int *y) [FL_OVERRIDE](#)
Computes the coordinates of the current origin of graphics functions.
- void [origin](#) (int x, int y) [FL_OVERRIDE](#)
Sets the position of the origin of graphics in the drawable part of the drawing surface.
- int [printable_rect](#) (int *w, int *h) [FL_OVERRIDE](#)
Computes the width and height of the drawable area of the drawing surface.
- void [set_current](#) () [FL_OVERRIDE](#)
Make this surface the current drawing surface.
- int **w** ()
Returns the pixel width of the copy surface.
- [~Fl_Copy_Surface](#) ()
the destructor

Public Member Functions inherited from FI_Widget_Surface

- void [draw](#) ([FI_Widget](#) *widget, int delta_x=0, int delta_y=0)
Draws the widget on the drawing surface.
- void [draw_decorated_window](#) ([FI_Window](#) *win, int x_offset=0, int y_offset=0)
Draws a window with its title bar and frame if any.
- void [print_window_part](#) ([FI_Window](#) *win, int x, int y, int w, int h, int delta_x=0, int delta_y=0)
Draws a rectangular part of an on-screen window.

Public Member Functions inherited from FI_Surface_Device

- [FI_Graphics_Driver](#) * **driver** ()
Returns the graphics driver of this drawing surface.
- virtual ~[FI_Surface_Device](#) ()
The destructor.

Protected Member Functions

- void [translate](#) (int x, int y) [FL_OVERRIDE](#)
Translates the current graphics origin accounting for the current rotation.
- void [untranslate](#) () [FL_OVERRIDE](#)
Undoes the effect of a previous [translate\(\)](#) call.

Protected Member Functions inherited from FI_Widget_Surface

- [FI_Widget_Surface](#) ([FI_Graphics_Driver](#) *d)
The constructor.

Protected Member Functions inherited from FI_Surface_Device

- void **driver** ([FI_Graphics_Driver](#) *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void [end_current](#) ()
FLTK calls this each time a surface ceases to be the current drawing surface.
- [FI_Surface_Device](#) ([FI_Graphics_Driver](#) *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

Additional Inherited Members

Static Public Member Functions inherited from FI_Surface_Device

- static [FI_Surface_Device](#) * [pop_current](#) ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void [push_current](#) ([FI_Surface_Device](#) *new_current)
*Pushes *new_current* on top of the stack of current drawing surfaces, and makes it current.*
- static [FI_Surface_Device](#) * [surface](#) ()
The current drawing surface.

Protected Attributes inherited from FI_Widget_Surface

- int **x_offset**
horizontal offset to the origin of graphics coordinates
- int **y_offset**
vertical offset to the origin of graphics coordinates

11.27.1 Detailed Description

Supports copying of graphical data to the clipboard.

After creation of an [Fl_Copy_Surface](#) object, make it the current drawing surface calling [Fl_Surface_Device::push_current\(\)](#), and all subsequent graphics requests will be recorded in the clipboard. It's possible to draw widgets (using [Fl_Copy_Surface::draw\(\)](#)) or to use any of the [Drawing functions](#) or the [Color & Font functions](#). Finally, delete the [Fl_Copy_Surface](#) object to load the clipboard with the graphical data.

[Fl_Gl_Window](#) 's can be copied to the clipboard as well.

Usage example:

```
Fl_Widget *g = ...; // a widget you want to copy to the clipboard
Fl_Copy_Surface *copy_surf = new Fl_Copy_Surface(g->w(), g->h()); // create an Fl_Copy_Surface object
Fl_Surface_Device::push_current(copy_surf); // direct graphics requests to the clipboard
fl_color(FL_WHITE); fl_rectf(0, 0, g->w(), g->h()); // draw a white background
copy_surf->draw(g); // draw the g widget in the clipboard
Fl_Surface_Device::pop_current(); // direct graphics requests back to their previous destination
delete copy_surf; // after this, the clipboard is loaded
```

Platform details:

- Windows: Transparent RGB images copy without transparency. The graphical data are copied to the clipboard in two formats: 1) as an 'enhanced metafile'; 2) as a color bitmap. Applications to which the clipboard content is pasted can use the format that suits them best.
- Mac OS: The graphical data are copied to the clipboard (a.k.a. pasteboard) in two 'flavors': 1) in vectorial form as PDF data; 2) in bitmap form as a TIFF image. Applications to which the clipboard content is pasted can use the flavor that suits them best.
- X11 and Wayland: the graphical data are copied to the clipboard as an image in BMP format.

11.27.2 Constructor & Destructor Documentation

11.27.2.1 Fl_Copy_Surface()

```
Fl_Copy_Surface::Fl_Copy_Surface (
    int w,
    int h)
```

the constructor

Parameters

<i>w,h</i>	Width and height of the drawing surface in FLTK units

11.27.3 Member Function Documentation

11.27.3.1 is_current()

```
bool Fl_Copy_Surface::is_current () [virtual]
```

Is this surface the current drawing surface?

Reimplemented from [Fl_Surface_Device](#).

11.27.3.2 origin() [1/2]

```
void Fl_Copy_Surface::origin (
    int * x,
    int * y) [virtual]
```

Computes the coordinates of the current origin of graphics functions.

Parameters

out	<i>x,y</i>	If non-null, *x and *y are set to the horizontal and vertical coordinates of the graphics origin.

Reimplemented from [Fl_Widget_Surface](#).

11.27.3.3 origin() [2/2]

```
void Fl_Copy_Surface::origin (
    int x,
    int y) [virtual]
```

Sets the position of the origin of graphics in the drawable part of the drawing surface.

Arguments should be expressed relatively to the result of a previous [printable_rect\(\)](#) call. That is, `printable_rect(&w, &h); origin(w/2, 0);` sets the graphics origin at the top center of the drawable area. Successive [origin\(\)](#) calls don't combine their effects. [Origin\(\)](#) calls are not affected by [rotate\(\)](#) calls (for classes derived from [Fl_Paged_Device](#)).

Parameters

in	x,y	Horizontal and vertical positions in the drawing surface of the desired origin of graphics.

Reimplemented from [Fl_Widget_Surface](#).

11.27.3.4 printable_rect()

```
int Fl_Copy_Surface::printable_rect (
    int * w,
    int * h) [virtual]
```

Computes the width and height of the drawable area of the drawing surface.

Values are in the same unit as that used by FLTK drawing functions and are unchanged by calls to [origin\(\)](#). If the object is derived from class [Fl_Paged_Device](#), values account for the user-selected paper type and print orientation and are changed by [scale\(\)](#) calls.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Widget_Surface](#).

11.27.3.5 set_current()

```
void Fl_Copy_Surface::set_current (
    void ) [virtual]
```

Make this surface the current drawing surface.

This surface will receive all future graphics requests.

Since FLTK 1.4.0 the preferred API to change the current drawing surface is [Fl_Surface_Device::push_current\(\)](#) / [Fl_Surface_Device::pop_current\(\)](#).

Note

It is recommended to use this function only as follows :

- The current drawing surface is the display;
- make current another surface, e.g., an [Fl_Printer](#) or an [Fl_Image_Surface](#) object, calling [set_current\(\)](#) on this object;
- draw to that surface;
- make the display current again with [Fl_Display_Device::display_device\(\)->set_current\(\)](#);
don't do any other call to [set_current\(\)](#) before this one.

Other scenarios of drawing surface changes should be performed via [Fl_Surface_Device::push_current\(\)](#) and [Fl_Surface_Device::pop_current\(\)](#).

Reimplemented from [Fl_Surface_Device](#).

11.27.3.6 translate()

```
void Fl_Copy_Surface::translate (
    int x,
    int y) [protected], [virtual]
```

Translates the current graphics origin accounting for the current rotation.

Each [translate\(\)](#) call must be matched by an [untranslate\(\)](#) call. Successive [translate\(\)](#) calls add up their effects.

Reimplemented from [Fl_Widget_Surface](#).

11.27.3.7 untranslate()

```
void Fl_Copy_Surface::untranslate (
    void ) [protected], [virtual]
```

Undoes the effect of a previous [translate\(\)](#) call.

Reimplemented from [Fl_Widget_Surface](#).

The documentation for this class was generated from the following files:

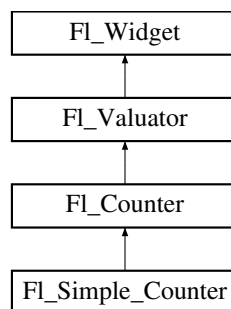
- [Fl_Copy_Surface.H](#)
- [Fl_Copy_Surface.cxx](#)

11.28 Fl_Counter Class Reference

Controls a single floating point value with button (or keyboard) arrows.

```
#include <Fl_Counter.H>
```

Inheritance diagram for [Fl_Counter](#):



Public Member Functions

- [Fl_Counter](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Counter](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- void [lstep](#) (double a)
Sets the increment for the large step buttons.
- double [step](#) () const
Returns the increment for normal step buttons.
- void [step](#) (double a)
Sets the increment for the normal step buttons.
- void [step](#) (double a, double b)
Sets the increments for the normal and large step buttons.
- [Fl_Color](#) [textcolor](#) () const
Gets the font color.
- void [textcolor](#) ([Fl_Color](#) s)
Sets the font color to s.
- [Fl_Font](#) [textfont](#) () const

- Gets the text font.*
 - void **textfont** ([FI_Font](#) s)
 - Sets the text font to s.*
 - [FI_Fonsize](#) **textsize** () const
 - Gets the font size.*
 - void **textsize** ([FI_Fonsize](#) s)
 - Sets the font size to s.*
- **~FI_Counter** ()
 - Destroys the valuator.*

Public Member Functions inherited from [FI_Valuator](#)

- void **bounds** (double a, double b)
 - Sets the minimum (a) and maximum (b) values for the valuator widget.*
- double **clamp** (double)
 - Clamps the passed value to the valuator range.*
- virtual int **format** (char *)
 - Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.*
- double **increment** (double, int)
 - Adds n times the step value to the passed value.*
- double **maximum** () const
 - Gets the maximum value for the valuator.*
- void **maximum** (double a)
 - Sets the maximum value for the valuator.*
- double **minimum** () const
 - Gets the minimum value for the valuator.*
- void **minimum** (double a)
 - Sets the minimum value for the valuator.*
- void **precision** (int digits)
 - Sets the step value to $1.0 / 10^{\text{digits}}$.*
- void **range** (double a, double b)
 - Sets the minimum and maximum values for the valuator.*
- double **round** (double)
 - Round the passed value to the nearest step increment.*
- double **step** () const
 - Gets or sets the step value.*
- void **step** (double a, int b)
 - See double [FI_Valuator::step\(\)](#) const*
- void **step** (double s)
 - See double [FI_Valuator::step\(\)](#) const.*
- void **step** (int a)
 - See double [FI_Valuator::step\(\)](#) const*
- double **value** () const
 - Gets the floating point(double) value.*
- int **value** (double)
 - Sets the current value.*
- **~FI_Valuator** () [FL_OVERRIDE](#)
 - Destructor is accessible despite protected constructor.*

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color color` () const

Gets the background color of the widget.
- void `color` (FL_Color bg)

Sets the background color of the widget.
- void `color` (FL_Color bg, FL_Color sel)

Sets the background and selection color of the widget.
- `FL_Color color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const FL_Widget *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([FL_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- `void arrow_widths (int &w1, int &w2)`
Compute sizes (widths) of arrow boxes.
- `void draw () FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from [FL_Valuator](#)

- [FL_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FL_Valuator](#) widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void [value_damage](#) ()
Asks for partial redraw.

Protected Member Functions inherited from [FL_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FL_Boxtype](#) t, [FL_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FL_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.
- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FL_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)

Internal use only.

- void `x` (int v)

Internal use only.

- void `y` (int v)

Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *widget, void *data)

The default callback for all widgets that don't set a callback.

- static unsigned int `label_shortcut` (const char *t)

Returns the Unicode value of the '&x' shortcut in a given text.

- static int `test_shortcut` (const char *, const bool require_alt=false)

Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
 ,
`OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
 ,
`MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
 ,
`GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
 = 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }

flags possible values enumeration.

11.28.1 Detailed Description

Controls a single floating point value with button (or keyboard) arrows.
 Double arrows buttons achieve larger steps than simple arrows.

See also

`FI_Spinner` for `value` input with vertical `step` arrows.

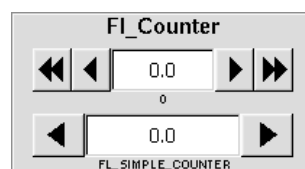


Figure 11.14 `FI_Counter`

The type of an `FI_Counter` object can be set using `FI_Widget::type(uchar)` to:

- `FL_NORMAL_COUNTER`: Displays a counter with 4 arrow buttons.
- `FL_SIMPLE_COUNTER`: Displays a counter with only 2 arrow buttons.

11.28.2 Constructor & Destructor Documentation

11.28.2.1 Fl_Counter()

```
Fl_Counter::Fl_Counter (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Counter](#) widget using the given position, size, and label string. The default type is FL_NORMAL_COUNTER.

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

11.28.3 Member Function Documentation

11.28.3.1 arrow_widths()

```
void Fl_Counter::arrow_widths (
    int & w1,
    int & w2) [protected]
```

Compute sizes (widths) of arrow boxes.

This method computes the two sizes of the arrow boxes of [Fl_Counter](#). You can override it in a subclass if you want to draw fancy arrows or change the layout. However, the basic layout is fixed and can't be changed w/o overriding the [draw\(\)](#) and [handle\(\)](#) methods.

Basic layout:

```
+-----+-----+-----+-----+
| « | < |   value   | > | » |
+-----+-----+-----+-----+
```

The returned value w2 should be zero if the counter [type\(\)](#) is FL_SIMPLE_COUNTER.

Parameters

out	<i>w1</i>	width of single arrow box
out	<i>w2</i>	width of double arrow box

11.28.3.2 draw()

```
void Fl_Counter::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.28.3.3 handle()

```
int Fl_Counter::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

11.28.3.4 lstep()

```
void Fl_Counter::lstep (
    double a) [inline]
```

Sets the increment for the large step buttons.
The default value is 1.0.

Parameters

in	<i>a</i>	large step increment.

11.28.3.5 step() [1/2]

```
void Fl_Counter::step (
    double a) [inline]
```

Sets the increment for the normal step buttons.

Parameters

in	<i>a</i>	normal step increment.

11.28.3.6 step() [2/2]

```
void Fl_Counter::step (
    double a,
    double b) [inline]
```

Sets the increments for the normal and large step buttons.

Parameters

in	a,b	normal and large step increments.

The documentation for this class was generated from the following files:

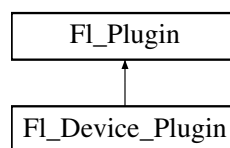
- FI_Counter.H
- FI_Counter.cxx

11.29 FI_Device_Plugin Class Reference

This plugin socket allows the integration of new device drivers for special window or screen types.

```
#include <FI_Device.H>
```

Inheritance diagram for FI_Device_Plugin:



Public Member Functions

- **FI_Device_Plugin** (const char *pluginName)
The constructor.
- virtual const char * **klass** ()
Returns the class name.
- virtual const char * **name** ()=0
Returns the plugin name.
- virtual int **print** (FI_Widget *w)=0
Prints a widget.
- virtual FI_RGB_Image * **rectangle_capture** (FI_Widget *widget, int x, int y, int w, int h)=0
Captures a rectangle of a widget as an image.

Public Member Functions inherited from FI_Plugin

- FI_Plugin (const char *klass, const char *name)
Create a plugin.
- virtual ~FI_Plugin ()
Clear the plugin and remove it from the database.

Static Public Member Functions

- static FI_Device_Plugin * **opengl_plugin** ()
Returns the OpenGL plugin.

11.29.1 Detailed Description

This plugin socket allows the integration of new device drivers for special window or screen types.

This class is not intended for use outside the FLTK library. It is currently used to provide an automated printing service and screen capture for OpenGL windows, if linked with fltk_gl.

11.29.2 Member Function Documentation

11.29.2.1 rectangle_capture()

```
virtual Fl\_RGB\_Image * Fl_Device_Plugin::rectangle_capture (
    Fl\_Widget * widget,
    int x,
    int y,
    int w,
    int h) [pure virtual]
```

Captures a rectangle of a widget as an image.

Returns

The captured pixels as an RGB image

The documentation for this class was generated from the following files:

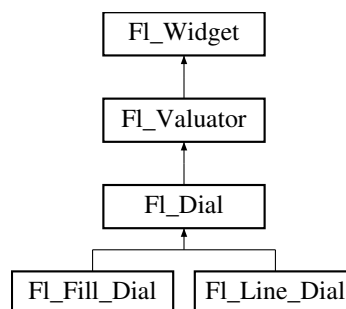
- [Fl_Device.H](#)
- [Fl_Device.cxx](#)

11.30 Fl_Dial Class Reference

The [Fl_Dial](#) widget provides a circular dial to control a single floating point value.

```
#include <Fl_Dial.H>
```

Inheritance diagram for [Fl_Dial](#):



Public Member Functions

- short [angle1](#) () const
Sets Or gets the angles used for the minimum and maximum values.
- void [angle1](#) (short a)
See short [angle1\(\)](#) const.
- short [angle2](#) () const
See short [angle1\(\)](#) const.
- void [angle2](#) (short a)
See short [angle1\(\)](#) const.
- void [angles](#) (short a, short b)
See short [angle1\(\)](#) const.
- [Fl_Dial](#) (int x, int y, int w, int h, const char *l=0)
Creates a new [Fl_Dial](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Allow subclasses to handle event based on current position and size.

Public Member Functions inherited from FI_Valuator

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)
Sets the maximum value for the valuator.
- double **minimum** () const
Gets the minimum value for the valuator.
- void **minimum** (double a)
Sets the minimum value for the valuator.
- void **precision** (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void **range** (double a, double b)
Sets the minimum and maximum values for the valuator.
- double **round** (double)
Round the passed value to the nearest step increment.
- double **step** () const
Gets or sets the step value.
- void **step** (double a, int b)
See double FI_Valuator::step() const
- void **step** (double s)
See double FI_Valuator::step() const.
- void **step** (int a)
See double FI_Valuator::step() const
- double **value** () const
Gets the floating point(double) value.
- int **value** (double)
Sets the current value.
- ~FI_Valuator () **FL_OVERRIDE**
Destructor is accessible despite protected constructor.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- FI_Align **align** () const

- Gets the label alignment.*

 - void [align](#) ([FI_Align](#) alignment)
- Sets the label alignment.*

 - long [argument](#) () const
- Gets the current user data (long) argument that is passed to the callback function.*

 - void [argument](#) (long v)
- Sets the current user data (long) argument that is passed to the callback function.*

 - virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
- Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).*

 - virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).*

 - virtual [FI_Group](#) * [as_group](#) ()
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*

 - virtual [FI_Group](#) const * [as_group](#) () const
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*

 - virtual [FI_Window](#) * [as_window](#) ()
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*

 - virtual [FI_Window](#) const * [as_window](#) () const
- Sets the image to use as part of the widget label when in the inactive state.*

 - void [bind_deimage](#) ([FI_Image](#) *img)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*

 - void [bind_deimage](#) (int f)
- Sets the image to use as part of the widget label when in the active state.*

 - void [bind_image](#) ([FI_Image](#) *img)
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*

 - void [bind_image](#) (int f)
- Gets the box type of the widget.*

 - [FI_Boxtype](#) [box](#) () const
- Sets the box type for the widget.*

 - void [box](#) ([FI_Boxtype](#) new_box)
- Gets the current callback function for the widget.*

 - [FI_Callback_p](#) [callback](#) () const
- Sets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb)
- Sets the current callback function and managed user data for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and data for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, void *p)
- Sets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback0](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)
- Checks if the widget value changed since the last callback.*

 - unsigned int [changed](#) () const
- Marks the widget as inactive without sending events or changing focus.*

 - void [clear_active](#) ()
- Marks the value of the widget as unchanged.*

 - void [clear_changed](#) ()
- Clears or sets the damage flags.*

 - void [clear_damage](#) (uchar c=0)
- Sets a widget to accept input.*

 - void [clear_output](#) ()
- Sets a widget to accept input.*

 - void [clear_visible](#) ()

- Hides the widget.*
- void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*
- `FL_Color` `color` () const
- Gets the background color of the widget.*
- void `color` (`FL_Color` bg)
- Sets the background color of the widget.*
- void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*
- `FL_Color` `color2` () const
- For back compatibility only.*
- void `color2` (unsigned a)
- For back compatibility only.*
- int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*
- void `copy_label` (const char *new_label)
- Sets the current label.*
- void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*
- `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*
- void `damage` (`uchar` c)
- Sets the damage bits for the widget.*
- void `damage` (`uchar` c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int `damage_resize` (int, int, int, int)
- Internal use only.*
- void `deactivate` ()
- Deactivates the widget.*
- `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*
- const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with default arguments.*
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*
- void `draw_label` (int, int, int, int, `FL_Align`) const
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int `h` () const
- Gets the widget height.*
- virtual void `hide` ()
- Makes a widget invisible.*

- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FI_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FI_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FI_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FI_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FI_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FI_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FI_Color](#) c)
Sets the label color.
- [FI_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FI_Font](#) f)
Sets the font to use.
- [FI_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FI_Fonsize](#) pix)
Sets the font size in pixels.
- [FI_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FI_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

- Returns if a widget is used for output only.*

 - `Fl_Group * parent () const`

Returns a pointer to the parent widget.
- `void parent (Fl_Group *p)`

Internal use only - "for hacks only".
- `void position (int X, int Y)`

Repositions the window or widget.
- `void redraw ()`

Schedules the drawing of the widget.
- `void redraw_label ()`

Schedules the drawing of the label.
- `virtual void resize (int x, int y, int w, int h)`

Changes the size or position of the widget.
- `Fl_Color selection_color () const`

Gets the selection color.
- `void selection_color (Fl_Color a)`

Sets the selection color.
- `void set_active ()`

Marks the widget as active without sending events or changing focus.
- `void set_changed ()`

Marks the value of the widget as changed.
- `void set_output ()`

Sets a widget to output only.
- `void set_visible ()`

Makes the widget visible.
- `void set_visible_focus ()`

Enables keyboard focus navigation with this widget.
- `int shortcut_label () const`

Returns whether the widget's label uses '&' to indicate shortcuts.
- `void shortcut_label (int value)`

Sets whether the widget's label uses '&' to indicate shortcuts.
- `virtual void show ()`

Makes a widget visible.
- `void size (int W, int H)`

Changes the size of the widget.
- `int take_focus ()`

Gives the widget the keyboard focus.
- `unsigned int takeevents () const`

Returns if the widget is able to take events.
- `int test_shortcut ()`

Returns true if the widget's label contains the entered '&x' shortcut.
- `const char * tooltip () const`

Gets the current tooltip text.
- `void tooltip (const char *text)`

Sets the current tooltip text.
- `Fl_Window * top_window () const`

Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`

Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`

Gets the widget type.

- void **type** (uchar t)
Sets the widget type.
- int **use_accents_menu** ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const
Gets the user data for this widget.
- void **user_data** (Fl_Callback_User_Data *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **Fl_When** when () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual ~**Fl_Widget** ()
Destroys the widget.

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
Draws dial at current position and size.
- void **draw** (int X, int Y, int W, int H)
Draws dial at given position and size.
- int **handle** (int event, int X, int Y, int W, int H)
Allows subclasses to handle event based on given position and size.

Protected Member Functions inherited from FI_Valuator

- **FI_Valuator** (int X, int Y, int W, int H, const char *L)
Creates a new FI_Valuator widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void **value_damage** ()
Asks for partial redraw.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **FI_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)

Internal use only.

- void `x` (int v)

Internal use only.

- void `y` (int v)

Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget` *widget, void *data)

The default callback for all widgets that don't set a callback.

- static unsigned int `label_shortcut` (const char *t)

Returns the Unicode value of the '&x' shortcut in a given text.

- static int `test_shortcut` (const char *, const bool require_alt=false)

Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `Fl_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
 ,
`OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
 ,
`MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
 ,
`GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
 = 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }

flags possible values enumeration.

11.30.1 Detailed Description

The `Fl_Dial` widget provides a circular dial to control a single floating point value.

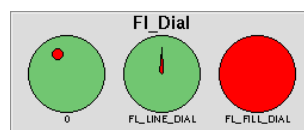


Figure 11.15 `Fl_Dial`

Use `type()` to set the type of the dial to:

- `FL_NORMAL_DIAL` - Draws a normal dial with a knob.
- `FL_LINE_DIAL` - Draws a dial with a line.
- `FL_FILL_DIAL` - Draws a dial with a filled arc.

11.30.2 Constructor & Destructor Documentation

11.30.2.1 Fl_Dial()

```
Fl_Dial::Fl_Dial (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Dial](#) widget using the given position, size, and label string. The default type is FL_NORMAL_DIAL.

11.30.3 Member Function Documentation

11.30.3.1 angle1()

```
short Fl_Dial::angle1 () const [inline]
```

Sets Or gets the angles used for the minimum and maximum values.

The default values are 45 and 315 (0 degrees is straight down and the angles progress clockwise). Normally angle1 is less than angle2, but if you reverse them the dial moves counter-clockwise.

11.30.3.2 draw() [1/2]

```
void Fl_Dial::draw (
    void ) [protected], [virtual]
```

Draws dial at current position and size.

Implements [Fl_Widget](#).

11.30.3.3 draw() [2/2]

```
void Fl_Dial::draw (
    int X,
    int Y,
    int W,
    int H) [protected]
```

Draws dial at given position and size.

Parameters

in	<i>X,Y,W,H</i>	position and size

11.30.3.4 handle() [1/2]

```
int Fl_Dial::handle (
    int event,
    int X,
    int Y,
    int W,
    int H) [protected]
```

Allows subclasses to handle event based on given position and size.

Parameters

in	<i>event,X,Y,W,H</i>	event to handle, related position and size.

11.30.3.5 handle() [2/2]

```
int Fl_Dial::handle (
    int e) [virtual]
```

Allow subclasses to handle event based on current position and size.

Reimplemented from [Fl_Widget](#).

The documentation for this class was generated from the following files:

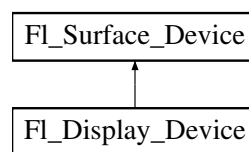
- [Fl_Dial.H](#)
- [Fl_Dial.cxx](#)

11.31 Fl_Display_Device Class Reference

The computer's display.

```
#include <Fl_Device.H>
```

Inheritance diagram for `Fl_Display_Device`:



Static Public Member Functions

- static [Fl_Display_Device](#) * **display_device** ()
Returns a pointer to the unique display device.

Static Public Member Functions inherited from [Fl_Surface_Device](#)

- static [Fl_Surface_Device](#) * **pop_current** ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void **push_current** ([Fl_Surface_Device](#) *new_current)
Pushes new_current on top of the stack of current drawing surfaces, and makes it current.
- static [Fl_Surface_Device](#) * **surface** ()
The current drawing surface.

Additional Inherited Members

Public Member Functions inherited from [Fl_Surface_Device](#)

- [Fl_Graphics_Driver](#) * **driver** ()
Returns the graphics driver of this drawing surface.
- virtual bool **is_current** ()
Is this surface the current drawing surface?
- virtual void **set_current** (void)
Make this surface the current drawing surface.
- virtual ~**Fl_Surface_Device** ()
The destructor.

Protected Member Functions inherited from [Fl_Surface_Device](#)

- void **driver** ([Fl_Graphics_Driver](#) *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void **end_current** ()

FLTK calls this each time a surface ceases to be the current drawing surface.

- **FL_Surface_Device** (FL_Graphics_Driver *graphics_driver)

Constructor that sets the graphics driver to use for the created surface.

11.31.1 Detailed Description

The computer's display.

When FLTK begins to access the computer's display, it creates an object of class [FL_Display_Device](#) and makes it the current drawing surface.

The documentation for this class was generated from the following files:

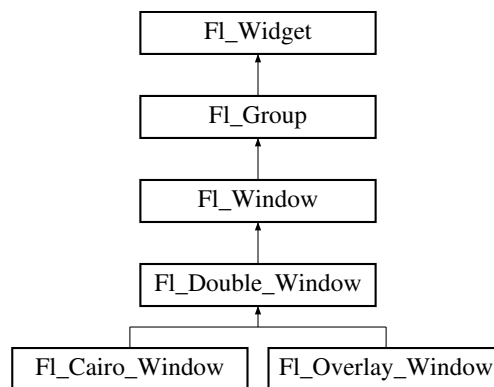
- [FL_Device.H](#)
- [FL_Device.cxx](#)

11.32 FL_Double_Window Class Reference

The [FL_Double_Window](#) provides a double-buffered window.

```
#include <FL_Double_Window.H>
```

Inheritance diagram for [FL_Double_Window](#):



Public Member Functions

- [FL_Double_Window * as_double_window \(\)](#) **FL_OVERRIDE**
Return non-null if this is an [FL_Double_Window](#) object.
- **FL_Double_Window** (int W, int H, const char *l=0)
Creates a new [FL_Double_Window](#) widget using the given position, size, and label (title) string.
- **FL_Double_Window** (int X, int Y, int W, int H, const char *l=0)
See [FL_Double_Window::FL_Double_Window\(int w, int h, const char *label = 0\)](#)
- void **flush ()** **FL_OVERRIDE**
Forces the window to be drawn, this window is also made current and calls [draw\(\)](#).
- void **hide ()** **FL_OVERRIDE**
Makes a widget invisible.
- void **resize** (int, int, int, int) **FL_OVERRIDE**
Changes the size or position of the widget.
- void **show ()** **FL_OVERRIDE**
Makes a widget visible.
- void **show** (int a, char **b)
Same as [FL_Window::show\(int a, char **b\)](#)
- **~FL_Double_Window ()**
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Window](#)

- void [allow_expand_outside_parent](#) ()
Allow this subwindow to expand outside the area of its parent window.
- virtual class [FL_Overlay_Window](#) * [as_overlay_window](#) ()
Return non-null if this is an [FL_Overlay_Window](#) object.
- [FL_Window](#) const * [as_window](#) () const [FL_OVERRIDE](#)
- [FL_Window](#) * [as_window](#) () [FL_OVERRIDE](#)
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).
- unsigned int **border** () const
Returns whether the window possesses a border.
- void [border](#) (int b)
Sets whether or not the window manager border is around the window.
- void [clear_border](#) ()
Fast inline function to turn the window manager border off.
- void [clear_modal_states](#) ()
Clears the "modal" flags and converts a "modal" or "non-modal" window back into a "normal" window.
- void **copy_label** (const char *a)
Sets the window titlebar label to a copy of a character string.
- void [cursor](#) (const [FL_RGB_Image](#) *, int, int)
Changes the cursor for this window using the provided image as cursor's shape.
- void [cursor](#) ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void [cursor](#) ([FL_Cursor](#))
Changes the cursor for this window.
- int [decorated_h](#) () const
Returns the window height including any window title bar and any frame added by the window manager.
- int [decorated_w](#) () const
Returns the window width including any frame added by the window manager.
- void [default_cursor](#) ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void [default_cursor](#) ([FL_Cursor](#))
Sets the default window cursor.
- void **draw_backdrop** ()
Draw the background image if one is set and is aligned inside.
- [FL_Window](#) (int w, int h, const char *title=0)
Creates a window from the given width w, height h, and title.
- [FL_Window](#) (int x, int y, int w, int h, const char *title=0)
Creates a window from the given position (x, y), size (w, h) and title.
- void [free_position](#) ()
Undoes the effect of a previous [resize\(\)](#) or [show\(\)](#) so that the next time [show\(\)](#) is called the window manager is free to position the window.
- void [fullscreen](#) ()
Makes the window completely fill one or more screens, without any window manager border visible.
- unsigned int **fullscreen_active** () const
Returns non zero if [FULLSCREEN](#) flag is set, 0 otherwise.
- void **fullscreen_off** ()
Turns off any side effects of [fullscreen\(\)](#)
- void **fullscreen_off** (int X, int Y, int W, int H)
Turns off any side effects of [fullscreen\(\)](#) and does [resize\(x,y,w,h\)](#).
- void [fullscreen_screens](#) (int top, int bottom, int left, int right)

Sets which screens should be used when this window is in fullscreen mode.

- **uchar** **get_size_range** (int *minw, int *minh, int *maxw=NULL, int *maxh=NULL, int *dw=NULL, int *dh=NULL, int *aspect=NULL)

Gets the allowable range to which the user can resize this window.

- int **handle** (int) **FL_OVERRIDE**

Handles the specified event.

- void **hide** () **FL_OVERRIDE**

Removes the window from the screen.

- void **hotspot** (const **FL_Widget** &p, int offscreen=0)

*See void **FL_Window::hotspot**(int x, int y, int offscreen = 0)*

- void **hotspot** (const **FL_Widget** *, int offscreen=0)

*See void **FL_Window::hotspot**(int x, int y, int offscreen = 0)*

- void **hotspot** (int x, int y, int offscreen=0)

Positions the window so that the mouse is pointing at the given position, or at the center of the given widget, which may be the window itself.

- const void * **icon** () const

Gets the current icon window target dependent data.

- void **icon** (const **FL_RGB_Image** *)

Sets or resets a single window icon.

- void **icon** (const void *ic)

Platform-specific method to set the window icon usable on Windows and X11 only.

- void **iconize** ()

Iconifies the window.

- const char * **iconlabel** () const

*See void **FL_Window::iconlabel**(const char*)*

- void **iconlabel** (const char *)

Sets the icon label.

- void **icons** (const **FL_RGB_Image** *[], int)

Sets the window icons.

- void **icons** (HICON big_icon, HICON small_icon)

Sets the window icons using HICON handles (Windows platform only).

- const char * **label** () const

*See void **FL_Window::label**(const char*)*

- void **label** (const char *)

Sets the window title bar label.

- void **label** (const char *label, const char *iconlabel)

Sets the icon label.

- void **make_current** ()

*Sets things up so that the drawing functions in <**FL/fl_draw.H**> will go into this window.*

- void **maximize** ()

Maximizes a top-level window to its current screen.

- unsigned int **maximize_active** () const

Returns whether the window is currently maximized.

- unsigned int **menu_window** () const

Returns true if this window is a menu window.

- unsigned int **modal** () const

Returns true if this window is modal.

- unsigned int **non_modal** () const

Returns true if this window is modal or non-modal.

- `fl_uintptr_t os_id ()`
Returns a platform-specific identification of a shown window, or 0 if not shown.
- `unsigned int override () const`
Returns non zero if `OVERRIDE` flag is set, 0 otherwise.
- `void resize (int X, int Y, int W, int H) FL_OVERRIDE`
Changes the size and position of the window.
- `int screen_num ()`
The number of the screen containing the mapped window.
- `void screen_num (int screen_num)`
Set the number of the screen where to map the window.
- `void set_menu_window ()`
Marks the window as a menu window.
- `void set_modal ()`
A "modal" window, when `shown()`, will prevent any events from being delivered to other windows in the same program, and will also remain on top of the other windows (if the X window manager supports the "transient for" property).
- `void set_non_modal ()`
A "non-modal" window (terminology borrowed from Microsoft Windows) acts like a `modal()` one in that it remains on top, but it has no effect on event delivery.
- `void set_override ()`
Activates the flags `NOBORDER|OVERRIDE`.
- `void set_tooltip_window ()`
Marks the window as a tooltip window.
- `const FL_Image * shape ()`
Returns the image controlling the window shape or `NULL`.
- `void shape (const FL_Image &b)`
Set the window's shape with an `FL_Image`.
- `void shape (const FL_Image *img)`
Assigns a non-rectangular shape to the window.
- `void show () FL_OVERRIDE`
Puts the window on the screen.
- `void show (int argc, char **argv)`
Puts the window on the screen with `show()` and parses command-line arguments.
- `int shown ()`
Returns non-zero if `show()` has been called (but not `hide()`).
- `void size_range (int minw, int minh, int maxw=0, int maxh=0, int dw=0, int dh=0, int aspect=0)`
Sets the allowable range to which the user can resize this window.
- `unsigned int tooltip_window () const`
Returns true if this window is a tooltip window.
- `void un_maximize ()`
Returns a previously maximized top-level window to its previous size.
- `void wait_for_expose ()`
Waits for the window to be displayed after calling `show()`.
- `int x_root () const`
Gets the x position of the window on the screen.
- `const char * xclass () const`
Returns the xclass for this window, or a default.
- `void xclass (const char *c)`
Sets the xclass for this window.
- `int y_root () const`
Gets the y position of the window on the screen.
- `virtual ~FL_Window ()`
The destructor also deletes all the children.

Public Member Functions inherited from FL_Group

- [FL_Widget](#) *[_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void **add** ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void **add_resizable** ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * **array** () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * **as_group** () const [FL_OVERRIDE](#)
- [FL_Group](#) * **as_group** () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
*Removes the widget at *index* from the group and deletes it.*
- void **end** ()
*Exactly the same as *current(this->parent())*.*
- int **find** (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int **find** (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char * = 0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void **focus** ([FL_Widget](#) *W)
- void **forms_end** ()
This is for forms compatibility only.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** ([FL_Widget](#) &o, [FL_Widget](#) *before)
*This does *insert(w, find(before))*.*
- void **remove** ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void **remove** ([FL_Widget](#) *o)
Removes the widget o from the group.
- void **remove** (int index)

- Removes the widget at `index` from the group but does not delete it.*

 - `FI_Widget * resizable () const`
 - Returns the group's resizable widget.*
 - `void resizable (FI_Widget &o)`
 - Sets the group's resizable widget.*
 - `void resizable (FI_Widget *o)`
 - The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
 - `virtual ~FI_Group ()`
 - The destructor also deletes all the children.*

Public Member Functions inherited from `FI_Widget`

- `void _clear_fullscreen ()`
- `void _set_fullscreen ()`
- `void activate ()`
 - Activates the widget.*
- `unsigned int active () const`
 - Returns whether the widget is active.*
- `int active_r () const`
 - Returns whether the widget and all of its parents are active.*
- `FI_Align align () const`
 - Gets the label alignment.*
- `void align (FI_Align alignment)`
 - Sets the label alignment.*
- `long argument () const`
 - Gets the current user data (long) argument that is passed to the callback function.*
- `void argument (long v)`
 - Sets the current user data (long) argument that is passed to the callback function.*
- `virtual class FI_GL_Window * as_gl_window ()`
 - Returns an `FI_GL_Window` pointer if this widget is an `FI_GL_Window`.*
- `virtual class FI_GL_Window const * as_gl_window () const`
- `void bind_deimage (FI_Image *img)`
 - Sets the image to use as part of the widget label when in the inactive state.*
- `void bind_deimage (int f)`
 - Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- `void bind_image (FI_Image *img)`
 - Sets the image to use as part of the widget label when in the active state.*
- `void bind_image (int f)`
 - Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- `FI_Boxtype box () const`
 - Gets the box type of the widget.*
- `void box (FI_Boxtype new_box)`
 - Sets the box type for the widget.*
- `FI_Callback_p callback () const`
 - Gets the current callback function for the widget.*
- `void callback (FI_Callback *cb)`
 - Sets the current callback function for the widget.*
- `void callback (FI_Callback *cb, FI_Callback_User_Data *p, bool auto_free)`
 - Sets the current callback function and managed user data for the widget.*
- `void callback (FI_Callback *cb, void *p)`
 - Sets the current callback function and data for the widget.*

- void `callback` (`FI_Callback0` *cb)
Sets the current callback function for the widget.
- void `callback` (`FI_Callback1` *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int `changed` () const
Checks if the widget value changed since the last callback.
- void `clear_active` ()
Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()
Marks the value of the widget as unchanged.
- void `clear_damage` (`uchar` c=0)
Clears or sets the damage flags.
- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FI_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FI_Color` bg)
Sets the background color of the widget.
- void `color` (`FI_Color` bg, `FI_Color` sel)
Sets the background and selection color of the widget.
- `FI_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FI_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FI_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FI_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FI_Image` *img)

- Sets the image to use as part of the widget label when in the inactive state.*

 - `int deimage_bound () const`

Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FL_Align) const`

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`

Gets the widget height.
- `int horizontal_label_margin ()`

Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`

Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`

Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`

Gets the image that is used as part of the widget label when in the active state.
- `void image (FL_Image &img)`

Sets the image to use as part of the widget label when in the active state.
- `void image (FL_Image *img)`

Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`

Returns whether the image is managed by the widget.
- `int inside (const FL_Widget *wgt) const`

Checks if this widget is a child of wgt.
- `int is_label_copied () const`

Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`

Gets the current label text.
- `void label (const char *text)`

Sets the current label pointer.
- `void label (FL_Labeltype a, const char *b)`

Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`

Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`

Set the gap between the label and the image in pixels.
- `FL_Color labelcolor () const`

Gets the label color.
- `void labelcolor (FL_Color c)`

Sets the label color.
- `FL_Font labelfont () const`

Gets the font to use.
- `void labelfont (FL_Font f)`

Sets the font to use.
- `FL_Fonsize labelsiz () const`

Gets the font size in pixels.

- void [labelsize](#) ([Fl_Fontsize](#) pix)
Sets the font size in pixels.
- [Fl_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([Fl_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [Fl_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([Fl_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [Fl_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

- Gets the current tooltip text.*

 - void **tooltip** (const char *text)

Sets the current tooltip text.
- **Fl_Window** * **top_window** () const

Returns a pointer to the top-level window for the widget.
- **Fl_Window** * **top_window_offset** (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- **uchar** **type** () const

Gets the widget type.
- void **type** (**uchar** t)

Sets the widget type.
- int **use_accents_menu** ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const

Gets the user data for this widget.
- void **user_data** (**Fl_Callback_User_Data** *v, bool auto_free)

Sets the user data for this widget.
- void **user_data** (void *v)

Sets the user data for this widget.
- int **vertical_label_margin** ()

Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const

Returns whether a widget is visible.
- unsigned int **visible_focus** () const

Checks whether this widget has a visible focus.
- void **visible_focus** (int v)

Modifies keyboard focus navigation.
- int **visible_r** () const

Returns whether a widget and all its parents are visible.
- int **w** () const

Gets the widget width.
- **Fl_When** **when** () const

Returns the conditions under which the callback is called.
- void **when** (**uchar** i)

Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const

Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const

Gets the widget position in its window.
- int **y** () const

Gets the widget position in its window.
- virtual **~Fl_Widget** ()

Destroys the widget.

Additional Inherited Members

Public Types inherited from **Fl_Window**

- typedef struct HICON__ * **HICON**

Static Public Member Functions inherited from FI_Window

- static [FI_Window](#) * [current](#) ()
Returns the last window that was made current.
- static void [default_callback](#) ([FI_Window](#) *, void *v)
Back compatibility: Sets the default callback v for win to call on close event.
- static void [default_icon](#) (const [FI_RGB_Image](#) *)
Sets a single default window icon.
- static void [default_icons](#) (const [FI_RGB_Image](#) *[], int)
Sets the default window icons.
- static void [default_icons](#) (HICON big_icon, HICON small_icon)
Sets the default window icons (Windows platform only).
- static const char * [default_xclass](#) ()
Returns the default xclass.
- static void [default_xclass](#) (const char *)
Sets the default window xclass.
- static bool [is_a_rescale](#) ()
Returns true when a window is being rescaled.
- static char [show_next_window_iconic](#) ()
Returns the static flag whether the next window should be opened iconified.
- static void [show_next_window_iconic](#) (char stat)
Sets a static flag whether the next window should be opened iconified.

Static Public Member Functions inherited from FI_Group

- static [FI_Group](#) * [current](#) ()
Returns the currently active group.
- static void [current](#) ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from FI_Widget

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from [FL_Window](#)

- void [default_size_range](#) ()
Protected method to calculate the default size range of a window.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- int [force_position](#) () const
Returns the internal state of the window's FORCE_POSITION flag.
- void [force_position](#) (int force)
Sets an internal flag that tells FLTK and the window manager to honor position requests.
- void [free_icons](#) ()
Deletes all icons previously attached to the window.
- int [is_resizable](#) ()
Protected method to determine whether a window is resizable.

Protected Member Functions inherited from [FL_Group](#)

- [FL_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw_child](#) ([FL_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [FL_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) ([FL_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int [on_move](#) (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void [on_remove](#) (int)
Allow derived groups to act when a child widget is removed from the group.
- int * [sizes](#) ()
Returns the internal array of widget sizes and positions.
- void [update_child](#) ([FL_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FL_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FL_Boxtype](#) t, [FL_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FL_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.

- void `draw_focus` (`Fl_Boxtype` t, int x, int y, int w, int h, `Fl_Color` bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `Fl_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Static Protected Attributes inherited from `Fl_Window`

- static `Fl_Window` * `current_`
Stores the last window that was made current.

11.32.1 Detailed Description

The `Fl_Double_Window` provides a double-buffered window.

It will draw the window data into an off-screen pixmap, and then copy it to the on-screen window.

11.32.2 Constructor & Destructor Documentation

11.32.2.1 `~Fl_Double_Window()`

```
Fl_Double_Window::~~Fl_Double_Window ()
```

The destructor *also deletes all the children*.

This allows a whole tree to be deleted at once, without having to keep a pointer to all the children in the user code.

11.32.3 Member Function Documentation

11.32.3.1 `as_double_window()`

```
Fl_Double_Window * Fl_Double_Window::as_double_window () [inline], [virtual]
```

Return non-null if this is an `Fl_Double_Window` object.

Reimplemented from `Fl_Window`.

11.32.3.2 `flush()`

```
void Fl_Double_Window::flush () [virtual]
```

Forces the window to be drawn, this window is also made current and calls `draw()`.

Reimplemented from `Fl_Window`.

Reimplemented in `Fl_Overlay_Window`.

11.32.3.3 `hide()`

```
void Fl_Double_Window::hide () [virtual]
```

Makes a widget invisible.

See also

[show\(\)](#), [visible\(\)](#), [visible_r\(\)](#)

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Overlay_Window](#).

11.32.3.4 `resize()`

```
void Fl_Double_Window::resize (
    int x,
    int y,
    int w,
    int h) [virtual]
```

Changes the size or position of the widget.

This is a virtual function so that the widget may implement its own handling of resizing. The default version does *not* call the [redraw\(\)](#) method, but instead relies on the parent widget to do so because the parent may know a faster way to update the display, such as scrolling from the old position.

Some window managers under X11 call [resize\(\)](#) a lot more often than needed. Please verify that the position or size of a widget did actually change before doing any extensive calculations.

`position(X, Y)` is a shortcut for `resize(X, Y, w(), h())`, and `size(W, H)` is a shortcut for `resize(x(), y(), W, H)`.

Parameters

in	<i>x,y</i>	new position relative to the parent window
in	<i>w,h</i>	new size

See also

[position\(int,int\)](#), [size\(int,int\)](#)

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Overlay_Window](#).

11.32.3.5 `show()`

```
void Fl_Double_Window::show () [virtual]
```

Makes a widget visible.

An invisible widget never gets redrawn and does not get keyboard or mouse events, but can receive a few other events like `FL_SHOW`.

The [visible\(\)](#) method returns true if the widget is set to be visible. The [visible_r\(\)](#) method returns true if the widget and all of its parents are visible. A widget is only visible if [visible\(\)](#) is true on it *and all of its parents*.

Changing it will send `FL_SHOW` or `FL_HIDE` events to the widget. *Do not change it if the parent is not visible, as this will send false `FL_SHOW` or `FL_HIDE` events to the widget.* [redraw\(\)](#) is called if necessary on this or the parent.

See also

[hide\(\)](#), [visible\(\)](#), [visible_r\(\)](#)

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Overlay_Window](#).

The documentation for this class was generated from the following files:

- `Fl_Double_Window.H`
- [Fl_Double_Window.cxx](#)

11.33 Fl_End Class Reference

This is a dummy class that allows you to end a [Fl_Group](#) in a constructor list of a class:

```
#include <Fl_Group.H>
```

Public Member Functions

- [Fl_End](#) ()

All it does is calling [Fl_Group::current\(\)->end\(\)](#)

11.33.1 Detailed Description

This is a dummy class that allows you to end a [Fl_Group](#) in a constructor list of a class:

```
class MyClass {
    Fl_Group group;
    Fl_Button button_in_group;
    Fl_End end;
    Fl_Button button_outside_group;
    MyClass();
};

MyClass::MyClass() :
    group(10,10,100,100),
    button_in_group(20,20,60,30),
    end(),
    button_outside_group(10,120,60,30) {
    [...ctor code...]
```

The documentation for this class was generated from the following file:

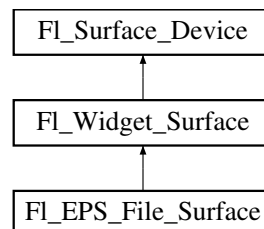
- [Fl_Group.H](#)

11.34 Fl_EPS_File_Surface Class Reference

Encapsulated PostScript drawing surface.

```
#include <Fl_PostScript.H>
```

Inheritance diagram for [Fl_EPS_File_Surface](#):



Public Member Functions

- int [close](#) ()
Completes all EPS output.
- FILE * [file](#) ()
Returns the underlying FILE pointer.
- [Fl_EPS_File_Surface](#) (int width, int height, FILE *eps_output, [Fl_Color](#) background=FL_WHITE, [Fl_PostScript_Close_Command](#) closef=NULL)
Constructor.
- void [origin](#) (int *px, int *py) [FL_OVERRIDE](#)
Computes the coordinates of the current origin of graphics functions.
- void [origin](#) (int x, int y) [FL_OVERRIDE](#)
Sets the position of the origin of graphics in the drawable part of the drawing surface.
- int [printable_rect](#) (int *w, int *h) [FL_OVERRIDE](#)
Computes the width and height of the drawable area of the drawing surface.

- void [translate](#) (int x, int y) [FL_OVERRIDE](#)
Translates the current graphics origin accounting for the current rotation.
- void [untranslate](#) () [FL_OVERRIDE](#)
Undoes the effect of a previous [translate\(\)](#) call.
- [~FI_EPS_File_Surface](#) ()
Destructor.

Public Member Functions inherited from [FI_Widget_Surface](#)

- void [draw](#) ([FI_Widget](#) *widget, int delta_x=0, int delta_y=0)
Draws the widget on the drawing surface.
- void [draw_decorated_window](#) ([FI_Window](#) *win, int x_offset=0, int y_offset=0)
Draws a window with its title bar and frame if any.
- void [print_window_part](#) ([FI_Window](#) *win, int x, int y, int w, int h, int delta_x=0, int delta_y=0)
Draws a rectangular part of an on-screen window.

Public Member Functions inherited from [FI_Surface_Device](#)

- [FI_Graphics_Driver](#) * **driver** ()
Returns the graphics driver of this drawing surface.
- virtual bool [is_current](#) ()
Is this surface the current drawing surface?
- virtual void [set_current](#) (void)
Make this surface the current drawing surface.
- virtual [~FI_Surface_Device](#) ()
The destructor.

Protected Member Functions

- [FI_PostScript_Graphics_Driver](#) * **driver** ()
Returns the PostScript driver of this drawing surface.

Protected Member Functions inherited from [FI_Widget_Surface](#)

- [FI_Widget_Surface](#) ([FI_Graphics_Driver](#) *d)
The constructor.

Protected Member Functions inherited from [FI_Surface_Device](#)

- void **driver** ([FI_Graphics_Driver](#) *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void [end_current](#) ()
FLTK calls this each time a surface ceases to be the current drawing surface.
- **FI_Surface_Device** ([FI_Graphics_Driver](#) *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Surface_Device](#)

- static [FI_Surface_Device](#) * [pop_current](#) ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void [push_current](#) ([FI_Surface_Device](#) *new_current)
*Pushes *new_current* on top of the stack of current drawing surfaces, and makes it current.*
- static [FI_Surface_Device](#) * [surface](#) ()
The current drawing surface.

Protected Attributes inherited from [Fl_Widget_Surface](#)

- `int x_offset`
horizontal offset to the origin of graphics coordinates
- `int y_offset`
vertical offset to the origin of graphics coordinates

11.34.1 Detailed Description

Encapsulated PostScript drawing surface.

This drawing surface allows to store any FLTK graphics in vectorial form in an "Encapsulated PostScript" file.

Usage example:

```
Fl_Window *win = ...// Window to draw to an .eps file
int ww = win->decorated_w();
int wh = win->decorated_h();
FILE *eps = fl_fopen("/path/to/mywindow.eps", "w");
if (eps) {
    Fl_EPS_File_Surface *surface = new Fl_EPS_File_Surface(ww, wh, eps, win->color());
    Fl_Surface_Device::push_current(surface);
    surface->draw_decorated_window(win);
    Fl_Surface_Device::pop_current();
    delete surface; // the .eps file is not complete until the destructor was run
}
```

11.34.2 Constructor & Destructor Documentation

11.34.2.1 Fl_EPS_File_Surface()

```
Fl_EPS_File_Surface::Fl_EPS_File_Surface (
    int width,
    int height,
    FILE * eps_output,
    Fl_Color background = FL_WHITE,
    Fl_PostScript_Close_Command closef = NULL)
```

Constructor.

Parameters

<i>width,height</i>	Width and height of the EPS drawing area
<i>eps_output</i>	A writable FILE pointer where the Encapsulated PostScript data will be sent
<i>background</i>	Color expected to cover the background of the EPS drawing area. This parameter affects only the drawing of transparent Fl_RGB_Image objects: transparent areas of RGB images are blended with the <i>background</i> color. Under the X11 + pango platform, transparent RGB images are correctly blended to their background, thus this parameter has no effect.
<i>closef</i>	If not NULL, the destructor or close() will call <code>closef(eps_output)</code> after all EPS data has been sent. If NULL, <code>fclose(eps_output)</code> is called instead. This allows to close the FILE pointer by, e.g., <code>pclose</code> , or, using a function such as <code>"int keep_open(FILE*) {return 0;}"</code> , to keep it open after completion of all output to <i>eps_output</i> . Function <i>closef</i> should return non zero to indicate an error.

11.34.2.2 ~Fl_EPS_File_Surface()

```
Fl_EPS_File_Surface::~~Fl_EPS_File_Surface ()
```

Destructor.

By default, the destructor closes with function `fclose()` the underlying FILE. See the constructor for how to close it differently or to keep it open. Use [close\(\)](#) before object destruction to receive the status code of output operations. If [close\(\)](#) is not used and if EPS output results in error, the destructor displays an alert message with `fl_alert()`.

11.34.3 Member Function Documentation

11.34.3.1 close()

```
int Fl_EPS_File_Surface::close ()
```

Completes all EPS output.

The only operation possible with the [Fl_EPS_File_Surface](#) object after calling [close\(\)](#) is its destruction.

Returns

The status code of output operations to the FILE object. 0 indicates success.

11.34.3.2 origin() [1/2]

```
void Fl_EPS_File_Surface::origin (
    int * x,
    int * y) [virtual]
```

Computes the coordinates of the current origin of graphics functions.

Parameters

out	x,y	If non-null, *x and *y are set to the horizontal and vertical coordinates of the graphics origin.

Reimplemented from [Fl_Widget_Surface](#).

11.34.3.3 origin() [2/2]

```
void Fl_EPS_File_Surface::origin (
    int x,
    int y) [virtual]
```

Sets the position of the origin of graphics in the drawable part of the drawing surface.

Arguments should be expressed relatively to the result of a previous [printable_rect\(\)](#) call. That is, `printable_rect(&w, &h); origin(w/2, 0);` sets the graphics origin at the top center of the drawable area. Successive [origin\(\)](#) calls don't combine their effects. [Origin\(\)](#) calls are not affected by [rotate\(\)](#) calls (for classes derived from [Fl_Paged_Device](#)).

Parameters

in	x,y	Horizontal and vertical positions in the drawing surface of the desired origin of graphics.

Reimplemented from [Fl_Widget_Surface](#).

11.34.3.4 printable_rect()

```
int Fl_EPS_File_Surface::printable_rect (
    int * w,
    int * h) [virtual]
```

Computes the width and height of the drawable area of the drawing surface.

Values are in the same unit as that used by FLTK drawing functions and are unchanged by calls to [origin\(\)](#). If the object is derived from class [Fl_Paged_Device](#), values account for the user-selected paper type and print orientation and are changed by [scale\(\)](#) calls.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Widget_Surface](#).

11.34.3.5 translate()

```
void Fl_EPS_File_Surface::translate (
    int x,
    int y) [virtual]
```

Translates the current graphics origin accounting for the current rotation.

Each [translate\(\)](#) call must be matched by an [untranslate\(\)](#) call. Successive [translate\(\)](#) calls add up their effects.

Reimplemented from [Fl_Widget_Surface](#).

11.34.3.6 untranslate()

```
void Fl_EPS_File_Surface::untranslate (
    void ) [virtual]
```

Undoes the effect of a previous [translate\(\)](#) call.

Reimplemented from [Fl_Widget_Surface](#).

The documentation for this class was generated from the following file:

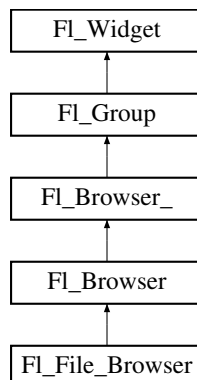
- [Fl_PostScript.H](#)

11.35 Fl_File_Browser Class Reference

The [Fl_File_Browser](#) widget displays a list of filenames, optionally with file-specific icons.

```
#include <Fl_File_Browser.H>
```

Inheritance diagram for [Fl_File_Browser](#):



Public Types

- enum { **FILES** , **DIRECTORIES** }

Public Types inherited from [Fl_Browser](#)

- enum [Fl_Line_Position](#) { **TOP** , **BOTTOM** , **MIDDLE** }
- For internal use only?*

Public Types inherited from [Fl_Browser_](#)

- enum {
[HORIZONTAL](#) = 1 , [VERTICAL](#) = 2 , [BOTH](#) = 3 , [ALWAYS_ON](#) = 4 ,
[HORIZONTAL_ALWAYS](#) = 5 , [VERTICAL_ALWAYS](#) = 6 , [BOTH_ALWAYS](#) = 7 }
Values for [has_scrollbar\(\)](#).

Public Member Functions

- const char * [errmsg](#) () const

- Returns OS error messages, or NULL if none.*

 - void **errmsg** (const char *errmsg)
 - Sets OS error message to a string, which can be NULL.*
- int **filetype** () const
 - Sets or gets the file browser type, FILES or DIRECTORIES.*
- void **filetype** (int t)
 - Sets or gets the file browser type, FILES or DIRECTORIES.*
- const char * **filter** () const
 - Sets or gets the filename filter.*
- void **filter** (const char *pattern)
 - Sets or gets the filename filter.*
- **FI_File_Browser** (int, int, int, int, const char *=0)
 - The constructor creates the **FI_File_Browser** widget at the specified position and size.*
- uchar **iconsize** () const
 - Sets or gets the size of the icons.*
- void **iconsize** (uchar s)
 - Sets or gets the size of the icons.*
- int **load** (const char *directory, **FI_File_Sort_F** *sort=fl_numericsort)
 - Loads the specified directory into the browser.*
- **FI_Fontsize** **textsize** () const
- void **textsize** (**FI_Fontsize** s)

Public Member Functions inherited from **FI_Browser**

- void **add** (const char *newtext, void *d=0)
 - Adds a new line to the end of the browser.*
- void **bottomline** (int line)
 - Scrolls the browser so the bottom item in the browser is showing the specified *line*.*
- void **clear** ()
 - Removes all the lines in the browser.*
- char **column_char** () const
 - Gets the current column separator character.*
- void **column_char** (char c)
 - Sets the column separator to *c*.*
- const int * **column_widths** () const
 - Gets the current column width array.*
- void **column_widths** (const int *arr)
 - Sets the current array to *arr*.*
- void * **data** (int line) const
 - Returns the user **data()** for specified *line*.*
- void **data** (int line, void *d)
 - Sets the user data for specified *line* to *d*.*
- void **display** (int line, int val=1)
 - For back compatibility.*
- int **displayed** (int line) const
 - Returns non-zero if *line* has been scrolled to a position where it is being displayed.*
- **FI_Browser** (int X, int Y, int W, int H, const char *L=0)
 - The constructor makes an empty browser.*
- char **format_char** () const
 - Gets the current format code prefix character, which by default is '@'.*
- void **format_char** (char c)

- Sets the current format code prefix character to `c`.*
- void **hide** () **FL_OVERRIDE**
 - Hides the entire `Fl_Browser` widget – opposite of `show()`.*
- void **hide** (int line)
 - Makes `line` invisible, preventing selection by the user.*
- **Fl_Image** * **icon** (int line) const
 - Returns the icon currently defined for `line`.*
- void **icon** (int line, **Fl_Image** *icon)
 - Set the image icon for `line` to the value `icon`.*
- void **insert** (int line, const char *newtext, void *d=0)
 - Insert a new entry whose label is `newtext` above given `line`, optional data `d`.*
- void **lineposition** (int line, **Fl_Line_Position** pos)
 - Updates the browser so that `line` is shown at position `pos`.*
- int **load** (const char *filename)
 - Clears the browser and reads the file, adding each line from the file to the browser.*
- void **make_visible** (int line)
 - Make the item at the specified `line` **visible()**.*
- void **middleline** (int line)
 - Scrolls the browser so the middle item in the browser is showing the specified `line`.*
- void **move** (int to, int from)
 - Line `from` is removed and reinserted at `to`.*
- void **remove** (int line)
 - Remove entry for given `line` number, making the browser one line shorter.*
- void **remove_icon** (int line)
 - Removes the icon for `line`.*
- void **replace** (int a, const char *b)
 - For back compatibility only.*
- int **select** (int line, int val=1)
 - Sets the selection state of the item at `line` to the value `val`.*
- int **selected** (int line) const
 - Returns 1 if specified `line` is selected, 0 if not.*
- void **show** () **FL_OVERRIDE**
 - Shows the entire `Fl_Browser` widget – opposite of `hide()`.*
- void **show** (int line)
 - Makes `line` visible, and available for selection by user.*
- int **size** () const
 - Returns how many lines are in the browser.*
- void **size** (int W, int H)
- void **swap** (int a, int b)
 - Swaps two browser lines `a` and `b`.*
- const char * **text** (int line) const
 - Returns the label text for the specified `line`.*
- void **text** (int line, const char *newtext)
 - Sets the text for the specified `line` to `newtext`.*
- **Fl_Fontsize** **textsize** () const
 - Gets the default text size (in pixels) for the lines in the browser.*
- void **textsize** (**Fl_Fontsize** newSize)
 - Sets the default text size (in pixels) for the lines in the browser to `newSize`.*
- int **topline** () const
 - Returns the line that is currently visible at the top of the browser.*
- void **topline** (int line)

- Scrolls the browser so the top item in the browser is showing the specified `line`.*

 - `int value () const`
Returns the line number of the currently selected line, or 0 if none selected.
 - `void value (int line)`
Sets the browser's `value()`, which selects the specified `line`.
 - `int visible (int line) const`
Returns non-zero if the specified `line` is visible, 0 if hidden.
 - `~FI_Browser ()`
The destructor deletes all list items and destroys the browser.

Public Member Functions inherited from `FI_Browser_`

- `int deselect (int docallbacks=0)`
Deselects all items in the list and returns 1 if the state changed or 0 if it did not.
- `void display (void *item)`
Displays the `item`, scrolling the list as necessary.
- `int handle (int event) FL_OVERRIDE`
Handles the `event` within the normal widget bounding box.
- `uchar has_scrollbar () const`
Returns the current scrollbar mode, see `FI_Browser_::has_scrollbar(uchar)`
- `void has_scrollbar (uchar mode)`
Sets whether the widget should have scrollbars or not (default `FI_Browser_::BOTH`).
- `int hposition () const`
Gets the horizontal scroll position of the list as a pixel position `pos`.
- `void hposition (int)`
Sets the horizontal scroll position of the list to pixel position `pos`.
- `int linespacing () const`
Return the height of additional spacing between browser lines.
- `void linespacing (int pixels)`
Add some space between browser lines.
- `int position () const`
- `void position (int pos)`
- `void position (int x, int y)`
- `void resize (int X, int Y, int W, int H) FL_OVERRIDE`
Repositions and/or resizes the browser.
- `void scrollbar_left ()`
Moves the vertical scrollbar to the lefthand side of the list.
- `void scrollbar_right ()`
Moves the vertical scrollbar to the righthand side of the list.
- `int scrollbar_size () const`
Gets the current size of the scrollbars' troughs, in pixels.
- `void scrollbar_size (int newSize)`
Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.
- `int scrollbar_width () const`
Returns the global value `FI::scrollbar_size()`.
- `void scrollbar_width (int width)`
Sets the global `FI::scrollbar_size()`, and forces this instance of the widget to use it.
- `int select (void *item, int val=1, int docallbacks=0)`
Sets the selection state of `item` to `val`, and returns 1 if the state changed or 0 if it did not.
- `int select_only (void *item, int docallbacks=0)`
Selects `item` and returns 1 if the state changed or 0 if it did not.

- void **sort** (int *flags*=0)
Sort the items in the browser based on flags.
- **FL_Color** **textcolor** () const
Gets the default text color for the lines in the browser.
- void **textcolor** (**FL_Color** *col*)
Sets the default text color for the lines in the browser to color col.
- **FL_Font** **textfont** () const
Gets the default text font for the lines in the browser.
- void **textfont** (**FL_Font** *font*)
Sets the default text font for the lines in the browser to font.
- **FL_Fontsize** **textsize** () const
Gets the default text size (in pixels) for the lines in the browser.
- void **textsize** (**FL_Fontsize** *newSize*)
Sets the default text size (in pixels) for the lines in the browser to size.
- int **vposition** () const
Gets the vertical scroll position of the list as a pixel position pos.
- void **vposition** (int *pos*)
Sets the vertical scroll position of the list to pixel position pos.

Public Member Functions inherited from **FL_Group**

- **FL_Widget** ***&_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** (**FL_Widget** &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** (**FL_Widget** *o)
See void FL_Group::add(FL_Widget &w)
- void **add_resizable** (**FL_Widget** &o)
Adds a widget to the group and makes it the resizable widget.
- **FL_Widget** *const * **array** () const
Returns a pointer to the internal array of children.
- **FL_Group** const * **as_group** () const **FL_OVERRIDE**
- **FL_Group** * **as_group** () **FL_OVERRIDE**
Returns an FL_Group pointer if this widget is an FL_Group.
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- **FL_Widget** * **child** (int *n*) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int *c*)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int *n*)
Removes the widget at index from the group and deletes it.
- void **end** ()
Exactly the same as current(this->parent()).
- int **find** (const **FL_Widget** &o) const

- See `int FL_Group::find(const FL_Widget *w) const`.
- `int find (const FL_Widget *) const`
Searches the children for the widget and returns the index.
- `FL_Group (int, int, int, int, const char * = 0)`
Creates a new FL_Group widget using the given position, size, and label string.
- `void focus (FL_Widget *W)`
- `void forms_end ()`
This is for forms compatibility only.
- `int handle (int) FL_OVERRIDE`
Handles the specified event.
- `void init_sizes ()`
Resets the internal array of widget sizes and positions.
- `void insert (FL_Widget &, int i)`
The widget is removed from its current group (if any) and then inserted into this group.
- `void insert (FL_Widget &o, FL_Widget *before)`
This does insert(w, find(before)).
- `void remove (FL_Widget &)`
Removes a widget from the group but does not delete it.
- `void remove (FL_Widget *o)`
Removes the widget o from the group.
- `void remove (int index)`
Removes the widget at index from the group but does not delete it.
- `FL_Widget * resizable () const`
Returns the group's resizable widget.
- `void resizable (FL_Widget &o)`
Sets the group's resizable widget.
- `void resizable (FL_Widget *o)`
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- `void resize (int, int, int, int) FL_OVERRIDE`
Resizes the FL_Group widget and all of its children.
- `virtual ~FL_Group ()`
The destructor also deletes all the children.

Public Member Functions inherited from FL_Widget

- `void _clear_fullscreen ()`
- `void _set_fullscreen ()`
- `void activate ()`
Activates the widget.
- `unsigned int active () const`
Returns whether the widget is active.
- `int active_r () const`
Returns whether the widget and all of its parents are active.
- `FL_Align align () const`
Gets the label alignment.
- `void align (FL_Align alignment)`
Sets the label alignment.
- `long argument () const`
Gets the current user data (long) argument that is passed to the callback function.
- `void argument (long v)`
Sets the current user data (long) argument that is passed to the callback function.

- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color](#) [color2](#) () const

- For back compatibility only.*

 - void `color2` (unsigned a)
- For back compatibility only.*

 - int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)

Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const

Gets the widget height.
- int `horizontal_label_margin` ()

Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)

Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()

Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const

Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the active state.

- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const
Gets the label type.
- void `labeltype` (`FL_Labeltype` a)
Sets the label type.
- void `measure_label` (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const
Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int `output` () const
Returns if a widget is used for output only.
- `FL_Group` * `parent` () const
Returns a pointer to the parent widget.
- void `parent` (`FL_Group` *p)
Internal use only - "for hacks only".
- void `position` (int X, int Y)
Repositions the window or widget.
- void `redraw` ()
Schedules the drawing of the widget.
- void `redraw_label` ()
Schedules the drawing of the label.
- `FL_Color` `selection_color` () const

- Gets the selection color.*

 - void **selection_color** (**Fl_Color** a)

Sets the selection color.
- void **set_active** ()

Marks the widget as active without sending events or changing focus.
- void **set_changed** ()

Marks the value of the widget as changed.
- void **set_output** ()

Sets a widget to output only.
- void **set_visible** ()

Makes the widget visible.
- void **set_visible_focus** ()

Enables keyboard focus navigation with this widget.
- int **shortcut_label** () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void **shortcut_label** (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- void **size** (int W, int H)

Changes the size of the widget.
- int **take_focus** ()

Gives the widget the keyboard focus.
- unsigned int **takeevents** () const

Returns if the widget is able to take events.
- int **test_shortcut** ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * **tooltip** () const

Gets the current tooltip text.
- void **tooltip** (const char *text)

Sets the current tooltip text.
- **Fl_Window** * **top_window** () const

Returns a pointer to the top-level window for the widget.
- **Fl_Window** * **top_window_offset** (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- **uchar** **type** () const

Gets the widget type.
- void **type** (**uchar** t)

Sets the widget type.
- int **use_accents_menu** ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const

Gets the user data for this widget.
- void **user_data** (**Fl_Callback_User_Data** *v, bool auto_free)

Sets the user data for this widget.
- void **user_data** (void *v)

Sets the user data for this widget.
- int **vertical_label_margin** ()

Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const

Returns whether a widget is visible.

- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [FI_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) (uchar i)
Sets the flags used to decide when a callback is called.
- [FI_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual [~FI_Widget](#) ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * [current](#) ()
Returns the currently active group.
- static void [current](#) ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [FI_Widget](#)

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Public Attributes inherited from [FI_Browser_](#)

- [FI_Scrollbar](#) [hscrollbar](#)
Horizontal scrollbar.
- [FI_Scrollbar](#) [scrollbar](#)
Vertical scrollbar.

Protected Types inherited from [FL_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from [FL_Browser](#)

- FL_BLINE * [_remove](#) (int line)
Removes the item at the specified line.
- FL_BLINE * [find_line](#) (int line) const
Returns the item for specified line.
- void [insert](#) (int line, FL_BLINE *item)
Insert specified item above line.
- void * [item_at](#) (int line) const [FL_OVERRIDE](#)
Return the item at specified line.
- void * [item_first](#) () const [FL_OVERRIDE](#)
Returns the very first item in the list.
- void * [item_last](#) () const [FL_OVERRIDE](#)
Returns the very last item in the list.
- void * [item_next](#) (void *item) const [FL_OVERRIDE](#)
Returns the next item after item.
- void * [item_prev](#) (void *item) const [FL_OVERRIDE](#)
Returns the previous item before item.
- void [item_select](#) (void *item, int val) [FL_OVERRIDE](#)
Change the selection state of item to the value val.
- int [item_selected](#) (void *item) const [FL_OVERRIDE](#)
See if item is selected.
- void [item_swap](#) (void *a, void *b) [FL_OVERRIDE](#)
Swap the items a and b.
- const char * [item_text](#) (void *item) const [FL_OVERRIDE](#)
Returns the label text for item.
- int [lineno](#) (void *item) const
Returns line number corresponding to item, or zero if not found.
- void [swap](#) (FL_BLINE *a, FL_BLINE *b)
Swap the two items a and b.

Protected Member Functions inherited from FI_Browser_

- void **bbox** (int &X, int &Y, int &W, int &H) const
Returns the bounding box for the interior of the list's display window, inside the scrollbars.
- void **deleting** (void *item)
*This method should be used when *item* is being deleted from the list.*
- int **displayed** (void *item) const
*Returns non-zero if *item* has been scrolled to a position where it is being displayed.*
- void **draw** () **FL_OVERRIDE**
Draws the list within the normal widget bounding box.
- void * **find_item** (int ypos)
*This method returns the item under mouse y position *ypos*.*
- **FI_Browser_** (int X, int Y, int W, int H, const char *L=0)
The constructor makes an empty browser.
- virtual int **full_width** () const
This method may be provided by the subclass to indicate the full width of the item list, in pixels.
- void **inserting** (void *a, void *b)
This method should be used when an item is in the process of being inserted into the list.
- virtual int **item_quick_height** (void *item) const
*This method may be provided by the subclass to return the height of the *item*, in pixels.*
- int **leftedge** () const
This method returns the X position of the left edge of the list area after adjusting for the scrollbar and border, if any.
- void **new_list** ()
This method should be called when the list data is completely replaced or cleared.
- void **redraw_line** (void *item)
*This method should be called when the contents of *item* has changed, but not its height.*
- void **redraw_lines** ()
This method will cause the entire list to be redrawn.
- void **replacing** (void *a, void *b)
*This method should be used when item *a* is being replaced by item *b*.*
- void * **selection** () const
Returns the item currently selected, or NULL if there is no selection.
- void **swapping** (void *a, void *b)
*This method should be used when two items *a* and *b* are being swapped.*
- void * **top** () const
Returns the item that appears at the top of the list.

Protected Member Functions inherited from FI_Group

- **FI_Rect** * **bounds** ()
Returns the internal array of widget sizes and positions.
- void **draw** () **FL_OVERRIDE**
Draws the widget.
- void **draw_child** (**FI_Widget** &widget) const
Forces a child to redraw.
- void **draw_children** ()
Draws all children of the group.
- void **draw_outside_label** (const **FI_Widget** &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int **on_insert** (**FI_Widget** *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int **on_move** (int, int)

- *Allow derived groups to act when a widget is moved within the group.*
- virtual void **on_remove** (int)
Allow derived groups to act when a child widget is removed from the group.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** (FI_Widget &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.35.1 Detailed Description

The [FI_File_Browser](#) widget displays a list of filenames, optionally with file-specific icons.

11.35.2 Constructor & Destructor Documentation

11.35.2.1 Fl_File_Browser()

```
Fl_File_Browser::Fl_File_Browser (
    int X,
    int Y,
    int W,
    int H,
    const char * I = 0)
```

The constructor creates the [Fl_File_Browser](#) widget at the specified position and size.
The destructor destroys the widget and frees all memory that has been allocated.

11.35.3 Member Function Documentation

11.35.3.1 errmsg() [1/2]

```
const char * Fl_File_Browser::errmsg () const [inline]
```

Returns OS error messages, or NULL if none.
Use when advised.

11.35.3.2 errmsg() [2/2]

```
void Fl_File_Browser::errmsg (
    const char * emsg)
```

Sets OS error message to a string, which can be NULL.
Frees previous if any. void [errmsg\(const char *emsg\)](#);

11.35.3.3 filetype() [1/2]

```
int Fl_File_Browser::filetype () const [inline]
```

Sets or gets the file browser type, FILES or DIRECTORIES.
When set to FILES, both files and directories are shown. Otherwise only directories are shown.

11.35.3.4 filetype() [2/2]

```
void Fl_File_Browser::filetype (
    int t) [inline]
```

Sets or gets the file browser type, FILES or DIRECTORIES.
When set to FILES, both files and directories are shown. Otherwise only directories are shown.

11.35.3.5 filter() [1/2]

```
const char * Fl_File_Browser::filter () const [inline]
```

Sets or gets the filename filter.
The pattern matching uses the [fl_filename_match\(\)](#) function in FLTK.

11.35.3.6 filter() [2/2]

```
void Fl_File_Browser::filter (
    const char * pattern)
```

Sets or gets the filename filter.
The pattern matching uses the [fl_filename_match\(\)](#) function in FLTK.

11.35.3.7 iconsize() [1/2]

```
uchar Fl_File_Browser::iconsize () const [inline]
```

Sets or gets the size of the icons.
The default size is 20 pixels.

11.35.3.8 iconsize() [2/2]

```
void Fl_File_Browser::iconsize (
    uchar s) [inline]
```

Sets or gets the size of the icons.

The default size is 20 pixels.

11.35.3.9 load()

```
int Fl_File_Browser::load (
    const char * directory,
    Fl_File_Sort_F * sort = fl_numericsort)
```

Loads the specified directory into the browser.

If icons have been loaded then the correct icon is associated with each file in the list.

If directory is "", all mount points (unix) or drive letters (Windows) are listed.

The sort argument specifies a sort function to be used with [fl_filename_list\(\)](#).

Return value is the number of filename entries, or 0 if none. On error, 0 is returned, and [errmsg\(\)](#) has OS error string if non-NULL.

The documentation for this class was generated from the following files:

- [Fl_File_Browser.H](#)
- [Fl_File_Browser.cxx](#)

11.36 FI_File_Chooser Class Reference

The [Fl_File_Chooser](#) widget displays a standard file selection dialog that supports various selection modes.

Public Types

- enum [Type](#) { [SINGLE](#) = 0 , [MULTI](#) = 1 , [CREATE](#) = 2 , [DIRECTORY](#) = 4 }

Determines the type of file chooser presented to the user.

Public Member Functions

- [Fl_Widget](#) * [add_extra](#) ([Fl_Widget](#) *gr)
Adds an extra widget at the bottom of the [Fl_File_Chooser](#) window.
- void [callback](#) (void(*cb)([Fl_File_Chooser](#) *, void *), void *d=0)
Sets the file chooser callback cb and associated data d.
- [Fl_Color](#) [color](#) ()
Gets the background color of the [Fl_File_Browser](#) list.
- void [color](#) ([Fl_Color](#) c)
Sets the background color of the [Fl_File_Browser](#) list.
- int [count](#) ()
Returns the number of selected files.
- char * [directory](#) ()
Gets the current directory.
- void [directory](#) (const char *d)
Sets the current directory.
- const char * [filter](#) ()
Gets the current filename filter patterns.
- void [filter](#) (const char *p)
Sets the current filename filter patterns.
- int [filter_value](#) ()
Gets the current filename filter selection.

- void **filter_value** (int f)
Sets the current filename filter selection.
- **FL_File_Chooser** (const char *pathname, const char *pattern, int type_val, const char *title)
The constructor creates the [FL_File_Chooser](#) dialog shown.
- int **h** () const
- void **hide** ()
Hides the [FL_File_Chooser](#) window.
- uchar **iconsize** ()
Gets the size of the icons in the [FL_File_Browser](#).
- void **iconsize** (uchar s)
Sets the size of the icons in the [FL_File_Browser](#).
- const char * **label** ()
Gets the title bar text for the [FL_File_Chooser](#).
- void **label** (const char *l)
Sets the title bar text for the [FL_File_Chooser](#).
- const char * **ok_label** ()
Gets the label for the "ok" button in the [FL_File_Chooser](#).
- void **ok_label** (const char *l)
Sets the label for the "ok" button in the [FL_File_Chooser](#).
- void **position** (int x, int y)
- int **preview** () const
Returns the current state of the preview box.
- void **preview** (int e)
Enable or disable the preview tile.
- void **rescan** ()
Reloads the current directory in the [FL_File_Browser](#).
- void **rescan_keep_filename** ()
Rescan the current directory without clearing the filename, then select the file if it is in the list.
- void **resize** (int x, int y, int w, int h)
- void **show** ()
Shows the [FL_File_Chooser](#) window.
- int **shown** ()
Returns non-zero if the file chooser main window [show\(\)](#) has been called, but not [hide\(\)](#).
- void **size** (int w, int h)
- **FL_Color** **textcolor** ()
Gets the current [FL_File_Browser](#) text color.
- void **textcolor** (**FL_Color** c)
Sets the current [FL_File_Browser](#) text color.
- **FL_Font** **textfont** ()
Gets the current [FL_File_Browser](#) text font.
- void **textfont** (**FL_Font** f)
Sets the current [FL_File_Browser](#) text font.
- **FL_Fontsize** **textsize** ()
Gets the current [FL_File_Browser](#) text size.
- void **textsize** (**FL_Fontsize** s)
Sets the current [FL_File_Browser](#) text size.
- int **type** ()
Gets the current type of [FL_File_Chooser](#).
- void **type** (int t)
Sets the current type of [FL_File_Chooser](#).
- void * **user_data** () const

- Gets the file chooser user data.*
- void **user_data** (void *d)
Sets the file chooser user data d.
- void **value** (const char *filename)
Sets the current value of the selected file.
- const char * **value** (int f=1)
Gets the current value of the selected file(s).
- int **visible** ()
Returns 1 if the [FI_File_Chooser](#) window is visible.
- int **w** () const
- int **x** () const
- int **y** () const
- ~**FI_File_Chooser** ()
Destroys the widget and frees all memory used by it.

Public Attributes

- [FI_Button](#) * **newButton**
The "new directory" button is exported so that application developers can control the appearance and use.
- [FI_Check_Button](#) * **previewButton**
The "preview" button is exported so that application developers can control the appearance and use.
- [FI_Check_Button](#) * **showHiddenButton**
When checked, hidden files (i.e., filename begins with dot) are displayed.

Static Public Attributes

- static const char * **add_favorites_label** = "Add to Favorites"
[standard text may be customized at run-time]
- static const char * **all_files_label** = "All Files (*)"
[standard text may be customized at run-time]
- static const char * **custom_filter_label** = "Custom Filter"
[standard text may be customized at run-time]
- static const char * **existing_file_label** = "Please choose an existing file!"
[standard text may be customized at run-time]
- static const char * **favorites_label** = "Favorites"
[standard text may be customized at run-time]
- static const char * **filename_label** = "Filename:"
[standard text may be customized at run-time]
- static const char * **filesystems_label** = [FI::system_driver](#)()->filesystems_label()
[standard text may be customized at run-time]
- static const char * **hidden_label** = "Show hidden files"
[standard text may be customized at run-time]
- static const char * **manage_favorites_label** = "Manage Favorites"
[standard text may be customized at run-time]
- static const char * **new_directory_label** = "New Directory?"
[standard text may be customized at run-time]
- static const char * **new_directory_tooltip** = "Create a new directory."
[standard text may be customized at run-time]
- static const char * **preview_label** = "Preview"
[standard text may be customized at run-time]
- static const char * **save_label** = "Save"
[standard text may be customized at run-time]

- static const char * **show_label** = "Show:"
[standard text may be customized at run-time]
- static [FI_File_Sort_F](#) * **sort** = [fi_numericsort](#)
the sort function that is used when loading the contents of a directory.

Protected Member Functions

- void **show_error_box** (int val)
Show error box if val=1, hide if val=0.

Related Symbols

(Note that these are not member symbols.)

- char * [fi_dir_chooser](#) (const char *message, const char *fname, int relative)
Shows a file chooser dialog and gets a directory.
- char * [fi_file_chooser](#) (const char *message, const char *pat, const char *fname, int relative)
Shows a file chooser dialog and gets a filename.
- void [fi_file_chooser_callback](#) (void(*cb)(const char *))
Set the file chooser callback.
- void [fi_file_chooser_ok_label](#) (const char *l)
Set the "OK" button label.

11.36.1 Detailed Description

The [FI_File_Chooser](#) widget displays a standard file selection dialog that supports various selection modes.

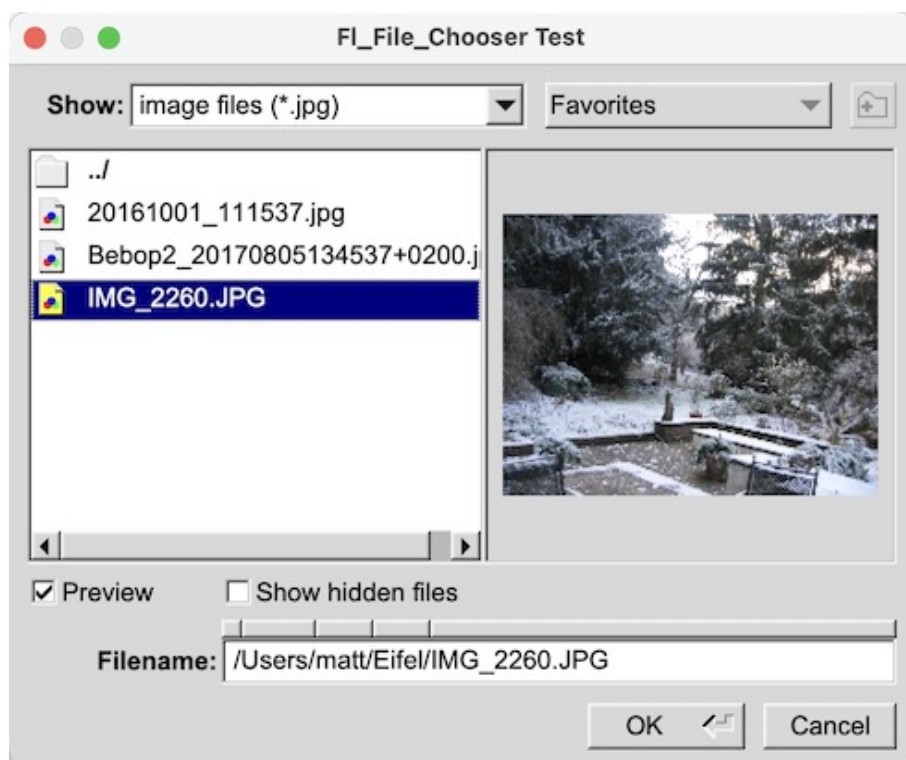


Figure 11.16 FI_File_Chooser

Features include:

- Multiple filter patterns can be specified, with parenthesis around filters, and tabs to separate each pattern, e.g.:

```
char pattern[] = "Image Files (*.{bmp,gif,jpg,png,xbm,xpm})\t"
                "Web Files (*.{htm,html,php})\t"
                "All Files (*)";
```

- If no "*" pattern is provided, then an entry for "All Files (*)" is automatically added.
- An optional file preview box is provided which can be toggled by programmer or user showing images, or the first 2048 bytes of printable text.
- Preview image loading functions can be registered to provide custom file previews.
- The favorites button shows up to 100 user-saved favorite directories, the user's home directory, and a filesystems item.
- A simple dialog is provided for managing saved directories.
- Shortcut keys are provided:

Shortcut	Description
Alt+a	Adds a directory to the favorites list
Alt+m	Manages the favorites list
Alt+f	Shows the filesystem list
Alt+h	Go to the home directory
Alt+0..9	going to any of the first 10 favorites

The [Fl_File_Chooser](#) widget transmits UTF-8 encoded filenames to its user. It is recommended to open files that may have non-ASCII names with the [fl_fopen\(\)](#) or [fl_open\(\)](#) utility functions that handle these names in a cross-platform way (whereas the standard `fopen()/open()` functions fail on the Windows platform to open files with a non-ASCII name).

The [Fl_File_Chooser](#) class also exports several static values that may be used to localize or customize the appearance of all file chooser dialogs:

Member	Default value
<code>add_favorites_label</code>	"Add to Favorites"
<code>all_files_label</code>	"All Files (*)"
<code>custom_filter_label</code>	"Custom Filter"
<code>existing_file_label</code>	"Please choose an existing file!"
<code>favorites_label</code>	"Favorites"
<code>filename_label</code>	"Filename:"
<code>filesystems_label</code>	"My Computer" (Windows) "File Systems" (all others)
<code>hidden_label</code>	"Show hidden files:"
<code>manage_favorites_label</code>	"Manage Favorites"
<code>new_directory_label</code>	"New Directory?"
<code>new_directory_tooltip</code>	"Create a new directory."
<code>preview_label</code>	"Preview"
<code>save_label</code>	"Save"
<code>show_label</code>	"Show:"
<code>sort</code>	<code>fl_numericsort</code>

The [Fl_File_Chooser::sort](#) member specifies the sort function that is used when loading the contents of a directory and can be customized at run-time.

The [Fl_File_Chooser](#) class also exports the [Fl_File_Chooser::newButton](#) and [Fl_File_Chooser::previewButton](#) widgets so that application developers can control their appearance and use.

11.36.2 Member Enumeration Documentation

11.36.2.1 Type

enum [Fl_File_Chooser::Type](#)

Determines the type of file chooser presented to the user.

Enumerator

SINGLE	Select a single, existing file.
MULTI	Select one or more existing files.
CREATE	When used alone, select a single, existing file or specify a new filename. Can be combined with DIRECTORY (e.g. CREATE DIRECTORY) to have the same effect for directories.
DIRECTORY	Select a single, existing directory. Can be combined with CREATE.

11.36.3 Constructor & Destructor Documentation

11.36.3.1 Fl_File_Chooser()

```
Fl_File_Chooser::Fl_File_Chooser (
    const char * pathname,
    const char * pattern,
    int type_val,
    const char * title)
```

The constructor creates the [Fl_File_Chooser](#) dialog shown.

- The `pathname` argument can be a directory name or a complete file name (in which case the corresponding file is highlighted in the list and in the filename input field.)
- The `pattern` argument can be a NULL string or "*" to list all files, or it can be a series of descriptions and filter strings separated by tab characters (\t). The format of filters is either "Description text (patterns)" or just "patterns". A file chooser that provides filters for HTML and image files might look like:

```
"HTML Files (*.html)\tImage Files (*.{bmp,gif,jpg,png})"
```

The file chooser will automatically add the "All Files (*)" pattern to the end of the string you pass if you do not provide one. The first filter in the string is the default filter. See the FLTK documentation on [fl_filename_match\(\)](#) for the kinds of pattern strings that are supported.
- The `type_val` argument can be one of the [Fl_File_Chooser::Type](#) values.
- The `title` argument is used to set the title bar text for the [Fl_File_Chooser](#) window.

11.36.4 Member Function Documentation

11.36.4.1 add_extra()

```
Fl_Widget * Fl_File_Chooser::add_extra (
    Fl_Widget * extra)
```

Adds an extra widget at the bottom of the [Fl_File_Chooser](#) window.

You can use any [Fl_Widget](#) or [Fl_Group](#). If you use an [Fl_Group](#), set its (x, y) coordinates to (0, 0) and position its children relative to (0, 0) inside the [Fl_Group](#) container widget. Make sure that all child widgets of the [Fl_Group](#) are entirely included inside the bounding box of their parents, i.e. the [Fl_Group](#) widget, and the [Fl_File_Chooser](#) window, respectively.

Note

The width of the [Fl_File_Chooser](#) window is an undocumented implementation detail and may change in the future.

If `extra` is NULL any previous extra widget is removed.

Parameters

<code>in</code>	<code>extra</code>	Custom widget or group to be added to the Fl_File_Chooser window.

Returns

Pointer to previous extra widget or NULL if not set previously.

Note

[Fl_File_Chooser](#) does **not** delete the extra widget in its destructor! The extra widget is removed from the [Fl_File_Chooser](#) window before the [Fl_File_Chooser](#) widget gets destroyed. To prevent memory leakage, don't forget to delete unused extra widgets.

11.36.4.2 filter()

```
void Fl_File_Chooser::filter (
    const char * p)
```

Sets the current filename filter patterns.

The filter patterns use [fl_filename_match\(\)](#). Multiple patterns can be used by separating them with tabs, like "*.jpg\t*.png\t*.gif\t*". In addition, you can provide human-readable labels with the patterns inside parenthesis, like "JPEG Files (*.jpg)\tPNG Files (*.png)\tGIF Files (*.gif)\tAll Files (*)" .

Use filter(NULL) to show all files.

11.36.4.3 iconsize() [1/2]

```
uchar Fl_File_Chooser::iconsize ()
```

Gets the size of the icons in the [Fl_File_Browser](#).

By default the icon size is set to 1.5 times the [textsize\(\)](#).

11.36.4.4 iconsize() [2/2]

```
void Fl_File_Chooser::iconsize (
    uchar s)
```

Sets the size of the icons in the [Fl_File_Browser](#).

By default the icon size is set to 1.5 times the [textsize\(\)](#).

11.36.4.5 preview()

```
void Fl_File_Chooser::preview (
    int e)
```

Enable or disable the preview tile.

1 = enable preview, 0 = disable preview.

11.36.4.6 shown()

```
int Fl_File_Chooser::shown ()
```

Returns non-zero if the file chooser main window [show\(\)](#) has been called, but not [hide\(\)](#).

See also

[Fl_Window::shown\(\)](#)

11.36.4.7 value() [1/2]

```
void Fl_File_Chooser::value (
    const char * filename)
```

Sets the current value of the selected file.

If a relative path is provided in `filename` it is converted to an absolute path.

If `NULL` or an empty string is provided, the working directory is changed to the user's current directory and the filename is set to "".

After assigning the filename the entire string (if any) is selected, i.e.

- `insert_position()` is 0 (zero)
- `mark()` is `strlen(<expanded filename>)`.

Note

The selection of the entire string may not always be desired but it is kept for backwards compatibility.

Parameters

in	<i>filename</i>	relative or absolute filename, may be <code>NULL</code> or ""

11.36.4.8 value() [2/2]

```
const char * Fl_File_Chooser::value (
    int f = 1)
```

Gets the current value of the selected file(s).

`f` is a 1-based index into a list of file names. The number of selected files is returned by `Fl_File_Chooser::count()`.

This sample code loops through all selected files:

```
// Get list of filenames user selected from a MULTI chooser
for ( int t=1; t<=chooser->count(); t++ ) {
    const char *filename = chooser->value(t);
    ...
}
```

11.36.5 Member Data Documentation**11.36.5.1 showHiddenButton**

`Fl_Check_Button*` `Fl_File_Chooser::showHiddenButton`

When checked, hidden files (i.e., filename begins with dot) are displayed.

The "showHiddenButton" button is exported so that application developers can control its appearance.

The documentation for this class was generated from the following files:

- `Fl_File_Chooser.H`
- `Fl_File_Chooser.cxx`
- `Fl_File_Chooser2.cxx`
- `fl_file_dir.cxx`

11.37 Fl_File_Icon Class Reference

The `Fl_File_Icon` class manages icon images that can be used as labels in other widgets and as icons in the `FileBrowser` widget.

```
#include <Fl_File_Icon.H>
```

Public Types

- enum {
 ANY , **PLAIN** , **FIFO** , **DEVICE** ,
 LINK , **DIRECTORY** }
- enum {
 END , **COLOR** , **LINE** , **CLOSEDLINE** ,
 POLYGON , **OUTLINEPOLYGON** , **VERTEX** }

Public Member Functions

- short * **add** (short d)
 Adds a keyword value to the icon array, returning a pointer to it.
- short * **add_color** (FI_Color c)
 Adds a color value to the icon array, returning a pointer to it.
- short * **add_vertex** (float x, float y)
 Adds a vertex value to the icon array, returning a pointer to it.
- short * **add_vertex** (int x, int y)
 Adds a vertex value to the icon array, returning a pointer to it.
- void **clear** ()
 Clears all icon data from the icon.
- void **draw** (int x, int y, int w, int h, FI_Color ic, int active=1)
 Draws an icon in the indicated area.
- FI_File_Icon (const char *p, int t, int nd=0, short *d=0)
 Creates a new FI_File_Icon with the specified information.
- void **label** (FI_Widget *w)
 Applies the icon to the widget, registering the FI_File_Icon label type as needed.
- void **load** (const char *f)
 Loads the specified icon image.
- int **load_fti** (const char *fti)
 Loads an SGI icon file.
- int **load_image** (const char *i)
 Load an image icon file from an image filename.
- FI_File_Icon * **next** ()
 Returns next file icon object.
- const char * **pattern** ()
 Returns the filename matching pattern for the icon.
- int **size** ()
 Returns the number of words of data used by the icon.
- int **type** ()
 Returns the filetype associated with the icon, which can be one of the following:
- short * **value** ()
 Returns the data array for the icon.
- ~FI_File_Icon ()
 The destructor destroys the icon and frees all memory that has been allocated for it.

Static Public Member Functions

- static FI_File_Icon * **find** (const char *filename, int filetype=ANY)
 Finds an icon that matches the given filename and file type.
- static FI_File_Icon * **first** ()
 Returns a pointer to the first icon in the list.
- static void **labeltype** (const FI_Label *o, int x, int y, int w, int h, FI_Align a)

Draw the icon label.

- static void [load_system_icons](#) (void)

Loads all system-defined icons.

11.37.1 Detailed Description

The [Fl_File_Icon](#) class manages icon images that can be used as labels in other widgets and as icons in the FileBrowser widget.

11.37.2 Constructor & Destructor Documentation

11.37.2.1 Fl_File_Icon()

```
Fl_File_Icon::Fl_File_Icon (
    const char * p,
    int t,
    int nd = 0,
    short * d = 0)
```

Creates a new [Fl_File_Icon](#) with the specified information.

Parameters

in	<i>p</i>	filename pattern
in	<i>t</i>	file type
in	<i>nd</i>	number of data values
in	<i>d</i>	data values

11.37.3 Member Function Documentation

11.37.3.1 add()

```
short * Fl_File_Icon::add (
    short d)
```

Adds a keyword value to the icon array, returning a pointer to it.

Parameters

in	<i>d</i>	data value

11.37.3.2 add_color()

```
short * Fl_File_Icon::add_color (
    Fl\_Color c) [inline]
```

Adds a color value to the icon array, returning a pointer to it.

Parameters

in	<i>c</i>	color value

11.37.3.3 add_vertex() [1/2]

```
short * Fl_File_Icon::add_vertex (
    float x,
    float y) [inline]
```

Adds a vertex value to the icon array, returning a pointer to it.

The floating point version goes from 0.0 to 1.0. The origin (0.0) is in the lower-lefthand corner of the icon.

Parameters

in	<i>x,y</i>	vertex coordinates

11.37.3.4 add_vertex() [2/2]

```
short * Fl_File_Icon::add_vertex (
    int x,
    int y) [inline]
```

Adds a vertex value to the icon array, returning a pointer to it.

The integer version accepts coordinates from 0 to 10000. The origin (0.0) is in the lower-lefthand corner of the icon.

Parameters

in	<i>x,y</i>	vertex coordinates

11.37.3.5 draw()

```
void Fl_File_Icon::draw (
    int x,
    int y,
    int w,
    int h,
    Fl_Color ic,
    int active = 1)
```

Draws an icon in the indicated area.

Parameters

in	<i>x,y,w,h</i>	position and size
in	<i>ic</i>	icon color
in	<i>active</i>	status, default is active [non-zero]

11.37.3.6 find()

```
Fl_File_Icon * Fl_File_Icon::find (
    const char * filename,
    int filetype = ANY) [static]
```

Finds an icon that matches the given filename and file type.

Parameters

in	<i>filename</i>	name of file
in	<i>filetype</i>	enumerated file type

Returns

matching file icon or NULL

11.37.3.7 label()

```
void Fl_File_Icon::label (
    Fl_Widget * w)
```

Applies the icon to the widget, registering the [Fl_File_Icon](#) label type as needed.

Parameters

in	<i>w</i>	widget for which this icon will become the label

11.37.3.8 labeltype()

```
void Fl_File_Icon::labeltype (
    const Fl_Label * o,
    int x,
    int y,
    int w,
    int h,
    Fl_Align a) [static]
```

Draw the icon label.

Parameters

in	<i>o</i>	label data
in	<i>x,y,w,h</i>	position and size of label
in	<i>a</i>	label alignment [not used]

11.37.3.9 load()

```
void Fl_File_Icon::load (
    const char * f)
```

Loads the specified icon image.

The format is deduced from the filename.

Parameters

in	<i>f</i>	filename

11.37.3.10 load_fti()

```
int Fl_File_Icon::load_fti (
    const char * fti)
```

Loads an SGI icon file.

Parameters

in	<i>fti</i>	icon filename

Returns

0 on success, non-zero on error

11.37.3.11 load_image()

```
int Fl_File_Icon::load_image (
    const char * ifile)
```

Load an image icon file from an image filename.

Parameters

in	<i>ifile</i>	image filename

Returns

0 on success, non-zero on error

11.37.3.12 load_system_icons()

```
void Fl_File_Icon::load_system_icons (
    void ) [static]
```

Loads all system-defined icons.

This call is useful when using the FileChooser widget and should be used when the application starts:

```
Fl_File_Icon::load_system_icons();
```

11.37.3.13 next()

```
Fl_File_Icon * Fl_File_Icon::next () [inline]
```

Returns next file icon object.

See [Fl_File_Icon::first\(\)](#)

11.37.3.14 type()

```
int Fl_File_Icon::type () [inline]
```

Returns the filetype associated with the icon, which can be one of the following:

- `Fl_File_Icon::ANY`, any kind of file.
- `Fl_File_Icon::PLAIN`, plain files.
- `Fl_File_Icon::FIFO`, named pipes.
- `Fl_File_Icon::DEVICE`, character and block devices.
- `Fl_File_Icon::LINK`, symbolic links.
- `Fl_File_Icon::DIRECTORY`, directories.

The documentation for this class was generated from the following files:

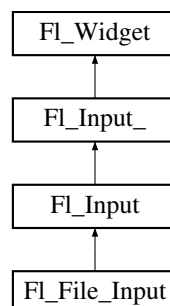
- `Fl_File_Icon.H`
- `Fl_File_Icon.cxx`
- `Fl_File_Icon2.cxx`

11.38 Fl_File_Input Class Reference

This widget displays a pathname in a text input field.

```
#include <Fl_File_Input.H>
```

Inheritance diagram for `Fl_File_Input`:



Public Member Functions

- [FI_Boxtype](#) **down_box** () const
Gets the box type used for the navigation bar.
- void **down_box** ([FI_Boxtype](#) b)
Sets the box type to use for the navigation bar.
- [FI_Color](#) **errorcolor** () const
Gets the current error color.
- void **errorcolor** ([FI_Color](#) c)
Sets the current error color to c.
- [FI_File_Input](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [FI_File_Input](#) widget using the given position, size, and label string.
- int **handle** (int event) [FL_OVERRIDE](#)
Handle events in the widget.
- const char * **value** ()
Returns the current value, which is a pointer to an internal buffer and is valid only until the next event is handled.
- int **value** (const char *str)
Sets the value of the widget given a new string value.
- int **value** (const char *str, int len)
Sets the value of the widget given a new string value and its length.

Public Member Functions inherited from [FI_Input](#)

- [FI_Input](#) (int, int, int, int, const char *L=0)
Creates a new [FI_Input](#) widget using the given position, size, and label string.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [FI_Input_](#)

- int **append** (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool **can_redo** () const
Check if there is a redo action available.
- bool **can_undo** () const
Check if the last operation can be undone.
- int **copy** (int clipboard)
Put the current selection into the clipboard.
- int **copy_cuts** ()
Copies the yank buffer to the clipboard.
- [FI_Color](#) **cursor_color** () const
Gets the color of the cursor.
- void **cursor_color** ([FI_Color](#) n)
Sets the color of the cursor.
- int **cut** ()
Deletes the current selection.
- int **cut** (int a, int b)
Deletes all characters between index a and b.
- int **cut** (int n)
Deletes the next n bytes rounded to characters before or after the cursor.
- double **dvalue** () const
Returns the widget text interpreted as a floating point number.

- `Fl_Input_` (int, int, int, int, const char * = 0)
Creates a new `Fl_Input_` widget.
- unsigned int `index` (int i) const
Returns the character at index `i`.
- int `input_type` () const
Gets the input field type.
- void `input_type` (int t)
Sets the input field type.
- int `insert` (const char *t, int l = 0)
Inserts text at the cursor position.
- int `insert_position` () const
Gets the position of the text cursor.
- int `insert_position` (int p)
Sets the cursor position and mark.
- int `insert_position` (int p, int m)
Sets the index for the cursor and mark.
- int `ivalue` () const
Returns the widget text interpreted as a signed integer.
- int `mark` () const
Gets the current selection mark.
- int `mark` (int m)
Sets the current selection mark.
- int `maximum_size` () const
Gets the maximum length of the input field in characters.
- void `maximum_size` (int m)
Sets the maximum length of the input field in characters.
- int `position` () const
- int `position` (int p)
- int `position` (int p, int m)
- int `readonly` () const
Gets the read-only state of the input field.
- void `readonly` (int b)
Sets the read-only state of the input field.
- int `redo` ()
Redo previous undo operation.
- int `replace` (int b, int e, const char *text, int len = 0)
Deletes text from `b` to `e` and inserts the new string `text`.
- void `resize` (int, int, int, int) `FL_OVERRIDE`
Changes the size of the widget.
- int `shortcut` () const
Return the shortcut key associated with this widget.
- void `shortcut` (int s)
Sets the shortcut key associated with this widget.
- int `size` () const
Returns the number of bytes in `value()`.
- void `size` (int W, int H)
Sets the width and height of this widget.
- int `static_value` (const char *)
Changes the widget text.
- int `static_value` (const char *, int)
Changes the widget text.

- int [tab_nav](#) () const
Gets whether the Tab key causes focus navigation in multiline input fields or not.
- void [tab_nav](#) (int val)
Sets whether the Tab key does focus navigation, or inserts tab characters into [FI_Multiline_Input](#).
- [FI_Color](#) [textcolor](#) () const
Gets the color of the text in the input field.
- void [textcolor](#) ([FI_Color](#) n)
Sets the color of the text in the input field.
- [FI_Font](#) [textfont](#) () const
Gets the font of the text in the input field.
- void [textfont](#) ([FI_Font](#) s)
Sets the font of the text in the input field.
- [FI_Fontsize](#) [textsize](#) () const
Gets the size of the text in the input field.
- void [textsize](#) ([FI_Fontsize](#) s)
Sets the size of the text in the input field.
- int [undo](#) ()
Undoes previous changes to the text buffer.
- const char * [value](#) () const
Returns the text displayed in the widget.
- int [value](#) (const char *)
Changes the widget text.
- int [value](#) (const char *, int)
Changes the widget text.
- int [value](#) (double value)
Changes the widget text to a floating point number ("%g").
- int [value](#) (int value)
Changes the widget text to a signed integer number.
- int [wrap](#) () const
Gets the word wrapping state of the input field.
- void [wrap](#) (int b)
Sets the word wrapping state of the input field.
- [~FI_Input](#) ()
Destroys the widget.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.

- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()
Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [Fl_Group](#) * [as_group](#) ()
Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).
- virtual [Fl_Group](#) const * [as_group](#) () const
- virtual [Fl_Window](#) * [as_window](#) ()
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- virtual [Fl_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([Fl_Boxtype](#) new_box)
Sets the box type for the widget.
- [Fl_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([Fl_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [Fl_Color](#) [color](#) () const
Gets the background color of the widget.

- void [color](#) ([FL_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FL_Color](#) bg, [FL_Color](#) sel)
Sets the background and selection color of the widget.
- [FL_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FL_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()

- Gets the image that is used as part of the widget label when in the active state.*

 - `const FI_Image * image () const`
- Gets the image that is used as part of the widget label when in the active state.*

 - `void image (FI_Image &img)`
- Sets the image to use as part of the widget label when in the active state.*

 - `void image (FI_Image *img)`
- Sets the image to use as part of the widget label when in the active state.*

 - `int image_bound () const`
- Returns whether the image is managed by the widget.*

 - `int inside (const FI_Widget *wgt) const`
- Checks if this widget is a child of wgt.*

 - `int is_label_copied () const`
- Returns whether the current label was assigned with copy_label().*

 - `const char * label () const`
- Gets the current label text.*

 - `void label (const char *text)`
- Sets the current label pointer.*

 - `void label (FI_Labeltype a, const char *b)`
- Shortcut to set the label text and type in one call.*

 - `int label_image_spacing ()`
- Return the gap size between the label and the image.*

 - `void label_image_spacing (int gap)`
- Set the gap between the label and the image in pixels.*

 - `FI_Color labelcolor () const`
- Gets the label color.*

 - `void labelcolor (FI_Color c)`
- Sets the label color.*

 - `FI_Font labelfont () const`
- Gets the font to use.*

 - `void labelfont (FI_Font f)`
- Sets the font to use.*

 - `FI_Fonsize labelsz () const`
- Gets the font size in pixels.*

 - `void labelsz (FI_Fonsize pix)`
- Sets the font size in pixels.*

 - `FI_Labeltype labeltype () const`
- Gets the label type.*

 - `void labeltype (FI_Labeltype a)`
- Sets the label type.*

 - `void measure_label (int &ww, int &hh) const`
- Sets width ww and height hh accordingly with the label size.*

 - `bool needs_keyboard () const`
- Returns whether this widget needs a keyboard.*

 - `void needs_keyboard (bool needs)`
- Sets whether this widget needs a keyboard.*

 - `unsigned int output () const`
- Returns if a widget is used for output only.*

 - `FI_Group * parent () const`
- Returns a pointer to the parent widget.*

 - `void parent (FI_Group *p)`
- Internal use only - "for hacks only".*

- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [Fl_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)

- Sets the user data for this widget.*
 - void **user_data** (void *v)
 - Sets the user data for this widget.*
 - int **vertical_label_margin** ()
 - Get the spacing between the label and the vertical edge of the widget.*
 - void **vertical_label_margin** (int px)
 - Set the spacing between the label and the vertical edge of the widget.*
 - unsigned int **visible** () const
 - Returns whether a widget is visible.*
 - unsigned int **visible_focus** () const
 - Checks whether this widget has a visible focus.*
 - void **visible_focus** (int v)
 - Modifies keyboard focus navigation.*
 - int **visible_r** () const
 - Returns whether a widget and all its parents are visible.*
 - int **w** () const
 - Gets the widget width.*
 - **FL_When when** () const
 - Returns the conditions under which the callback is called.*
 - void **when** (uchar i)
 - Sets the flags used to decide when a callback is called.*
 - **FL_Window * window** () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
 - int **x** () const
 - Gets the widget position in its window.*
 - int **y** () const
 - Gets the widget position in its window.*
 - virtual **~FL_Widget** ()
 - Destroys the widget.*

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
 - Draws the file input widget.*

Protected Member Functions inherited from **FL_Input**

- void **draw** () **FL_OVERRIDE**
 - Draws the widget.*
- int **handle_key** ()
 - Handles a keystroke.*
- int **handle_rmb** ()
 - Handle right mouse button down events.*

Protected Member Functions inherited from **FL_Input_**

- int **apply_undo** ()
 - Apply the current undo/redo operation.*
- void **drawtext** (int, int, int, int)
 - Draws the text in the passed bounding box.*
- void **drawtext** (int, int, int, int, bool draw_active)
 - Draws the text in the passed bounding box.*

- void **handle_mouse** (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int **handletext** (int e, int, int, int, int)
Handles all kinds of text field related events.
- int **line_end** (int i) const
Finds the end of a line.
- int **line_start** (int i) const
Finds the start of a line.
- int **linesPerPage** ()
- void **maybe_do_callback** (FI_Callback_Reason reason=FL_REASON_UNKNOWN)
- int **up_down_position** (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int **word_end** (int i) const
Finds the end of a word.
- int **word_start** (int i) const
Finds the start of a word.
- int **xscroll** () const
- int **yscroll** () const
- void **yscroll** (int yOffset)

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)

Internal use only.

- void **x** (int v)

Internal use only.

- void **y** (int v)

Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from [FL_Widget](#)

- static void [default_callback](#) ([FL_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Static Public Attributes inherited from [FL_Input](#)

- static const char * **copy_menu_text** = "Copy"
[this text may be customized at run-time]
- static const char * **cut_menu_text** = "Cut"
[this text may be customized at run-time]
- static const char * **paste_menu_text** = "Paste"
[this text may be customized at run-time]

Protected Types inherited from [FL_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.38.1 Detailed Description

This widget displays a pathname in a text input field.

A navigation bar located above the input field allows the user to navigate upward in the directory tree. You may want to handle `FL_WHEN_CHANGED` events for tracking text changes and also `FL_WHEN_RELEASE` for button release when changing to parent dir. `FL_WHEN_RELEASE` callback won't be called if the directory clicked is the same as the current one.

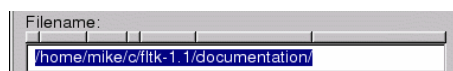


Figure 11.17 `FL_File_Input`

Note

As all [Fl_Input](#) derived objects, [Fl_File_Input](#) may call its callback when losing focus (see [FL_UNFOCUS](#)) to update its state like its cursor shape. One resulting side effect is that you should call [clear_changed\(\)](#) early in your callback to avoid reentrant calls if you plan to show another window or dialog box in the callback.

11.38.2 Constructor & Destructor Documentation**11.38.2.1 Fl_File_Input()**

```
Fl_File_Input::Fl_File_Input (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_File_Input](#) widget using the given position, size, and label string. The default boxtype is [FL_DOWN_BOX](#).

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

11.38.3 Member Function Documentation**11.38.3.1 down_box()**

```
void Fl_File_Input::down_box (
    Fl_Boxtype b) [inline]
```

Sets the box type to use for the navigation bar.

11.38.3.2 draw()

```
void Fl_File_Input::draw (
    void ) [protected], [virtual]
```

Draws the file input widget.

Implements [Fl_Widget](#).

11.38.3.3 errorcolor() [1/2]

```
Fl_Color Fl_File_Input::errorcolor () const [inline]
```

Gets the current error color.

Returns [FL_RED](#) since FLTK 1.4.0 (default in 1.3.x). Retained for backwards compatibility.

Deprecated Will be removed in FLTK 1.5.0 or higher.

Todo Remove [Fl_File_Input::errorcolor\(\)](#) in FLTK 1.5.0 or higher.

11.38.3.4 errorcolor() [2/2]

```
void Fl_File_Input::errorcolor (
    Fl_Color c) [inline]
```

Sets the current error color to *c*.

Does nothing since FLTK 1.4.0. Retained for backwards compatibility.

Deprecated Will be removed in FLTK 1.5.0 or higher.

Todo Remove [Fl_File_Input::errorcolor\(Fl_Color\)](#) in FLTK 1.5.0 or higher.

11.38.3.5 handle()

```
int Fl_File_Input::handle (
    int event) [virtual]
```

Handle events in the widget.

Return non zero if event is handled.

Parameters

in	<i>event</i>	

Reimplemented from [Fl_Widget](#).

11.38.3.6 value() [1/2]

```
int Fl_File_Input::value (
    const char * str)
```

Sets the value of the widget given a new string value.

Returns non 0 on success.

Parameters

in	<i>str</i>	new string value

11.38.3.7 value() [2/2]

```
int Fl_File_Input::value (
    const char * str,
    int len)
```

Sets the value of the widget given a new string value and its length.

Returns non 0 on success.

Parameters

in	<i>str</i>	new string value
in	<i>len</i>	length of value

The documentation for this class was generated from the following files:

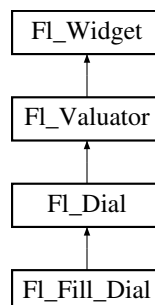
- [Fl_File_Input.H](#)
- [Fl_File_Input.cxx](#)

11.39 Fl_Fill_Dial Class Reference

Draws a dial with a filled arc.

```
#include <Fl_Fill_Dial.H>
```

Inheritance diagram for [Fl_Fill_Dial](#):



Public Member Functions

- **FI_Fill_Dial** (int X, int Y, int W, int H, const char *L=NULL)

Creates a filled dial, also setting its type to FL_FILL_DIAL.

Public Member Functions inherited from FI_Dial

- short **angle1** () const
Sets Or gets the angles used for the minimum and maximum values.
- void **angle1** (short a)
See short [angle1\(\)](#) const.
- short **angle2** () const
See short [angle1\(\)](#) const.
- void **angle2** (short a)
See short [angle1\(\)](#) const.
- void **angles** (short a, short b)
See short [angle1\(\)](#) const.
- **FI_Dial** (int x, int y, int w, int h, const char *l=0)
Creates a new [FI_Dial](#) widget using the given position, size, and label string.
- int **handle** (int) **FL_OVERRIDE**
Allow subclasses to handle event based on current position and size.

Public Member Functions inherited from FI_Valuator

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)
Sets the maximum value for the valuator.
- double **minimum** () const
Gets the minimum value for the valuator.
- void **minimum** (double a)
Sets the minimum value for the valuator.
- void **precision** (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void **range** (double a, double b)
Sets the minimum and maximum values for the valuator.
- double **round** (double)
Round the passed value to the nearest step increment.
- double **step** () const
Gets or sets the step value.
- void **step** (double a, int b)
See double [FI_Valuator::step\(\)](#) const
- void **step** (double s)

- See double [FI_Valuator::step\(\) const](#).
- void **step** (int a)
 - See double [FI_Valuator::step\(\) const](#)
- double **value** () const
 - Gets the floating point(double) value.
- int **value** (double)
 - Sets the current value.
- ~**FI_Valuator** () **FL_OVERRIDE**
 - Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
 - Activates the widget.
- unsigned int **active** () const
 - Returns whether the widget is active.
- int **active_r** () const
 - Returns whether the widget and all of its parents are active.
- [FI_Align](#) **align** () const
 - Gets the label alignment.
- void **align** ([FI_Align](#) alignment)
 - Sets the label alignment.
- long **argument** () const
 - Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
 - Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * **as_gl_window** ()
 - Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * **as_gl_window** () const
- virtual [FI_Group](#) * **as_group** ()
 - Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * **as_group** () const
- virtual [FI_Window](#) * **as_window** ()
 - Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * **as_window** () const
- void **bind_deimage** ([FI_Image](#) *img)
 - Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
 - Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** ([FI_Image](#) *img)
 - Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
 - Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) **box** () const
 - Gets the box type of the widget.
- void **box** ([FI_Boxtype](#) new_box)
 - Sets the box type for the widget.
- [FI_Callback_p](#) **callback** () const
 - Gets the current callback function for the widget.

- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) ([uchar](#) c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [FI_Image](#) * [deimage](#) ()

- Gets the image that is used as part of the widget label when in the inactive state.*

 - `const FL\_Image * deimage () const`
- Gets the image that is used as part of the widget label when in the inactive state.*

 - `void deimage (FL\_Image &img)`
- Sets the image to use as part of the widget label when in the inactive state.*

 - `void deimage (FL\_Image *img)`
- Sets the image to use as part of the widget label when in the inactive state.*

 - `int deimage\_bound () const`
- Returns whether the inactive image is managed by the widget.*

 - `void do\_callback (FL\_Callback\_Reason reason=FL\_REASON\_UNKNOWN)`
- Calls the widget callback function with default arguments.*

 - `void do\_callback (FL\_Widget *widget, long arg, FL\_Callback\_Reason reason=FL\_REASON\_UNKNOWN)`
- Calls the widget callback function with arbitrary arguments.*

 - `void do\_callback (FL\_Widget *widget, void *arg=0, FL\_Callback\_Reason reason=FL\_REASON\_UNKNOWN)`
- Calls the widget callback function with arbitrary arguments.*

 - `void draw\_label (int, int, int, int, FL\_Align) const`
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*

 - `int h () const`
- Gets the widget height.*

 - `virtual void hide ()`
- Makes a widget invisible.*

 - `int horizontal\_label\_margin ()`
- Get the spacing between the label and the horizontal edge of the widget.*

 - `void horizontal\_label\_margin (int px)`
- Set the spacing between the label and the horizontal edge of the widget.*

 - `FL\_Image * image ()`
- Gets the image that is used as part of the widget label when in the active state.*

 - `const FL\_Image * image () const`
- Gets the image that is used as part of the widget label when in the active state.*

 - `void image (FL\_Image &img)`
- Sets the image to use as part of the widget label when in the active state.*

 - `void image (FL\_Image *img)`
- Sets the image to use as part of the widget label when in the active state.*

 - `int image\_bound () const`
- Returns whether the image is managed by the widget.*

 - `int inside (const FL\_Widget *wgt) const`
- Checks if this widget is a child of wgt.*

 - `int is\_label\_copied () const`
- Returns whether the current label was assigned with [copy_label\(\)](#).*

 - `const char * label () const`
- Gets the current label text.*

 - `void label (const char *text)`
- Sets the current label pointer.*

 - `void label (FL\_Labeltype a, const char *b)`
- Shortcut to set the label text and type in one call.*

 - `int label\_image\_spacing ()`
- Return the gap size between the label and the image.*

 - `void label\_image\_spacing (int gap)`
- Set the gap between the label and the image in pixels.*

 - `FL\_Color labelcolor () const`
- Gets the label color.*

- void [labelcolor](#) ([FI_Color](#) c)
Sets the label color.
- [FI_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FI_Font](#) f)
Sets the font to use.
- [FI_Fontsize](#) [labelsizesize](#) () const
Gets the font size in pixels.
- void [labelsizesize](#) ([FI_Fontsize](#) pix)
Sets the font size in pixels.
- [FI_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FI_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FI_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FI_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)
Changes the size or position of the widget.
- [FI_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - virtual void **show** ()

Makes a widget visible.
- void **size** (int W, int H)

Changes the size of the widget.
- int **take_focus** ()

Gives the widget the keyboard focus.
- unsigned int **takeevents** () const

Returns if the widget is able to take events.
- int **test_shortcut** ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * **tooltip** () const

Gets the current tooltip text.
- void **tooltip** (const char *text)

Sets the current tooltip text.
- **Fl_Window** * **top_window** () const

Returns a pointer to the top-level window for the widget.
- **Fl_Window** * **top_window_offset** (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- **uchar** **type** () const

Gets the widget type.
- void **type** (**uchar** t)

Sets the widget type.
- int **use_accents_menu** ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const

Gets the user data for this widget.
- void **user_data** (**Fl_Callback_User_Data** *v, bool auto_free)

Sets the user data for this widget.
- void **user_data** (void *v)

Sets the user data for this widget.
- int **vertical_label_margin** ()

Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const

Returns whether a widget is visible.
- unsigned int **visible_focus** () const

Checks whether this widget has a visible focus.
- void **visible_focus** (int v)

Modifies keyboard focus navigation.
- int **visible_r** () const

Returns whether a widget and all its parents are visible.
- int **w** () const

Gets the widget width.
- **Fl_When** **when** () const

Returns the conditions under which the callback is called.
- void **when** (**uchar** i)

Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const

Returns a pointer to the nearest parent window up the widget hierarchy.

- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~FI_Widget** ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from **FI_Widget**

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from **FI_Widget**

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from **FI_Dial**

- void **draw** () **FL_OVERRIDE**
Draws dial at current position and size.
- void **draw** (int X, int Y, int W, int H)
Draws dial at given position and size.
- int **handle** (int event, int X, int Y, int W, int H)
Allows subclasses to handle event based on given position and size.

Protected Member Functions inherited from **FI_Valuator**

- **FI_Valuator** (int X, int Y, int W, int H, const char *L)
Creates a new FI_Valuator widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.

- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void **value_damage** ()
Asks for partial redraw.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.39.1 Detailed Description

Draws a dial with a filled arc.

The documentation for this class was generated from the following files:

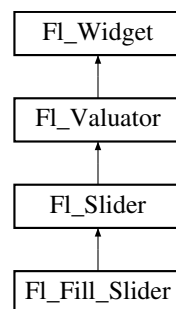
- Fl_Fill_Dial.H
- Fl_Dial.cxx

11.40 Fl_Fill_Slider Class Reference

Widget that draws a filled horizontal slider, useful as a progress or value meter.

```
#include <Fl_Fill_Slider.H>
```

Inheritance diagram for Fl_Fill_Slider:



Public Member Functions

- **Fl_Fill_Slider** (int X, int Y, int W, int H, const char *L=0)
Creates the slider from its position, size and optional title.

Public Member Functions inherited from [Fl_Slider](#)

- void [bounds](#) (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- [Fl_Slider](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Slider](#) widget using the given position, size, and label string.
- **Fl_Slider** ([uchar](#) t, int X, int Y, int W, int H, const char *L)
Creates a new [Fl_Slider](#) widget using the given type, position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- int [scrollvalue](#) (int pos, int [size](#), int first, int total)
Sets the size and position of the sliding knob in the box.
- [Fl_Boxtype](#) **slider** () const
Gets the slider box type.
- void **slider** ([Fl_Boxtype](#) c)
Sets the slider box type.
- float **slider_size** () const
Get the dimensions of the moving piece of slider.
- void [slider_size](#) (double v)
Set the dimensions of the moving piece of slider.

Public Member Functions inherited from [FI_Valuator](#)

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)
Sets the maximum value for the valuator.
- double **minimum** () const
Gets the minimum value for the valuator.
- void **minimum** (double a)
Sets the minimum value for the valuator.
- void **precision** (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void **range** (double a, double b)
Sets the minimum and maximum values for the valuator.
- double **round** (double)
Round the passed value to the nearest step increment.
- double **step** () const
Gets or sets the step value.
- void **step** (double a, int b)
See double [FI_Valuator::step\(\)](#) const
- void **step** (double s)
See double [FI_Valuator::step\(\)](#) const.
- void **step** (int a)
See double [FI_Valuator::step\(\)](#) const
- double **value** () const
Gets the floating point(double) value.
- int **value** (double)
Sets the current value.
- **~FI_Valuator** () **FL_OVERRIDE**
Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const

- Gets the label alignment.*
- void [align](#) ([FI_Align](#) alignment)
- Sets the label alignment.*
- long [argument](#) () const
- Gets the current user data (long) argument that is passed to the callback function.*
- void [argument](#) (long v)
- Sets the current user data (long) argument that is passed to the callback function.*
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
- Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).*
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
- Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).*
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void [bind_deimage](#) (int f)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- void [bind_image](#) ([FI_Image](#) *img)
- Sets the image to use as part of the widget label when in the active state.*
- void [bind_image](#) (int f)
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- [FI_Boxtype](#) [box](#) () const
- Gets the box type of the widget.*
- void [box](#) ([FI_Boxtype](#) new_box)
- Sets the box type for the widget.*
- [FI_Callback_p](#) [callback](#) () const
- Gets the current callback function for the widget.*
- void [callback](#) ([FI_Callback](#) *cb)
- Sets the current callback function for the widget.*
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*
- void [callback](#) ([FI_Callback](#) *cb, void *p)
- Sets the current callback function and data for the widget.*
- void [callback](#) ([FI_Callback0](#) *cb)
- Sets the current callback function for the widget.*
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
- Sets the current callback function for the widget.*
- unsigned int [changed](#) () const
- Checks if the widget value changed since the last callback.*
- void [clear_active](#) ()
- Marks the widget as inactive without sending events or changing focus.*
- void [clear_changed](#) ()
- Marks the value of the widget as unchanged.*
- void [clear_damage](#) (uchar c=0)
- Clears or sets the damage flags.*
- void [clear_output](#) ()
- Sets a widget to accept input.*
- void [clear_visible](#) ()

- Hides the widget.*
- void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*
- `FL_Color` `color` () const
- Gets the background color of the widget.*
- void `color` (`FL_Color` bg)
- Sets the background color of the widget.*
- void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*
- `FL_Color` `color2` () const
- For back compatibility only.*
- void `color2` (unsigned a)
- For back compatibility only.*
- int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*
- void `copy_label` (const char *new_label)
- Sets the current label.*
- void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*
- `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*
- void `damage` (`uchar` c)
- Sets the damage bits for the widget.*
- void `damage` (`uchar` c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int `damage_resize` (int, int, int, int)
- Internal use only.*
- void `deactivate` ()
- Deactivates the widget.*
- `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*
- const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with default arguments.*
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*
- void `draw_label` (int, int, int, int, `FL_Align`) const
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int `h` () const
- Gets the widget height.*
- virtual void `hide` ()
- Makes a widget invisible.*

- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FI_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FI_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FI_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FI_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FI_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FI_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FI_Color](#) c)
Sets the label color.
- [FI_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FI_Font](#) f)
Sets the font to use.
- [FI_Fontsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FI_Fontsize](#) pix)
Sets the font size in pixels.
- [FI_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FI_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

- Returns if a widget is used for output only.*

 - `Fl_Group * parent () const`

Returns a pointer to the parent widget.
- `void parent (Fl_Group *p)`

Internal use only - "for hacks only".
- `void position (int X, int Y)`

Repositions the window or widget.
- `void redraw ()`

Schedules the drawing of the widget.
- `void redraw_label ()`

Schedules the drawing of the label.
- `virtual void resize (int x, int y, int w, int h)`

Changes the size or position of the widget.
- `Fl_Color selection_color () const`

Gets the selection color.
- `void selection_color (Fl_Color a)`

Sets the selection color.
- `void set_active ()`

Marks the widget as active without sending events or changing focus.
- `void set_changed ()`

Marks the value of the widget as changed.
- `void set_output ()`

Sets a widget to output only.
- `void set_visible ()`

Makes the widget visible.
- `void set_visible_focus ()`

Enables keyboard focus navigation with this widget.
- `int shortcut_label () const`

Returns whether the widget's label uses '&' to indicate shortcuts.
- `void shortcut_label (int value)`

Sets whether the widget's label uses '&' to indicate shortcuts.
- `virtual void show ()`

Makes a widget visible.
- `void size (int W, int H)`

Changes the size of the widget.
- `int take_focus ()`

Gives the widget the keyboard focus.
- `unsigned int takeevents () const`

Returns if the widget is able to take events.
- `int test_shortcut ()`

Returns true if the widget's label contains the entered '&x' shortcut.
- `const char * tooltip () const`

Gets the current tooltip text.
- `void tooltip (const char *text)`

Sets the current tooltip text.
- `Fl_Window * top_window () const`

Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`

Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`

Gets the widget type.

- void **type** (uchar t)
Sets the widget type.
- int **use_accents_menu** ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const
Gets the user data for this widget.
- void **user_data** (FI_Callback_User_Data *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- FI_When **when** () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- FI_Window * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual ~FI_Widget ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [FL_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from [FL_Slider](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void **draw** (int, int, int, int)
- int **handle** (int, int, int, int, int)

Protected Member Functions inherited from [FL_Valuator](#)

- [FL_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FL_Valuator](#) widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an [FL_WHEN_CHANGED](#) event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an [FL_WHEN_RELEASE](#) event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an [FL_HORIZONTAL](#) one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void [value_damage](#) ()
Asks for partial redraw.

Protected Member Functions inherited from [FL_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If [FL_ALIGN_IMAGE_BACKDROP](#) is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.

- void **draw_box** (Fl_Boxtype t, Fl_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **Fl_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.40.1 Detailed Description

Widget that draws a filled horizontal slider, useful as a progress or value meter.
The documentation for this class was generated from the following files:

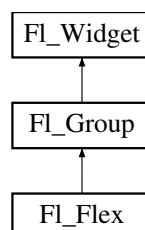
- Fl_Fill_Slider.H
- Fl_Slider.cxx

11.41 Fl_Flex Class Reference

Fl_Flex is a container (layout) widget for one row or one column of widgets.

```
#include <Fl_Flex.H>
```

Inheritance diagram for Fl_Flex:



Public Types

- enum { `VERTICAL` = 0 , `HORIZONTAL` = 1 , `COLUMN` = 0 , `ROW` = 1 }

Public Member Functions

- virtual void `end` ()
Ends automatic child addition and resizes all children.
- void `fixed` (`Fl_Widget` &`w`, int `size`)
Set the horizontal or vertical size of a child widget.
- int `fixed` (`Fl_Widget` *`w`) const
Return whether the given widget has a fixed size or resizes dynamically.
- void `fixed` (`Fl_Widget` *`w`, int `size`)
Set the horizontal or vertical size of a child widget.
- `Fl_Flex` (int `direction`)
Construct a new `Fl_Flex` widget specifying its layout.
- `Fl_Flex` (int `w`, int `h`, int `direction`)
Construct a new `Fl_Flex` widget specifying its layout and size.
- `Fl_Flex` (int `X`, int `Y`, int `W`, int `H`, const char *`L`=0)
Construct a new `Fl_Flex` widget with the given position, size, and label.
- `Fl_Flex` (int `x`, int `y`, int `w`, int `h`, int `direction`)
Construct a new `Fl_Flex` widget specifying its layout, position, and size.
- int `gap` () const
Return the gap size of the widget.
- void `gap` (int `g`)
Set the gap size of the widget.
- int `horizontal` () const
Returns non-zero (true) if `Fl_Flex` alignment is horizontal (row mode).
- void `layout` ()
Calculates the layout of the widget and redraws it.
- int `margin` () const
Returns the left margin size of the widget.
- int `margin` (int *`left`, int *`top`, int *`right`, int *`bottom`) const
Returns all (four) margin sizes of the widget.
- void `margin` (int `left`, int `top`, int `right`, int `bottom`)
Set the margin sizes at all four edges of the `Fl_Flex` widget.
- void `margin` (int `m`, int `g`=-1)
Set the margin and optionally the gap size of the widget.
- bool `need_layout` () const
Returns whether layout calculation is required.
- void `need_layout` (int `set`)
Set or reset the request to calculate the layout of children.
- void `resize` (int `x`, int `y`, int `w`, int `h`) `FL_OVERRIDE`
Resize the container and calculate all child positions and sizes.
- int `spacing` () const
Gets the number of extra pixels of blank space that are added between the children.
- void `spacing` (int `i`)
Sets the number of extra pixels of blank space that are added between the children.

Public Member Functions inherited from FL_Group

- [FL_Widget](#) *[_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void **add** ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void **add_resizable** ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * **array** () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * **as_group** () const [FL_OVERRIDE](#)
- [FL_Group](#) * **as_group** () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
*Removes the widget at *index* from the group and deletes it.*
- void **end** ()
*Exactly the same as *current(this->parent())*.*
- int **find** (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int **find** (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char * = 0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void **focus** ([FL_Widget](#) *W)
- void **forms_end** ()
This is for forms compatibility only.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** ([FL_Widget](#) &o, [FL_Widget](#) *before)
*This does *insert(w, find(before))*.*
- void **remove** ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void **remove** ([FL_Widget](#) *o)

- Removes the widget `o` from the group.*
 - void `remove` (int index)
- Removes the widget at `index` from the group but does not delete it.*
 - `FL_Widget * resizable` () const
- Returns the group's resizable widget.*
 - void `resizable` (`FL_Widget &o`)
- Sets the group's resizable widget.*
 - void `resizable` (`FL_Widget *o`)
- The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
 - void `resize` (int, int, int, int) `FL_OVERRIDE`
- Resizes the `FL_Group` widget and all of its children.*
 - virtual `~FL_Group` ()
- The destructor also deletes all the children.*

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
- Activates the widget.*
 - unsigned int `active` () const
- Returns whether the widget is active.*
 - int `active_r` () const
- Returns whether the widget and all of its parents are active.*
 - `FL_Align align` () const
- Gets the label alignment.*
 - void `align` (`FL_Align alignment`)
- Sets the label alignment.*
 - long `argument` () const
- Gets the current user data (long) argument that is passed to the callback function.*
 - void `argument` (long v)
- Sets the current user data (long) argument that is passed to the callback function.*
 - virtual class `FL_Gl_Window * as_gl_window` ()
- Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.*
 - virtual class `FL_Gl_Window` const * `as_gl_window` () const
- virtual `FL_Window * as_window` ()
- Returns an `FL_Window` pointer if this widget is an `FL_Window`.*
 - virtual `FL_Window` const * `as_window` () const
- void `bind_deimage` (`FL_Image *img`)
- Sets the image to use as part of the widget label when in the inactive state.*
 - void `bind_deimage` (int f)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
 - void `bind_image` (`FL_Image *img`)
- Sets the image to use as part of the widget label when in the active state.*
 - void `bind_image` (int f)
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
 - `FL_Boxtype box` () const
- Gets the box type of the widget.*
 - void `box` (`FL_Boxtype new_box`)
- Sets the box type for the widget.*
 - `FL_Callback_p callback` () const

- Gets the current callback function for the widget.*

 - void `callback` (`FL_Callback` *cb)
- Sets the current callback function for the widget.*

 - void `callback` (`FL_Callback` *cb, `FL_Callback_User_Data` *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*

 - void `callback` (`FL_Callback` *cb, void *p)
- Sets the current callback function and data for the widget.*

 - void `callback` (`FL_Callback0` *cb)
- Sets the current callback function for the widget.*

 - void `callback` (`FL_Callback1` *cb, long p=0)
- Sets the current callback function for the widget.*

 - unsigned int `changed` () const
- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()
- Marks the widget as inactive without sending events or changing focus.*

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*

 - void `clear_damage` (`uchar` c=0)
- Clears or sets the damage flags.*

 - void `clear_output` ()
- Sets a widget to accept input.*

 - void `clear_visible` ()
- Hides the widget.*

 - void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*

 - `FL_Color` `color` () const
- Gets the background color of the widget.*

 - void `color` (`FL_Color` bg)
- Sets the background color of the widget.*

 - void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*

 - `FL_Color` `color2` () const
- For back compatibility only.*

 - void `color2` (unsigned a)
- For back compatibility only.*

 - int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*

 - void `copy_label` (const char *new_label)
- Sets the current label.*

 - void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*

 - `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*

 - void `damage` (`uchar` c)
- Sets the damage bits for the widget.*

 - void `damage` (`uchar` c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int `damage_resize` (int, int, int, int)
- Internal use only.*

 - void `deactivate` ()
- Deactivates the widget.*

- `FI_Image * deimage ()`
Gets the image that is used as part of the widget label when in the inactive state.
- `const FI_Image * deimage () const`
Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FI_Image &img)`
Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FI_Image *img)`
Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`
Returns whether the inactive image is managed by the widget.
- `void do_callback (FI_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with default arguments.
- `void do_callback (FI_Widget *widget, long arg, FI_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void do_callback (FI_Widget *widget, void *arg=0, FI_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FI_Align) const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`
Gets the widget height.
- `virtual void hide ()`
Makes a widget invisible.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FI_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FI_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FI_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FI_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FI_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FI_Color labelcolor () const`

- Gets the label color.*

 - void [labelcolor](#) ([Fl_Color](#) c)
- Sets the label color.*

 - [Fl_Font](#) [labelfont](#) () const
- Gets the font to use.*

 - void [labelfont](#) ([Fl_Font](#) f)
- Sets the font to use.*

 - [Fl_Fontsize](#) [labelsize](#) () const
- Gets the font size in pixels.*

 - void [labelsize](#) ([Fl_Fontsize](#) pix)
- Sets the font size in pixels.*

 - [Fl_Labeltype](#) [labeltype](#) () const
- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const
- Sets width ww and height hh accordingly with the label size.*

 - bool [needs_keyboard](#) () const
- Returns whether this widget needs a keyboard.*

 - void [needs_keyboard](#) (bool needs)
- Sets whether this widget needs a keyboard.*

 - unsigned int [output](#) () const
- Returns if a widget is used for output only.*

 - [Fl_Group](#) * [parent](#) () const
- Returns a pointer to the parent widget.*

 - void [parent](#) ([Fl_Group](#) *p)
- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
- Repositions the window or widget.*

 - void [redraw](#) ()
- Schedules the drawing of the widget.*

 - void [redraw_label](#) ()
- Schedules the drawing of the label.*

 - [Fl_Color](#) [selection_color](#) () const
- Gets the selection color.*

 - void [selection_color](#) ([Fl_Color](#) a)
- Sets the selection color.*

 - void [set_active](#) ()
- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

 - void [set_output](#) ()
- Sets a widget to output only.*

 - void [set_visible](#) ()
- Makes the widget visible.*

 - void [set_visible_focus](#) ()
- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void [shortcut_label](#) (int value)
- Sets whether the widget's label uses '&' to indicate shortcuts.*

- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const

- *Gets the widget position in its window.*
- `int y () const`
Gets the widget position in its window.
- `virtual ~FL_Widget ()`
Destroys the widget.

Protected Member Functions

- `virtual int alloc_size (int size) const`
Return new size to be allocated for array of fixed size widgets.
- `void draw () FL_OVERRIDE`
Draw the widget.
- `void init (int t=VERTICAL)`
- `void on_remove (int) FL_OVERRIDE`
Allow derived groups to act when a child widget is removed from the group.

Protected Member Functions inherited from FL_Group

- `FL_Rect * bounds ()`
Returns the internal array of widget sizes and positions.
- `void draw () FL_OVERRIDE`
Draws the widget.
- `void draw_child (FL_Widget &widget) const`
Forces a child to redraw.
- `void draw_children ()`
Draws all children of the group.
- `void draw_outside_label (const FL_Widget &widget) const`
Parents normally call this to draw outside labels of child widgets.
- `virtual int on_insert (FL_Widget *, int)`
Allow derived groups to act when a widget is added as a child.
- `virtual int on_move (int, int)`
Allow derived groups to act when a widget is moved within the group.
- `int * sizes ()`
Returns the internal array of widget sizes and positions.
- `void update_child (FL_Widget &widget) const`
Draws a child only if it needs it.

Protected Member Functions inherited from FL_Widget

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`

- Draws a focus rectangle around the widget.*
- void `draw_focus` (`FI_Boxtype` t, int x, int y, int w, int h, `FI_Color` bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `FI_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from `FI_Group`

- static `FI_Group` * `current` ()
Returns the currently active group.
- static void `current` (`FI_Group` *g)
Sets the current group.

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }

flags possible values enumeration.

11.41.1 Detailed Description

[Fl_Flex](#) is a container (layout) widget for one row or one column of widgets.

It provides flexible positioning of its children either in one row or in one column.

[Fl_Flex](#) is designed to be as simple as possible. You can set individual widget sizes or let [Fl_Flex](#) position and size the widgets to fit in the container. All "flexible" (i.e. non-fixed size) widgets are assigned the same width or height, respectively. For details see below.

You can set the margins **around** all children at the inner side of the box frame (if any). [Fl_Flex](#) supports setting different margin sizes on top, bottom, left, and right sides. The default margin size is 0 on all edges of the container. You can set the gap size **between** all children. The gap size is always the same between all of its children. This is similar to the 'spacing' of [Fl_Pack](#). The default gap size is 0.

[Fl_Flex](#) can either consist of a single row, i.e. `type(Fl_Flex::HORIZONTAL)` or a single column, i.e. `type(Fl_Flex::VERTICAL)`. The default value is [Fl_Flex::VERTICAL](#) for consistency with [Fl_Pack](#) but you can use `type()` to assign a row ([Fl_Flex::HORIZONTAL](#)) layout.

If `type() == Fl_Flex::HORIZONTAL` widgets are resized horizontally to fit in the container and their height is the full [Fl_Flex](#) height minus border size and margins. You can set a fixed widget width by using `fixed()`.

If `type() == Fl_Flex::VERTICAL` widgets are resized vertically to fit in the container and their width is the full [Fl_Flex](#) width minus border size and margins. You can set a fixed widget height by using `fixed()`.

To create arbitrary spacing you can use invisible boxes of flexible or fixed sizes (see example below).

Alternate constructors let you specify the layout as [Fl_Flex::HORIZONTAL](#) or [Fl_Flex::VERTICAL](#) directly.

[Fl_Flex::ROW](#) is an alias of [Fl_Flex::HORIZONTAL](#) and [Fl_Flex::COLUMN](#) is an alias of [Fl_Flex::VERTICAL](#).

The default box type is `FL_NO_BOX` as inherited from [Fl_Group](#). You **may** need to set a box type with a solid background depending on your layout.

Important: You should always make sure that the [Fl_Flex](#) container cannot be resized smaller than its designed minimal size. This can usually be done by setting a `size_range()` on the window as shown in the example below. [Fl_Flex](#) does not take care of sensible sizes. If it is resized too small the behavior is undefined, i.e. widgets may overlap and/or shrink to zero size.

Hint: In many cases [Fl_Flex](#) can be used as a drop-in replacement for [Fl_Pack](#). This is the recommended single row/column container since FLTK 1.4.0. Its resizing behavior is much more predictable than that of [Fl_Pack](#) which *"resizes itself to shrink-wrap itself around all of the children"*.

[Fl_Flex](#) containers can be nested so you can create flexible layouts with multiple columns and rows. However, if your UI design is more complex you may want to use [Fl_Grid](#) instead.

Example:

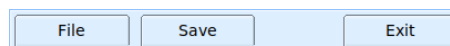


Figure 11.18 [Fl_Flex](#)

Example code:

```
#include <FL/Fl.H>
#include <FL/Fl_Double_Window.H>
#include <FL/Fl_Flex.H>
#include <FL/Fl_Box.H>
#include <FL/Fl_Button.H>

int main(int argc, char **argv) {
    Fl_Double_Window window(410, 40, "Simple Fl_Flex Demo");
    Fl_Flex flex(5, 5, 400, 30, Fl_Flex::HORIZONTAL);
    Fl_Button b1(0, 0, 0, 0, "File");
    Fl_Button b2(0, 0, 0, 0, "Save");
    Fl_Box bx(0, 0, 0, 0);
    Fl_Button b3(0, 0, 0, 0, "Exit");
    flex.fixed(bx, 60); // set fix width of invisible box
    flex.gap(10);
    flex.end();
    window.resizable(flex);
    window.end();
    window.size_range(300, 30);
    window.show(argc, argv);
    return Fl::run();
}
```

Since

1.4.0

11.41.2 Member Enumeration Documentation

11.41.2.1 anonymous enum

anonymous enum

Enumerator

VERTICAL	vertical layout (one column)
HORIZONTAL	horizontal layout (one row)
COLUMN	alias for VERTICAL
ROW	alias for HORIZONTAL

11.41.3 Constructor & Destructor Documentation

11.41.3.1 Fl_Flex() [1/4]

```
Fl_Flex::Fl_Flex (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Construct a new [Fl_Flex](#) widget with the given position, size, and label.

You can set `type(Fl_Flex::HORIZONTAL)` or `type(Fl_Flex::VERTICAL)`. The default is `type(Fl_Flex::VERTICAL)`.

Alternate constructors let you specify the layout as [Fl_Flex::HORIZONTAL](#) or [Fl_Flex::VERTICAL](#) directly. [Fl_Flex::ROW](#) is an alias of [Fl_Flex::HORIZONTAL](#) and [Fl_Flex::COLUMN](#) is an alias of [Fl_Flex::VERTICAL](#).

Parameters

in	<i>X,Y</i>	position
in	<i>W,H</i>	size (width and height)
in	<i>L</i>	label (optional)

See also

[Fl_Flex::Fl_Flex\(int direction\)](#)

[Fl_Flex::Fl_Flex\(int w, int h, int direction\)](#)

[Fl_Flex::Fl_Flex\(int x, int y, int w, int h, int direction\)](#)

[Fl_Flex::Fl_Flex\(int x, int y, int w, int h, const char *L\)](#)

11.41.3.2 Fl_Flex() [2/4]

```
Fl_Flex::Fl_Flex (
    int direction)
```

Construct a new [Fl_Flex](#) widget specifying its layout.

Use [Fl_Flex::HORIZONTAL](#) (aka [Fl_Flex::ROW](#)) or [Fl_Flex::VERTICAL](#) (aka [Fl_Flex::COLUMN](#)) as the `direction` argument.

This constructor sets the position and size to (0, 0, 0, 0) which is suitable for nested [Fl_Flex](#) widgets. Use one of the other constructors to set the desired position and size as well.

Parameters

in	<i>direction</i>	horizontal (row) or vertical (column) layout

See also

[FI_Flex::FI_Flex\(int w, int h, int direction\)](#)
[FI_Flex::FI_Flex\(int x, int y, int w, int h, int direction\)](#)
[FI_Flex::FI_Flex\(int x, int y, int w, int h, const char *L\)](#)

11.41.3.3 FI_Flex() [3/4]

```
FI_Flex::FI_Flex (
    int w,
    int h,
    int direction)
```

Construct a new [FI_Flex](#) widget specifying its layout and size.

Use [FI_Flex::HORIZONTAL](#) (aka [FI_Flex::ROW](#)) or [FI_Flex::VERTICAL](#) (aka [FI_Flex::COLUMN](#)) as the *direction* argument.

This constructor sets the position to (x = 0, y = 0) which is suitable for nested [FI_Flex](#) widgets. Use one of the other constructors to set the desired position as well.

Parameters

in	<i>w,h</i>	widget size
in	<i>direction</i>	horizontal (row) or vertical (column) layout

See also

[FI_Flex::FI_Flex\(int direction\)](#)
[FI_Flex::FI_Flex\(int x, int y, int w, int h, int direction\)](#)
[FI_Flex::FI_Flex\(int x, int y, int w, int h, const char *L\)](#)

11.41.3.4 FI_Flex() [4/4]

```
FI_Flex::FI_Flex (
    int x,
    int y,
    int w,
    int h,
    int direction)
```

Construct a new [FI_Flex](#) widget specifying its layout, position, and size.

Use [FI_Flex::HORIZONTAL](#) (aka [FI_Flex::ROW](#)) or [FI_Flex::VERTICAL](#) (aka [FI_Flex::COLUMN](#)) as the *direction* argument.

This constructor sets the position and size of the widget which is suitable for top level [FI_Flex](#) widgets but does not set a widget label. Use [FI_Widget::label\(\)](#) to set one if desired.

Parameters

in	<i>x,y</i>	widget position
in	<i>w,h</i>	widget size
in	<i>direction</i>	horizontal (row) or vertical (column) layout

See also

[FI_Flex::FI_Flex\(int direction\)](#)
[FI_Flex::FI_Flex\(int w, int h, int direction\)](#)
[FI_Flex::FI_Flex\(int x, int y, int w, int h, const char *L\)](#)

11.41.4 Member Function Documentation

11.41.4.1 `alloc_size()`

```
int Fl_Flex::alloc_size (
    int size) const [protected], [virtual]
```

Return new size to be allocated for array of fixed size widgets.

This method is called when the array of fixed size widgets needs to be expanded. The current `size` is provided (size can be 0). The default method adds 8 to the current size.

This can be used in derived classes to change the allocation strategy. Note that this method only `queries` the new size which shall be allocated but does not allocate the memory.

Parameters

in	size	current size

Returns

int new size (to be allocated)

11.41.4.2 `draw()`

```
void Fl_Flex::draw (
    void ) [protected], [virtual]
```

Draw the widget.

This will finally calculate the layout of the widget and of all its children if necessary and draw the widget.

Some changes of included children may require a new layout to be calculated. If this is the case the user may need to call [layout\(\)](#) to make sure everything is calculated properly.

See also

[layout\(\)](#)

Implements [Fl_Widget](#).

11.41.4.3 `end()`

```
void Fl_Flex::end () [virtual]
```

Ends automatic child addition and resizes all children.

This marks the [Fl_Flex](#) widget as changed (`need_layout(1)`) which forces the widget to calculate its layout depending on all children and whether they have been assigned fix sizes or not right before it is drawn.

See also

[need_layout\(int\)](#)

[draw\(\)](#)

11.41.4.4 `fixed()` [1/3]

```
void Fl_Flex::fixed (
    Fl_Widget & w,
    int size) [inline]
```

Set the horizontal or vertical size of a child widget.

Parameters

in	w	widget to be affected
in	size	width (Fl_Flex::HORIZONTAL) or height (Fl_Flex::VERTICAL)

See also

[fixed\(Fl_Widget *w, int size\)](#)

11.41.4.5 fixed() [2/3]

```
int Fl_Flex::fixed (
    Fl_Widget * w) const
```

Return whether the given widget has a fixed size or resizes dynamically.

Parameters

in	<i>w</i>	widget

Returns

whether the widget has a fixed size

Return values

1	the widget has a fixed size
0	the widget resizes dynamically

11.41.4.6 fixed() [3/3]

```
void Fl_Flex::fixed (
    Fl_Widget * child,
    int size)
```

Set the horizontal or vertical size of a child widget.

This sets either the width or height of a child widget, depending on the `type()` of the `Fl_Flex` container (`Fl_Flex::HORIZONTAL` or `Fl_Flex::VERTICAL`). The other dimension is set to the full width or height of the `Fl_Flex` widget minus border and margin sizes.

This can be used to set a fixed widget width or height of children of `Fl_Flex` so they are not resized dynamically.

If `size` is 0 (zero) or negative the widget size is reset to flexible size.

Parameters

in	<i>child</i>	widget to be affected
in	<i>size</i>	width (<code>Fl_Flex::HORIZONTAL</code>) or height (<code>Fl_Flex::VERTICAL</code>)

11.41.4.7 gap() [1/2]

```
int Fl_Flex::gap () const [inline]
```

Return the gap size of the widget.

Returns

gap size between all child widgets.

11.41.4.8 gap() [2/2]

```
void Fl_Flex::gap (
    int g) [inline]
```

Set the gap size of the widget.

The gap size is some free space **between** child widgets. The size must be ≥ 0 . Negative values are clamped to 0.

Parameters

in	<i>g</i>	gap size

11.41.4.9 horizontal()

```
int Fl_Flex::horizontal () const [inline]
```

Returns non-zero (true) if [Fl_Flex](#) alignment is horizontal (row mode).

Returns

non-zero if [Fl_Flex](#) alignment is horizontal

Return values

1	if <code>type() == Fl_Flex::HORIZONTAL</code>
0	if <code>type() == Fl_Flex::VERTICAL</code>

See class [Fl_Flex](#) documentation for details.

11.41.4.10 layout()

```
void Fl_Flex::layout ()
```

Calculates the layout of the widget and redraws it.

If you change widgets in the [Fl_Flex](#) container you should call this method to force recalculation of child widget sizes and positions. This can be useful (necessary) if you [hide\(\)](#), [show\(\)](#), [add\(\)](#) or [remove\(\)](#) children.

Call this method if you need to recalculate widget positions for usage in an algorithm that places widgets at certain positions or when you need to display (show) or hide one or more children depending on the current layout (for instance a side bar).

This method also calls [redraw\(\)](#) on the [Fl_Flex](#) widget.

11.41.4.11 margin() [1/4]

```
int Fl_Flex::margin () const [inline]
```

Returns the left margin size of the widget.

This returns the **left** margin of the widget which is not necessarily the same as all other margins.

Note

This method is useful if you never set different margin sizes.

See also

`int margin(int *left, int *top, int *right, int *bottom)` to get all four [margin](#) values.

Returns

size of left margin.

11.41.4.12 margin() [2/4]

```
int Fl_Flex::margin (
    int * left,
    int * top,
    int * right,
    int * bottom) const [inline]
```

Returns all (four) margin sizes of the widget.

All margin sizes are returned in the given arguments. If any argument is `NULL` the respective value is not returned.

Parameters

in	<i>left</i>	returns left margin if not <code>NULL</code>
in	<i>top</i>	returns top margin if not <code>NULL</code>

in	<i>right</i>	returns right margin if not NULL
in	<i>bottom</i>	returns bottom margin if not NULL

Returns

whether all margins are equal

Return values

1	all margins have the same size
0	at least one margin has a different size

11.41.4.13 margin() [3/4]

```
void Fl_Flex::margin (
    int left,
    int top,
    int right,
    int bottom) [inline]
```

Set the margin sizes at all four edges of the [Fl_Flex](#) widget.

The margin is the free space inside the widget border **around** all child widgets.

You must use all four parameters of this method to set the four margins in the order `left, top, right, bottom`.

Negative values are set to 0 (zero).

To set all margins to equal sizes, use `margin(int m)` which sets all four margins to the same size.

Parameters

in	<i>left, top, right, bottom</i>	margin sizes, must be ≥ 0

See also

[margin\(int, int\)](#)

11.41.4.14 margin() [4/4]

```
void Fl_Flex::margin (
    int m,
    int g = -1) [inline]
```

Set the margin and optionally the gap size of the widget.

This method can be used to set both the margin and the gap size.

If you don't use the second parameter `g` or supply a negative value the gap size is not changed.

The margin is the free space inside the widget border **around** all child widgets.

This method sets the margin to the same size at all four edges of the [Fl_Flex](#) widget.

The gap size `g` is the free space **between** child widgets. Negative values do not change the gap value. This is the default if this argument is omitted.

Parameters

in	<i>m</i>	margin size, must be ≥ 0
in	<i>g</i>	gap size (ignored, if negative)

See also

[gap\(int\)](#)

11.41.4.15 need_layout() [1/2]

```
bool Fl_Flex::need_layout () const [inline]
```

Returns whether layout calculation is required.

This should rarely be needed by user code. Used internally in [draw\(\)](#).

11.41.4.16 need_layout() [2/2]

```
void Fl_Flex::need_layout (
    int set) [inline]
```

Set or reset the request to calculate the layout of children.

This is intended for internal use but can also be used by user code to request layout calculation before the widget is drawn.

Call this if you changed attributes or sizes of children to ensure that the layout is calculated properly. Changing other [Fl_Flex](#) attributes or resizing the widget does this automatically.

Note

Never call this with 'set == 0' because this would defeat its purpose to recalculate the layout before the widget is drawn.

11.41.4.17 on_remove()

```
void Fl_Flex::on_remove (
    int index) [protected], [virtual]
```

Allow derived groups to act when a child widget is removed from the group.

Widgets derived from [Fl_Group](#) may store additional data for their children. Overriding this method will allow derived classes to remove these data structures just before the child is removed.

Parameters

<i>index</i>	remove the child at this position

Reimplemented from [Fl_Group](#).

11.41.4.18 resize()

```
void Fl_Flex::resize (
    int x,
    int y,
    int w,
    int h) [virtual]
```

Resize the container and calculate all child positions and sizes.

Parameters

in	<i>x,y</i>	position
in	<i>w,h</i>	width and height

Reimplemented from [Fl_Widget](#).

11.41.4.19 spacing() [1/2]

```
int Fl_Flex::spacing () const [inline]
```

Gets the number of extra pixels of blank space that are added between the children.

This method is the same as 'int [gap\(\)](#)' and is defined to enable using [Fl_Flex](#) as a drop-in replacement of [Fl_Pack](#).

See also

int [gap\(\)](#)

11.41.4.20 spacing() [2/2]

```
void Fl_Flex::spacing (
    int i) [inline]
```

Sets the number of extra pixels of blank space that are added between the children.

This method is the same as '[gap\(int\)](#)' and is defined to enable using [Fl_Flex](#) as a drop-in replacement of [Fl_Pack](#).

See also

```
void gap\(int\)
```

The documentation for this class was generated from the following files:

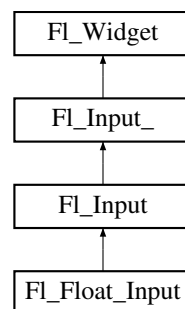
- [Fl_Flex.H](#)
- [Fl_Flex.cxx](#)

11.42 Fl_Float_Input Class Reference

The [Fl_Float_Input](#) class is a subclass of [Fl_Input](#) that only allows the user to type floating point numbers (sign, digits, decimal point, more digits, 'E' or 'e', sign, digits).

```
#include <Fl_Float_Input.H>
```

Inheritance diagram for [Fl_Float_Input](#):

**Public Member Functions**

- [Fl_Float_Input](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [Fl_Float_Input](#) widget using the given position, size, and label string.

Public Member Functions inherited from [Fl_Input](#)

- [Fl_Input](#) (int, int, int, int, const char *l=0)
Creates a new [Fl_Input](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [Fl_Input_](#)

- int [append](#) (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool [can_redo](#) () const
Check if there is a redo action available.
- bool [can_undo](#) () const
Check if the last operation can be undone.
- int [copy](#) (int clipboard)
Put the current selection into the clipboard.
- int [copy_cuts](#) ()

- Copies the yank buffer to the clipboard.*

 - `FL_Color cursor_color ()` const

Gets the color of the cursor.
- `void cursor_color (FL_Color n)`

Sets the color of the cursor.
- `int cut ()`

Deletes the current selection.
- `int cut (int a, int b)`

*Deletes all characters between index *a* and *b*.*
- `int cut (int n)`

*Deletes the next *n* bytes rounded to characters before or after the cursor.*
- `double dvalue ()` const

Returns the widget text interpreted as a floating point number.
- `FL_Input_ (int, int, int, int, const char *=0)`

*Creates a new *FL_Input_* widget.*
- `unsigned int index (int i)` const

*Returns the character at index *i*.*
- `int input_type ()` const

Gets the input field type.
- `void input_type (int t)`

Sets the input field type.
- `int insert (const char *t, int l=0)`

Inserts text at the cursor position.
- `int insert_position ()` const

Gets the position of the text cursor.
- `int insert_position (int p)`

Sets the cursor position and mark.
- `int insert_position (int p, int m)`

Sets the index for the cursor and mark.
- `int ivalue ()` const

Returns the widget text interpreted as a signed integer.
- `int mark ()` const

Gets the current selection mark.
- `int mark (int m)`

Sets the current selection mark.
- `int maximum_size ()` const

Gets the maximum length of the input field in characters.
- `void maximum_size (int m)`

Sets the maximum length of the input field in characters.
- `int position ()` const
- `int position (int p)`
- `int position (int p, int m)`
- `int readonly ()` const

Gets the read-only state of the input field.
- `void readonly (int b)`

Sets the read-only state of the input field.
- `int redo ()`

Redo previous undo operation.
- `int replace (int b, int e, const char *text, int ilen=0)`

*Deletes text from *b* to *e* and inserts the new string *text*.*
- `void resize (int, int, int, int) FL_OVERRIDE`

- Changes the size of the widget.*
- `int shortcut () const`
 - Return the shortcut key associated with this widget.*
- `void shortcut (int s)`
 - Sets the shortcut key associated with this widget.*
- `int size () const`
 - Returns the number of bytes in `value()`.*
- `void size (int W, int H)`
 - Sets the width and height of this widget.*
- `int static_value (const char *)`
 - Changes the widget text.*
- `int static_value (const char *, int)`
 - Changes the widget text.*
- `int tab_nav () const`
 - Gets whether the Tab key causes focus navigation in multiline input fields or not.*
- `void tab_nav (int val)`
 - Sets whether the Tab key does focus navigation, or inserts tab characters into `FI_Multiline_Input`.*
- `FI_Color textcolor () const`
 - Gets the color of the text in the input field.*
- `void textcolor (FI_Color n)`
 - Sets the color of the text in the input field.*
- `FI_Font textfont () const`
 - Gets the font of the text in the input field.*
- `void textfont (FI_Font s)`
 - Sets the font of the text in the input field.*
- `FI_Fontsize textsize () const`
 - Gets the size of the text in the input field.*
- `void textsize (FI_Fontsize s)`
 - Sets the size of the text in the input field.*
- `int undo ()`
 - Undoes previous changes to the text buffer.*
- `const char * value () const`
 - Returns the text displayed in the widget.*
- `int value (const char *)`
 - Changes the widget text.*
- `int value (const char *, int)`
 - Changes the widget text.*
- `int value (double value)`
 - Changes the widget text to a floating point number ("%g").*
- `int value (int value)`
 - Changes the widget text to a signed integer number.*
- `int wrap () const`
 - Gets the word wrapping state of the input field.*
- `void wrap (int b)`
 - Sets the word wrapping state of the input field.*
- `~FI_Input_ ()`
 - Destroys the widget.*

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*
- void `clear_damage` (uchar c=0)
- Clears or sets the damage flags.*
- void `clear_output` ()
- Sets a widget to accept input.*
- void `clear_visible` ()
- Hides the widget.*
- void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*
- `FL_Color color` () const
- Gets the background color of the widget.*
- void `color` (FL_Color bg)
- Sets the background color of the widget.*
- void `color` (FL_Color bg, FL_Color sel)
- Sets the background and selection color of the widget.*
- `FL_Color color2` () const
- For back compatibility only.*
- void `color2` (unsigned a)
- For back compatibility only.*
- int `contains` (const FL_Widget *w) const
- Checks if w is a child of this widget.*
- void `copy_label` (const char *new_label)
- Sets the current label.*
- void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*
- `uchar damage` () const
- Returns non-zero if `draw()` needs to be called.*
- void `damage` (uchar c)
- Sets the damage bits for the widget.*
- void `damage` (uchar c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int `damage_resize` (int, int, int, int)
- Internal use only.*
- void `deactivate` ()
- Deactivates the widget.*
- `FL_Image * deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*
- const `FL_Image * deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void `deimage` (FL_Image &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void `deimage` (FL_Image *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)
- Calls the widget callback function with default arguments.*

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([FI_Labeltype](#) a)

Sets the label type.

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.

 - bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.

 - void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.

 - unsigned int [output](#) () const

Returns if a widget is used for output only.

 - [FI_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.

 - void [parent](#) ([FI_Group](#) *p)

Internal use only - "for hacks only".

 - void [position](#) (int X, int Y)

Repositions the window or widget.

 - void [redraw](#) ()

Schedules the drawing of the widget.

 - void [redraw_label](#) ()

Schedules the drawing of the label.

 - [FI_Color](#) [selection_color](#) () const

Gets the selection color.

 - void [selection_color](#) ([FI_Color](#) a)

Sets the selection color.

 - void [set_active](#) ()

Marks the widget as active without sending events or changing focus.

 - void [set_changed](#) ()

Marks the value of the widget as changed.

 - void [set_output](#) ()

Sets a widget to output only.

 - void [set_visible](#) ()

Makes the widget visible.

 - void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.

 - int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.

 - void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.

 - virtual void [show](#) ()

Makes a widget visible.

 - void [size](#) (int W, int H)

Changes the size of the widget.

 - int [take_focus](#) ()

Gives the widget the keyboard focus.

 - unsigned int [takeevents](#) () const

Returns if the widget is able to take events.

 - int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

 - const char * [tooltip](#) () const

Gets the current tooltip text.

- void **tooltip** (const char *text)
Sets the current tooltip text.
- **Fl_Window** * **top_window** () const
Returns a pointer to the top-level window for the widget.
- **Fl_Window** * **top_window_offset** (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- **uchar** **type** () const
Gets the widget type.
- void **type** (**uchar** t)
Sets the widget type.
- int **use_accents_menu** ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const
Gets the user data for this widget.
- void **user_data** (**Fl_Callback_User_Data** *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **Fl_When** **when** () const
Returns the conditions under which the callback is called.
- void **when** (**uchar** i)
Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~Fl_Widget** ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void [default_callback](#) (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Static Public Attributes inherited from FI_Input

- static const char * [copy_menu_text](#) = "Copy"
[this text may be customized at run-time]
- static const char * [cut_menu_text](#) = "Cut"
[this text may be customized at run-time]
- static const char * [paste_menu_text](#) = "Paste"
[this text may be customized at run-time]

Protected Types inherited from FI_Widget

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from FI_Input

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- int [handle_key](#) ()
Handles a keystroke.
- int [handle_rmb](#) ()
Handle right mouse button down events.

Protected Member Functions inherited from FI_Input_

- int [apply_undo](#) ()
Apply the current undo/redo operation.
- void [drawtext](#) (int, int, int, int)
Draws the text in the passed bounding box.
- void [drawtext](#) (int, int, int, int, bool draw_active)
Draws the text in the passed bounding box.

- void **handle_mouse** (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int **handletext** (int e, int, int, int, int)
Handles all kinds of text field related events.
- int **line_end** (int i) const
Finds the end of a line.
- int **line_start** (int i) const
Finds the start of a line.
- int **linesPerPage** ()
- void **maybe_do_callback** (FI_Callback_Reason reason=FL_REASON_UNKNOWN)
- int **up_down_position** (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int **word_end** (int i) const
Finds the end of a word.
- int **word_start** (int i) const
Finds the start of a word.
- int **xscroll** () const
- int **yscroll** () const
- void **yscroll** (int yOffset)

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)

Internal use only.

- void **x** (int v)

Internal use only.

- void **y** (int v)

Internal use only.

11.42.1 Detailed Description

The [Fl_Float_Input](#) class is a subclass of [Fl_Input](#) that only allows the user to type floating point numbers (sign, digits, decimal point, more digits, 'E' or 'e', sign, digits).

11.42.2 Constructor & Destructor Documentation

11.42.2.1 Fl_Float_Input()

```
Fl_Float_Input::Fl_Float_Input (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Float_Input](#) widget using the given position, size, and label string.

The default boxtype is FL_DOWN_BOX.

Inherited destructor destroys the widget and any value associated with it.

The documentation for this class was generated from the following files:

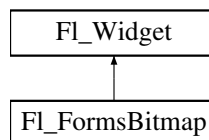
- Fl_Float_Input.H
- Fl_Input.cxx

11.43 Fl_FormsBitmap Class Reference

Forms compatibility Bitmap Image Widget.

```
#include <Fl_FormsBitmap.H>
```

Inheritance diagram for Fl_FormsBitmap:



Public Member Functions

- [Fl_Bitmap](#) * **bitmap** () const
Gets a the current associated [Fl_Bitmap](#) objects.
- void **bitmap** ([Fl_Bitmap](#) *B)
Sets a new bitmap.
- **Fl_FormsBitmap** ([Fl_Boxtype](#), int, int, int, int, const char *l=0)
Creates a bitmap widget from a box type, position, size and optional label specification.
- void **set** (int W, int H, const [uchar](#) *bits)
Sets a new bitmap bits with size W,H.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*
- void `clear_damage` (uchar c=0)
- Clears or sets the damage flags.*
- void `clear_output` ()
- Sets a widget to accept input.*
- void `clear_visible` ()
- Hides the widget.*
- void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*
- `FL_Color` `color` () const
- Gets the background color of the widget.*
- void `color` (`FL_Color` bg)
- Sets the background color of the widget.*
- void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*
- `FL_Color` `color2` () const
- For back compatibility only.*
- void `color2` (unsigned a)
- For back compatibility only.*
- int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*
- void `copy_label` (const char *new_label)
- Sets the current label.*
- void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*
- `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*
- void `damage` (uchar c)
- Sets the damage bits for the widget.*
- void `damage` (uchar c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int `damage_resize` (int, int, int, int)
- Internal use only.*
- void `deactivate` ()
- Deactivates the widget.*
- `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*
- const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with default arguments.*

- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual int `handle` (int event)
Handles the specified event.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)

- Sets the font size in pixels.*

 - [FI_Labeltype labeltype](#) () const
- Gets the label type.*

 - void [labeltype](#) ([FI_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const
- Sets width ww and height hh accordingly with the label size.*

 - bool [needs_keyboard](#) () const
- Returns whether this widget needs a keyboard.*

 - void [needs_keyboard](#) (bool needs)
- Sets whether this widget needs a keyboard.*

 - unsigned int [output](#) () const
- Returns if a widget is used for output only.*

 - [FI_Group](#) * [parent](#) () const
- Returns a pointer to the parent widget.*

 - void [parent](#) ([FI_Group](#) *p)
- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
- Repositions the window or widget.*

 - void [redraw](#) ()
- Schedules the drawing of the widget.*

 - void [redraw_label](#) ()
- Schedules the drawing of the label.*

 - virtual void [resize](#) (int x, int y, int w, int h)
- Changes the size or position of the widget.*

 - [FI_Color](#) [selection_color](#) () const
- Gets the selection color.*

 - void [selection_color](#) ([FI_Color](#) a)
- Sets the selection color.*

 - void [set_active](#) ()
- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

 - void [set_output](#) ()
- Sets a widget to output only.*

 - void [set_visible](#) ()
- Makes the widget visible.*

 - void [set_visible_focus](#) ()
- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void [shortcut_label](#) (int value)
- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - virtual void [show](#) ()
- Makes a widget visible.*

 - void [size](#) (int W, int H)
- Changes the size of the widget.*

 - int [take_focus](#) ()
- Gives the widget the keyboard focus.*

 - unsigned int [takeevents](#) () const
- Returns if the widget is able to take events.*

- `int test_shortcut ()`
Returns true if the widget's label contains the entered '&x' shortcut.
- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- void **draw** () [FL_OVERRIDE](#)
Draws the bitmap and its associated box.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members**Static Public Member Functions inherited from [FI_Widget](#)**

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.43.1 Detailed Description

Forms compatibility Bitmap Image Widget.

11.43.2 Member Function Documentation

11.43.2.1 `draw()`

```
void Fl_FormsBitmap::draw (
    void ) [protected], [virtual]
```

Draws the bitmap and its associated box.

Implements [Fl_Widget](#).

11.43.2.2 `set()`

```
void Fl_FormsBitmap::set (
    int W,
    int H,
    const uchar * bits)
```

Sets a new bitmap bits with size W,H.

Deletes the previous one.

The documentation for this class was generated from the following files:

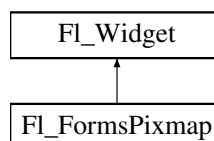
- `Fl_FormsBitmap.H`
- `forms_bitmap.cxx`

11.44 [Fl_FormsPixmap](#) Class Reference

Forms pixmap drawing routines.

```
#include <Fl_FormsPixmap.H>
```

Inheritance diagram for [Fl_FormsPixmap](#):



Public Member Functions

- [FL_FormsPixmap](#) ([FL_Boxtype](#) t, int X, int Y, int W, int H, const char *L=0)
Creates a new [FL_FormsPixmap](#) widget using the given box type, position, size and label string.
- [FL_Pixmap](#) * [Pixmap](#) () const
Get the internal pixmap pointer.
- void [Pixmap](#) ([FL_Pixmap](#) *B)
Set the internal pixmap pointer to an existing pixmap.
- void [set](#) (char *const *bits)
Set/create the internal pixmap using raw data.

Public Member Functions inherited from [FL_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FL_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FL_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FL_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).
- virtual class [FL_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FL_Group](#) * [as_group](#) ()
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- virtual [FL_Group](#) const * [as_group](#) () const
- virtual [FL_Window](#) * [as_window](#) ()
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).
- virtual [FL_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FL_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FL_Boxtype](#) new_box)
Sets the box type for the widget.
- [FL_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.

- void `callback` (`FI_Callback` *cb)
Sets the current callback function for the widget.
- void `callback` (`FI_Callback` *cb, `FI_Callback_User_Data` *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void `callback` (`FI_Callback` *cb, void *p)
Sets the current callback function and data for the widget.
- void `callback` (`FI_Callback0` *cb)
Sets the current callback function for the widget.
- void `callback` (`FI_Callback1` *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int `changed` () const
Checks if the widget value changed since the last callback.
- void `clear_active` ()
Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()
Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)
Clears or sets the damage flags.
- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FI_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FI_Color` bg)
Sets the background color of the widget.
- void `color` (`FI_Color` bg, `FI_Color` sel)
Sets the background and selection color of the widget.
- `FI_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FI_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- uchar `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)
Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FI_Image` * `deimage` ()

- Gets the image that is used as part of the widget label when in the inactive state.*
- const [FL_Image](#) * [deimage](#) () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void [deimage](#) ([FL_Image](#) &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void [deimage](#) ([FL_Image](#) *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int [deimage_bound](#) () const
- Returns whether the inactive image is managed by the widget.*
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- Calls the widget callback function with default arguments.*
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- Calls the widget callback function with arbitrary arguments.*
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- Calls the widget callback function with arbitrary arguments.*
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int [h](#) () const
- Gets the widget height.*
- virtual int [handle](#) (int event)
- Handles the specified event.*
- virtual void [hide](#) ()
- Makes a widget invisible.*
- int [horizontal_label_margin](#) ()
- Get the spacing between the label and the horizontal edge of the widget.*
- void [horizontal_label_margin](#) (int px)
- Set the spacing between the label and the horizontal edge of the widget.*
- [FL_Image](#) * [image](#) ()
- Gets the image that is used as part of the widget label when in the active state.*
- const [FL_Image](#) * [image](#) () const
- Gets the image that is used as part of the widget label when in the active state.*
- void [image](#) ([FL_Image](#) &img)
- Sets the image to use as part of the widget label when in the active state.*
- void [image](#) ([FL_Image](#) *img)
- Sets the image to use as part of the widget label when in the active state.*
- int [image_bound](#) () const
- Returns whether the image is managed by the widget.*
- int [inside](#) (const [FL_Widget](#) *wgt) const
- Checks if this widget is a child of wgt.*
- int [is_label_copied](#) () const
- Returns whether the current label was assigned with [copy_label\(\)](#).*
- const char * [label](#) () const
- Gets the current label text.*
- void [label](#) (const char *text)
- Sets the current label pointer.*
- void [label](#) ([FL_Labeltype](#) a, const char *b)
- Shortcut to set the label text and type in one call.*
- int [label_image_spacing](#) ()
- Return the gap size between the label and the image.*
- void [label_image_spacing](#) (int gap)
- Set the gap between the label and the image in pixels.*

- [FL_Color labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize labelsiz](#)e () const
Gets the font size in pixels.
- void [labelsiz](#)e ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group * parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)
Changes the size or position of the widget.
- [FL_Color selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

Gets the current tooltip text.
- void [tooltip](#) (const char *text)

Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const

Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const

Gets the widget type.
- void [type](#) ([uchar](#) t)

Sets the widget type.
- int [use_accents_menu](#) ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const

Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)

Sets the user data for this widget.
- void [user_data](#) (void *v)

Sets the user data for this widget.
- int [vertical_label_margin](#) ()

Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const

Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const

Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)

Modifies keyboard focus navigation.
- int [visible_r](#) () const

Returns whether a widget and all its parents are visible.
- int [w](#) () const

Gets the widget width.
- [Fl_When](#) [when](#) () const

Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)

Sets the flags used to decide when a callback is called.

- `FL_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~FL_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from `FL_Widget`

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const`
Draws a focus box for the widget at the given position and size.
- `void draw_label () const`
Draws the widget's label at the defined label position.
- `void draw_label (int, int, int, int) const`
Draws the label in an arbitrary bounding box.
- `FL_Widget (int x, int y, int w, int h, const char *label=0L)`
Creates a widget at the given position and size.
- `unsigned int flags () const`
Gets the widget flags mask.
- `void h (int v)`
Internal use only.
- `void set_flag (unsigned int c)`
Sets a flag in the flags mask.
- `void w (int v)`
Internal use only.
- `void x (int v)`
Internal use only.
- `void y (int v)`
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from Fl_Widget

- static void `default_callback` (`Fl_Widget *widget`, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from Fl_Widget

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
,
`OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
,
`MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
,
`GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

11.44.1 Detailed Description

Forms pixmap drawing routines.

11.44.2 Constructor & Destructor Documentation

11.44.2.1 Fl_FormsPixmap()

```
Fl_FormsPixmap::Fl_FormsPixmap (
    Fl_Boxtype t,
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new `Fl_FormsPixmap` widget using the given box type, position, size and label string.

Parameters

in	<i>t</i>	box type
in	<i>X,Y,W,H</i>	position and size
in	<i>L</i>	widget label, default is no label

11.44.3 Member Function Documentation

11.44.3.1 draw()

```
void Fl_FormsPixmap::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.44.3.2 Pixmap()

```
void Fl_FormsPixmap::Pixmap (
    Fl_Pixmap * B) [inline]
```

Set the internal pixmap pointer to an existing pixmap.

Parameters

in	<i>B</i>	existing pixmap

11.44.3.3 set()

```
void Fl_FormsPixmap::set (
    char *const * bits)
```

Set/create the internal pixmap using raw data.

Parameters

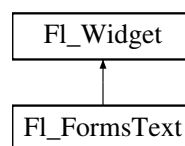
in	<i>bits</i>	raw data

The documentation for this class was generated from the following files:

- [Fl_FormsPixmap.H](#)
- [forms_pixmap.cxx](#)

11.45 Fl_FormsText Class Reference

Inheritance diagram for [Fl_FormsText](#):



Public Member Functions

- [Fl_FormsText](#) ([Fl_Boxtype](#) b, int X, int Y, int W, int H, const char *l=0)

Public Member Functions inherited from [Fl_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const

- Returns whether the widget is active.*

 - int `active_r` () const
- Returns whether the widget and all of its parents are active.*

 - `FL_Align align` () const
- Gets the label alignment.*

 - void `align` (`FL_Align alignment`)
- Sets the label alignment.*

 - long `argument` () const
- Gets the current user data (long) argument that is passed to the callback function.*

 - void `argument` (long v)
- Sets the current user data (long) argument that is passed to the callback function.*

 - virtual class `FL_Gl_Window` * `as_gl_window` ()
- Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.*

 - virtual class `FL_Gl_Window` const * `as_gl_window` () const
- Returns an `FL_Group` pointer if this widget is an `FL_Group`.*

 - virtual `FL_Group` * `as_group` ()
- Returns an `FL_Window` pointer if this widget is an `FL_Window`.*

 - virtual `FL_Group` const * `as_group` () const
- Returns an `FL_Window` pointer if this widget is an `FL_Window`.*

 - virtual `FL_Window` * `as_window` ()
- Returns an `FL_Window` pointer if this widget is an `FL_Window`.*

 - virtual `FL_Window` const * `as_window` () const
- Sets the image to use as part of the widget label when in the inactive state.*

 - void `bind_deimage` (`FL_Image *img`)
- Sets the image to use as part of the widget label when in the inactive state.*

 - void `bind_deimage` (int f)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*

 - void `bind_image` (`FL_Image *img`)
- Sets the image to use as part of the widget label when in the active state.*

 - void `bind_image` (int f)
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*

 - `FL_Boxtype box` () const
- Gets the box type of the widget.*

 - void `box` (`FL_Boxtype new_box`)
- Sets the box type for the widget.*

 - `FL_Callback_p callback` () const
- Gets the current callback function for the widget.*

 - void `callback` (`FL_Callback *cb`)
- Sets the current callback function for the widget.*

 - void `callback` (`FL_Callback *cb`, `FL_Callback_User_Data *p`, bool auto_free)
- Sets the current callback function and managed user data for the widget.*

 - void `callback` (`FL_Callback *cb`, void *p)
- Sets the current callback function and data for the widget.*

 - void `callback` (`FL_Callback0 *cb`)
- Sets the current callback function for the widget.*

 - void `callback` (`FL_Callback1 *cb`, long p=0)
- Sets the current callback function for the widget.*

 - unsigned int `changed` () const
- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()
- Marks the widget as inactive without sending events or changing focus.*

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*

 - void `clear_damage` (uchar c=0)

- Clears or sets the damage flags.*

 - void `clear_output` ()
- Sets a widget to accept input.*

 - void `clear_visible` ()
- Hides the widget.*

 - void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*

 - `FL_Color` `color` () const
- Gets the background color of the widget.*

 - void `color` (`FL_Color` bg)
- Sets the background color of the widget.*

 - void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*

 - `FL_Color` `color2` () const
- For back compatibility only.*

 - void `color2` (unsigned a)
- For back compatibility only.*

 - int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*

 - void `copy_label` (const char *new_label)
- Sets the current label.*

 - void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*

 - `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*

 - void `damage` (`uchar` c)
- Sets the damage bits for the widget.*

 - void `damage` (`uchar` c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int `damage_resize` (int, int, int, int)
- Internal use only.*

 - void `deactivate` ()
- Deactivates the widget.*

 - `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*

 - const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*

 - void `deimage` (`FL_Image` &img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - void `deimage` (`FL_Image` *img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*

 - void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with default arguments.*

 - void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*

 - void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*

 - void `draw_label` (int, int, int, int, `FL_Align`) const
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*

- `int h () const`
Gets the widget height.
- `virtual int handle (int event)`
Handles the specified event.
- `virtual void hide ()`
Makes a widget invisible.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FL_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FL_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FL_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FL_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FL_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FL_Color c)`
Sets the label color.
- `FL_Font labelfont () const`
Gets the font to use.
- `void labelfont (FL_Font f)`
Sets the font to use.
- `FL_Fonsize labelsize () const`
Gets the font size in pixels.
- `void labelsize (FL_Fonsize pix)`
Sets the font size in pixels.
- `FL_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (FL_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`

- Sets width ww and height hh accordingly with the label size.*

 - bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FI_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FI_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- virtual void `resize` (int x, int y, int w, int h)

Changes the size or position of the widget.
- `FI_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`FI_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.
- void `set_visible` ()

Makes the widget visible.
- void `set_visible_focus` ()

Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()

Makes a widget visible.
- void `size` (int W, int H)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const

Gets the current tooltip text.
- void `tooltip` (const char *text)

Sets the current tooltip text.

- `Fl_Window * top_window ()` const
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff)` const
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type ()` const
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data ()` const
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible ()` const
Returns whether a widget is visible.
- `unsigned int visible_focus ()` const
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r ()` const
Returns whether a widget and all its parents are visible.
- `int w ()` const
Gets the widget width.
- `Fl_When when ()` const
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window ()` const
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x ()` const
Gets the widget position in its window.
- `int y ()` const
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw ()` **FL_OVERRIDE**
Draws the widget.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.45.1 Member Function Documentation

11.45.1.1 [draw\(\)](#)

```
void Fl_FormsText::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl\_Widget *s = &scrollbar; // scrollbar is an embedded Fl\_Scrollbar
s->draw();                 // calls Fl\_Scrollbar::draw\(\)
```

Implements [Fl_Widget](#).

The documentation for this class was generated from the following file:

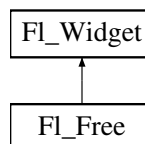
- forms.H

11.46 Fl_Free Class Reference

Emulation of the Forms "free" widget.

```
#include <Fl_Free.H>
```

Inheritance diagram for [Fl_Free](#):



Public Member Functions

- [Fl_Free](#) ([uchar](#) t, int X, int Y, int W, int H, const char *L, [FL_HANDLEPTR](#) hdl)
Create a new [Fl_Free](#) widget with type, position, size, label and handler.
- int [handle](#) (int e) [FL_OVERRIDE](#)
Handles the specified event.
- ~[Fl_Free](#) ()
The destructor will call the handle function with the event [FL_FREE_MEM](#).

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color color` () const

Gets the background color of the widget.
- void `color` (FL_Color bg)

Sets the background color of the widget.
- void `color` (FL_Color bg, FL_Color sel)

Sets the background and selection color of the widget.
- `FL_Color color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const FL_Widget *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void `labeltype` (`FL_Labeltype` a)
- Sets the label type.*

 - void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FL_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FL_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- virtual void `resize` (int x, int y, int w, int h)

Changes the size or position of the widget.
- `FL_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`FL_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.
- void `set_visible` ()

Makes the widget visible.
- void `set_visible_focus` ()

Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()

Makes a widget visible.
- void `size` (int W, int H)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
 ,
[OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
 ,
[MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
 ,
[GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
 = 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.46.1 Detailed Description

Emulation of the Forms "free" widget.

This emulation allows the free demo to run, and appears to be useful for porting programs written in Forms which use the free widget or make subclasses of the Forms widgets.

There are five types of free, which determine when the handle function is called:

- [FL_NORMAL_FREE](#) normal event handling.
- [FL_SLEEPING_FREE](#) deactivates event handling (widget is inactive).
- [FL_INPUT_FREE](#) accepts [FL_FOCUS](#) events.
- [FL_CONTINUOUS_FREE](#) sets a timeout callback 100 times a second and provides an [FL_STEP](#) event. This has obvious detrimental effects on machine performance.
- [FL_ALL_FREE](#) same as [FL_INPUT_FREE](#) and [FL_CONTINUOUS_FREE](#).

11.46.2 Constructor & Destructor Documentation

11.46.2.1 [Fl_Free\(\)](#)

```
Fl_Free::Fl_Free (
    uchar t,
    int X,
    int Y,
    int W,
    int H,
    const char * L,
    FL_HANDLEPTR hdl)
```

Create a new [Fl_Free](#) widget with type, position, size, label and handler.

Parameters

in	<i>t</i>	type
in	<i>X,Y,W,H</i>	position and size
in	<i>L</i>	widget label
in	<i>hdl</i>	handler function

The constructor takes both the type and the handle function. The handle function should be declared as follows:

```
int handle_function(Fl\_Widget *w,
    int event,
    float event_x,
```

```
float    event_y,
char     key)
```

This function is called from the [handle\(\)](#) method in response to most events, and is called by the [draw\(\)](#) method.

The event argument contains the event type:

```
// old event names for compatibility:
#define FL_MOUSE      FL_DRAG
#define FL_DRAW       0
#define FL_STEP       9
#define FL_FREEMEM    12
#define FL_FREEZE     FL_UNMAP
#define FL_THAW       FL_MAP
```

11.46.3 Member Function Documentation

11.46.3.1 draw()

```
void Fl_Free::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.46.3.2 handle()

```
int Fl_Free::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

The documentation for this class was generated from the following files:

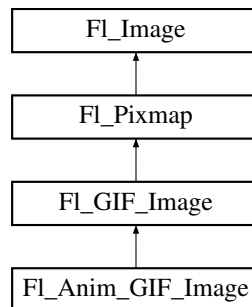
- [Fl_Free.H](#)
- [forms_free.cxx](#)

11.47 FI_GIF_Image Class Reference

The [FI_GIF_Image](#) class supports loading, caching, and drawing of Compuserve GIFSM images.

```
#include <FI_GIF_Image.H>
```

Inheritance diagram for [FI_GIF_Image](#):



Classes

- struct [GIF_FRAME](#)

Public Member Functions

- [FI_GIF_Image](#) (const char *filename)
This constructor loads a GIF image from the given file.
- [FI_GIF_Image](#) (const char *imagename, const unsigned char *data)
This constructor loads a GIF image from memory (deprecated).
- [FI_GIF_Image](#) (const char *imagename, const unsigned char *data, const size_t length)
This constructor loads a GIF image from memory.

Public Member Functions inherited from [FI_Pixmap](#)

- int **cache_h** ()
- int **cache_w** ()
- void [color_average](#) ([FI_Color](#) c, float i) [FL_OVERRIDE](#)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [FI_Image](#) * **copy** () const
- [FI_Image](#) * **copy** (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void [desaturate](#) () [FL_OVERRIDE](#)
The [desaturate\(\)](#) method converts an image to grayscale.
- void **draw** (int X, int Y)
- void **draw** (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [FI_Pixmap](#) (char *const *D)
The constructors create a new pixmap from the specified XPM data.
- [FI_Pixmap](#) (const char *const *D)
The constructors create a new pixmap from the specified XPM data.
- [FI_Pixmap](#) (const uchar *const *D)
The constructors create a new pixmap from the specified XPM data.
- [FI_Pixmap](#) (uchar *const *D)
The constructors create a new pixmap from the specified XPM data.
- void **label** ([FI_Menu_Item](#) *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.

- void **label** (FI_Widget *w) **FL_OVERRIDE**
This method is an obsolete way to set the image attribute of a widget or menu item.
- void **uncache** () **FL_OVERRIDE**
If the image has been cached for display, delete the cache data.
- virtual ~**FI_Pixmap** ()
The destructor frees all memory and server resources that are used by the pixmap.

Public Member Functions inherited from FI_Image

- virtual class FI_Shared_Image * **as_shared_image** ()
Returns whether an image is an FI_Shared_Image or not.
- FI_Image * **copy** () const
Creates a copy of the image in the same size.
- int **count** () const
Returns the number of data values associated with the image.
- int **d** () const
Returns the image depth.
- const char *const * **data** () const
Returns a pointer to the current image data array.
- int **data_h** () const
Returns the height of the image data.
- int **data_w** () const
Returns the width of the image data.
- void **draw** (int X, int Y)
Draws the image to the current drawing surface.
- int **fail** () const
Returns a value that is not 0 if there is currently no image available.
- FI_Image (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int **h** () const
Returns the current image drawing height in FLTK units.
- void **inactive** ()
*The **inactive()** method calls **color_average(FI_BACKGROUND_COLOR, 0.33f)** to produce an image that appears grayed out.*
- int **ld** () const
Returns the current line data size in bytes.
- virtual void **release** ()
Releases an FI_Image - the same as 'delete this'.
- virtual void **scale** (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- int **w** () const
Returns the current image drawing width in FLTK units.
- virtual ~**FI_Image** ()
The destructor is a virtual method that frees all memory used by the image.

Static Public Member Functions

- static bool **is_animated** (const char *name_)

Static Public Member Functions inherited from [FI_Image](#)

- static [FI_Labeltype](#) [define_FL_IMAGE_LABEL](#) ()
- static [FI_RGB_Scaling](#) [RGB_scaling](#) ()
Returns the currently used RGB image scaling method.
- static void [RGB_scaling](#) ([FI_RGB_Scaling](#))
Sets the RGB image scaling method used for copy(int, int).
- static [FI_RGB_Scaling](#) [scaling_algorithm](#) ()
Gets what algorithm is used when resizing a source image to draw it.
- static void [scaling_algorithm](#) ([FI_RGB_Scaling](#) algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Static Public Attributes

- static bool [animate](#) = false
Sets how the shared image core routine should treat animated GIF files.

Static Public Attributes inherited from [FI_Image](#)

- static const int [ERR_FILE_ACCESS](#) = -2
- static const int [ERR_FORMAT](#) = -3
- static const int [ERR_MEMORY_ACCESS](#) = -4
- static const int [ERR_NO_IMAGE](#) = -1
- static bool [register_images_done](#) = false
True after [fl_register_images\(\)](#) was called, false before.

Protected Member Functions

- [FI_GIF_Image](#) ()
The default constructor creates an empty GIF image.
- [FI_GIF_Image](#) (const char *filename, bool anim)
- [FI_GIF_Image](#) (const char *imagename, const unsigned char *data, const size_t length, bool anim)
- void [load](#) (const char *filename, bool anim)
The protected [load\(\)](#) methods are used by [FI_Anim_GIF_Image](#) to request loading of animated GIF's.
- void [load](#) (const char *imagename, const unsigned char *data, const size_t length, bool anim)
- void [load_gif_](#) (class [FI_Image_Reader](#) &rdr, bool anim=false)
- virtual void [on_extension_data](#) ([GIF_FRAME](#) &)
- virtual void [on_frame_data](#) ([GIF_FRAME](#) &)

Protected Member Functions inherited from [FI_Pixmap](#)

- void [measure](#) ()

Protected Member Functions inherited from [FI_Image](#)

- void [d](#) (int D)
Sets the current image depth.
- void [data](#) (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void [draw_empty](#) (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int [draw_scaled](#) (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void [h](#) (int H)
Sets the height of the image data.

- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Additional Inherited Members

Public Attributes inherited from [Fl_Pixmap](#)

- int **alloc_data**

Static Protected Member Functions inherited from [Fl_Image](#)

- static void **labeltype** (const [Fl_Label](#) *lo, int lx, int ly, int lw, int lh, [Fl_Align](#) la)
- static void **measure** (const [Fl_Label](#) *lo, int &lw, int &lh)

11.47.1 Detailed Description

The [Fl_GIF_Image](#) class supports loading, caching, and drawing of Compuserve GIFSM images. The class loads the first image and supports transparency.

11.47.2 Constructor & Destructor Documentation

11.47.2.1 [Fl_GIF_Image\(\)](#) [1/3]

```
Fl_GIF_Image::Fl_GIF_Image (
    const char * filename)
```

This constructor loads a GIF image from the given file.

If a GIF image is animated, [Fl_GIF_Image](#) will only read and display the first frame of the animation.

The destructor frees all memory and server resources that are used by the image.

Use [Fl_Image::fail\(\)](#) to check if [Fl_GIF_Image](#) failed to load. [fail\(\)](#) returns `ERR_FILE_ACCESS` if the file could not be opened or read, `ERR_FORMAT` if the GIF format could not be decoded, and `ERR_NO_IMAGE` if the image could not be loaded for another reason.

Parameters

<code>in</code>	<code>filename</code>	a full path and name pointing to a GIF image file.

See also

`Fl_GIF_Image::Fl_GIF_Image(const char *imagename, const unsigned char *data, const long length)`

11.47.2.2 [Fl_GIF_Image\(\)](#) [2/3]

```
Fl_GIF_Image::Fl_GIF_Image (
    const char * imagename,
    const unsigned char * data)
```

This constructor loads a GIF image from memory (deprecated).

Deprecated Please use `Fl_GIF_Image(const char *imagename, const unsigned char *data, const size_t length)` instead.

Note

Buffer overruns will not be checked.

This constructor should not be used because the caller can't supply the memory size and the image reader can't check for "end of memory" errors.

Note

A new constructor with parameter `length` is available since FLTK 1.4.0.

Parameters

in	<i>imagename</i>	A name given to this image or NULL
in	<i>data</i>	Pointer to the start of the GIF image in memory.

See also

[FI_GIF_Image\(const char *filename\)](#)

[FI_GIF_Image\(const char *imagename, const unsigned char *data, const size_t length\)](#)

11.47.2.3 FI_GIF_Image() [3/3]

```
FI_GIF_Image::FI_GIF_Image (
    const char * imagename,
    const unsigned char * data,
    const size_t length)
```

This constructor loads a GIF image from memory.

Construct an image from a block of memory inside the application. Fluid offers "binary data" chunks as a great way to add image data into the C++ source code. *imagename* can be NULL. If a name is given, the image is added to the list of shared images and will be available by that name.

If a GIF image is animated, [FI_GIF_Image](#) will only read and display the first frame of the animation.

The destructor frees all memory and server resources that are used by the image.

The third parameter *length* is used to test for buffer overruns, i.e. truncated images.

Use [FI_Image::fail\(\)](#) to check if [FI_GIF_Image](#) failed to load. [fail\(\)](#) returns ERR_FILE_ACCESS if the file could not be opened or read, ERR_FORMAT if the GIF format could not be decoded, and ERR_NO_IMAGE if the image could not be loaded for another reason.

Parameters

in	<i>imagename</i>	A name given to this image or NULL
in	<i>data</i>	Pointer to the start of the GIF image in memory.
in	<i>length</i>	Length of the GIF image in memory.

See also

[FI_GIF_Image::FI_GIF_Image\(const char *filename\)](#)

[FI_Shared_Image](#)

Since

1.4.0

11.47.3 Member Data Documentation**11.47.3.1 animate**

```
bool FI_GIF_Image::animate = false [static]
```

Sets how the shared image core routine should treat animated GIF files.

The default is to treat them as ordinary GIF's e.g. it creates a [FI_GIF_Image](#) object. If this variable is set, then an animated GIF object [FI_Anim_GIF_Image](#) is created.

The documentation for this class was generated from the following files:

- [FI_GIF_Image.H](#)
- [FI_Anim_GIF_Image.cxx](#)
- [FI_GIF_Image.cxx](#)

11.48 FL_Gl_Choice Class Reference

Public Member Functions

- **FL_Gl_Choice** (int m, const int *alistp, [FL_Gl_Choice](#) *n)

Friends

- class **FL_Gl_Window_Driver**

The documentation for this class was generated from the following file:

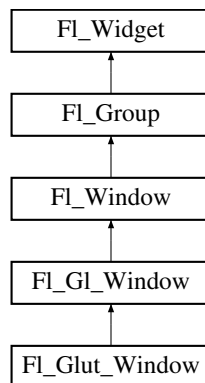
- [FL_Gl_Choice.H](#)

11.49 FL_Gl_Window Class Reference

The [FL_Gl_Window](#) widget sets things up so OpenGL works.

```
#include <Fl_Gl_Window.H>
```

Inheritance diagram for [FL_Gl_Window](#):



Public Member Functions

- [FL_Gl_Window](#) const * [as_gl_window](#) () const [FL_OVERRIDE](#)
- [FL_Gl_Window](#) * [as_gl_window](#) () [FL_OVERRIDE](#)
Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).
- int **can_do** ()
Returns non-zero if the hardware supports the current OpenGL mode.
- int [can_do_overlay](#) ()
Returns true if the hardware overlay is possible.
- [GLContext](#) [context](#) () const
Returns a pointer to the window's OpenGL rendering context.
- void [context](#) ([GLContext](#), int destroy_flag=0)
Sets a pointer to the [GLContext](#) that this window is using.
- char [context_valid](#) () const
Will only be set if the OpenGL context is created or recreated.
- void **context_valid** (char v)
See char [FL_Gl_Window::context_valid\(\)](#) const.
- [FL_Gl_Window](#) (int W, int H, const char *l=0)
Creates a new [FL_Gl_Window](#) widget using the given size, and label string.
- [FL_Gl_Window](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [FL_Gl_Window](#) widget using the given position, size, and label string.
- void **flush** () [FL_OVERRIDE](#)

- Forces the window to be drawn, this window is also made current and calls [draw\(\)](#).*

 - int [handle](#) (int) [FL_OVERRIDE](#)

Handle some FLTK events as needed.
 - void [hide](#) () [FL_OVERRIDE](#)

Hides the window and destroys the OpenGL context.
 - void [hide_overlay](#) ()

Hides the window if it is not this window, does nothing in Windows.
 - void [invalidate](#) ()

The [invalidate\(\)](#) method turns off [valid\(\)](#) and is equivalent to calling [value\(0\)](#).
 - void [make_current](#) ()

The [make_current\(\)](#) method selects the OpenGL context for the widget.
 - void [make_overlay_current](#) ()

Selects the OpenGL context for the widget's overlay.
 - [Fl_Mode mode](#) () const

Returns the current OpenGL capabilities of the window.
 - int [mode](#) (const int *a)

Set the OpenGL capabilities of the window using platform-specific data.
 - int [mode](#) (int a)

Set or change the OpenGL capabilities of the window.
 - void [ortho](#) ()

Sets the projection so 0,0 is in the lower left of the window and each pixel is 1 unit wide/tall.
 - int [pixel_h](#) ()

Gives the window height in OpenGL pixels.
 - int [pixel_w](#) ()

Gives the window width in OpenGL pixels.
 - float [pixels_per_unit](#) ()

The number of pixels per FLTK unit of length for the window.
 - void [redraw_overlay](#) ()

Causes [draw_overlay\(\)](#) to be called at a later time.
 - void [resize](#) (int, int, int, int) [FL_OVERRIDE](#)

Changes the size or position of the widget.
 - void [show](#) () [FL_OVERRIDE](#)

Makes a widget visible.
 - void [show](#) (int a, char **b)

*Same as [Fl_Window::show\(int a, char **b\)](#)*
 - void [swap_buffers](#) ()

The [swap_buffers\(\)](#) method swaps the back and front buffers.
 - int [swap_interval](#) () const

Gets the rate at which the GL windows swaps buffers.
 - void [swap_interval](#) (int)

Sets the rate at which the GL windows swaps buffers.
 - char [valid](#) () const

Is turned off when FLTK creates a new context for this window or when the window resizes, and is turned on after [draw\(\)](#) is called.
 - void [valid](#) (char v)

See char [Fl_Gl_Window::valid\(\)](#) const.
 - [~Fl_Gl_Window](#) ()

The destructor removes the widget and destroys the OpenGL context associated with it.

Public Member Functions inherited from FL_Window

- void [allow_expand_outside_parent](#) ()
Allow this subwindow to expand outside the area of its parent window.
- virtual class [FL_Double_Window](#) * [as_double_window](#) ()
Return non-null if this is an [FL_Double_Window](#) object.
- virtual class [FL_Overlay_Window](#) * [as_overlay_window](#) ()
Return non-null if this is an [FL_Overlay_Window](#) object.
- [FL_Window](#) const * [as_window](#) () const [FL_OVERRIDE](#)
- [FL_Window](#) * [as_window](#) () [FL_OVERRIDE](#)
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).
- unsigned int **border** () const
Returns whether the window possesses a border.
- void [border](#) (int b)
Sets whether or not the window manager border is around the window.
- void [clear_border](#) ()
Fast inline function to turn the window manager border off.
- void [clear_modal_states](#) ()
Clears the "modal" flags and converts a "modal" or "non-modal" window back into a "normal" window.
- void **copy_label** (const char *a)
Sets the window titlebar label to a copy of a character string.
- void [cursor](#) (const [FL_RGB_Image](#) *, int, int)
Changes the cursor for this window using the provided image as cursor's shape.
- void [cursor](#) ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void [cursor](#) ([FL_Cursor](#))
Changes the cursor for this window.
- int [decorated_h](#) () const
Returns the window height including any window title bar and any frame added by the window manager.
- int [decorated_w](#) () const
Returns the window width including any frame added by the window manager.
- void [default_cursor](#) ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void [default_cursor](#) ([FL_Cursor](#))
Sets the default window cursor.
- void **draw_backdrop** ()
Draw the background image if one is set and is aligned inside.
- [FL_Window](#) (int w, int h, const char *title=0)
Creates a window from the given width w, height h, and title.
- [FL_Window](#) (int x, int y, int w, int h, const char *title=0)
Creates a window from the given position (x, y), size (w, h) and title.
- void [free_position](#) ()
Undoes the effect of a previous [resize\(\)](#) or [show\(\)](#) so that the next time [show\(\)](#) is called the window manager is free to position the window.
- void [fullscreen](#) ()
Makes the window completely fill one or more screens, without any window manager border visible.
- unsigned int **fullscreen_active** () const
Returns non zero if [FULLSCREEN](#) flag is set, 0 otherwise.
- void **fullscreen_off** ()
Turns off any side effects of [fullscreen\(\)](#)
- void **fullscreen_off** (int X, int Y, int W, int H)

- Turns off any side effects of [fullscreen\(\)](#) and does [resize\(x,y,w,h\)](#).*

 - void [fullscreen_screens](#) (int top, int bottom, int left, int right)

Sets which screens should be used when this window is in fullscreen mode.
- [uchar get_size_range](#) (int *minw, int *minh, int *maxw=NULL, int *maxh=NULL, int *dw=NULL, int *dh=NULL, int *aspect=NULL)

Gets the allowable range to which the user can resize this window.
- int [handle](#) (int) [FL_OVERRIDE](#)

Handles the specified event.
- void [hide](#) () [FL_OVERRIDE](#)

Removes the window from the screen.
- void **hotspot** (const [FL_Widget](#) &p, int offscreen=0)

See void [FL_Window::hotspot](#)(int x, int y, int offscreen = 0)
- void **hotspot** (const [FL_Widget](#) *, int offscreen=0)

See void [FL_Window::hotspot](#)(int x, int y, int offscreen = 0)
- void [hotspot](#) (int x, int y, int offscreen=0)

Positions the window so that the mouse is pointing at the given position, or at the center of the given widget, which may be the window itself.
- const void * [icon](#) () const

Gets the current icon window target dependent data.
- void [icon](#) (const [FL_RGB_Image](#) *)

Sets or resets a single window icon.
- void [icon](#) (const void *ic)

Platform-specific method to set the window icon usable on Windows and X11 only.
- void [iconize](#) ()

Iconifies the window.
- const char * **iconlabel** () const

See void [FL_Window::iconlabel](#)(const char)*
- void **iconlabel** (const char *)

Sets the icon label.
- void [icons](#) (const [FL_RGB_Image](#) *[], int)

Sets the window icons.
- void [icons](#) (HICON big_icon, HICON small_icon)

Sets the window icons using HICON handles (Windows platform only).
- const char * **label** () const

See void [FL_Window::label](#)(const char)*
- void **label** (const char *)

Sets the window title bar label.
- void **label** (const char *label, const char *[iconlabel](#))

Sets the icon label.
- void [make_current](#) ()

Sets things up so that the drawing functions in [<FL/fl_draw.H>](#) will go into this window.
- void [maximize](#) ()

Maximizes a top-level window to its current screen.
- unsigned int **maximize_active** () const

Returns whether the window is currently maximized.
- unsigned int **menu_window** () const

Returns true if this window is a menu window.
- unsigned int [modal](#) () const

Returns true if this window is modal.

- unsigned int **non_modal** () const
Returns true if this window is modal or non-modal.
- [fl_uintptr_t os_id](#) ()
Returns a platform-specific identification of a shown window, or 0 if not shown.
- unsigned int **override** () const
Returns non zero if OVERRIDE flag is set, 0 otherwise.
- void [resize](#) (int X, int Y, int W, int H) [FL_OVERRIDE](#)
Changes the size and position of the window.
- int [screen_num](#) ()
The number of the screen containing the mapped window.
- void [screen_num](#) (int screen_num)
Set the number of the screen where to map the window.
- void [set_menu_window](#) ()
Marks the window as a menu window.
- void [set_modal](#) ()
A "modal" window, when [shown\(\)](#), will prevent any events from being delivered to other windows in the same program, and will also remain on top of the other windows (if the X window manager supports the "transient for" property).
- void [set_non_modal](#) ()
A "non-modal" window (terminology borrowed from Microsoft Windows) acts like a [modal\(\)](#) one in that it remains on top, but it has no effect on event delivery.
- void **set_override** ()
Activates the flags NOBORDER|OVERRIDE.
- void [set_tooltip_window](#) ()
Marks the window as a tooltip window.
- const [Fl_Image](#) * [shape](#) ()
Returns the image controlling the window shape or NULL.
- void [shape](#) (const [Fl_Image](#) &b)
Set the window's shape with an [Fl_Image](#).
- void [shape](#) (const [Fl_Image](#) *img)
Assigns a non-rectangular shape to the window.
- void [show](#) () [FL_OVERRIDE](#)
Puts the window on the screen.
- void [show](#) (int argc, char **argv)
Puts the window on the screen with [show\(\)](#) and parses command-line arguments.
- int [shown](#) ()
Returns non-zero if [show\(\)](#) has been called (but not [hide\(\)](#)).
- void [size_range](#) (int minw, int minh, int maxw=0, int maxh=0, int dw=0, int dh=0, int aspect=0)
Sets the allowable range to which the user can resize this window.
- unsigned int **tooltip_window** () const
Returns true if this window is a tooltip window.
- void [un_maximize](#) ()
Returns a previously maximized top-level window to its previous size.
- void [wait_for_expose](#) ()
Waits for the window to be displayed after calling [show\(\)](#).
- int **x_root** () const
Gets the x position of the window on the screen.
- const char * [xclass](#) () const
Returns the xclass for this window, or a default.
- void [xclass](#) (const char *c)
Sets the xclass for this window.
- int **y_root** () const
Gets the y position of the window on the screen.
- virtual [~Fl_Window](#) ()
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *[_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void [add](#) ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void [add](#) ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void [add_resizable](#) ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * [array](#) () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * [as_group](#) () const [FL_OVERRIDE](#)
- [FL_Group](#) * [as_group](#) () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void [begin](#) ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * [child](#) (int n) const
Returns the n'th child.
- int [children](#) () const
Returns how many child widgets the group has.
- void [clear](#) ()
Deletes all child widgets from memory recursively.
- unsigned int [clip_children](#) ()
Returns the current clipping mode.
- void [clip_children](#) (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int [delete_child](#) (int n)
*Removes the widget at *index* from the group and deletes it.*
- void [end](#) ()
*Exactly the same as *current(this->parent())*.*
- int [find](#) (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int [find](#) (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char *s=0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void [focus](#) ([FL_Widget](#) *W)
- void [forms_end](#) ()
This is for forms compatibility only.
- void [init_sizes](#) ()
Resets the internal array of widget sizes and positions.
- void [insert](#) ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void [insert](#) ([FL_Widget](#) &o, [FL_Widget](#) *before)
*This does *insert(w, find(before))*.*
- void [remove](#) ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void [remove](#) ([FL_Widget](#) *o)
Removes the widget o from the group.
- void [remove](#) (int index)

- Removes the widget at `index` from the group but does not delete it.*
- `FI_Widget * resizable () const`
 - Returns the group's resizable widget.*
- `void resizable (FI_Widget &o)`
 - Sets the group's resizable widget.*
- `void resizable (FI_Widget *o)`
 - The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
- `virtual ~FI_Group ()`
 - The destructor also deletes all the children.*

Public Member Functions inherited from FI_Widget

- `void _clear_fullscreen ()`
- `void _set_fullscreen ()`
- `void activate ()`
 - Activates the widget.*
- `unsigned int active () const`
 - Returns whether the widget is active.*
- `int active_r () const`
 - Returns whether the widget and all of its parents are active.*
- `FI_Align align () const`
 - Gets the label alignment.*
- `void align (FI_Align alignment)`
 - Sets the label alignment.*
- `long argument () const`
 - Gets the current user data (long) argument that is passed to the callback function.*
- `void argument (long v)`
 - Sets the current user data (long) argument that is passed to the callback function.*
- `void bind_deimage (FI_Image *img)`
 - Sets the image to use as part of the widget label when in the inactive state.*
- `void bind_deimage (int f)`
 - Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- `void bind_image (FI_Image *img)`
 - Sets the image to use as part of the widget label when in the active state.*
- `void bind_image (int f)`
 - Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- `FI_Boxtype box () const`
 - Gets the box type of the widget.*
- `void box (FI_Boxtype new_box)`
 - Sets the box type for the widget.*
- `FI_Callback_p callback () const`
 - Gets the current callback function for the widget.*
- `void callback (FI_Callback *cb)`
 - Sets the current callback function for the widget.*
- `void callback (FI_Callback *cb, FI_Callback_User_Data *p, bool auto_free)`
 - Sets the current callback function and managed user data for the widget.*
- `void callback (FI_Callback *cb, void *p)`
 - Sets the current callback function and data for the widget.*
- `void callback (FI_Callback0 *cb)`
 - Sets the current callback function for the widget.*
- `void callback (FI_Callback1 *cb, long p=0)`

- Sets the current callback function for the widget.*

 - unsigned int `changed` () const

Checks if the widget value changed since the last callback.
- void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FL_Color` bg)

Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)

Sets the background and selection color of the widget.
- `FL_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.

- void `do_callback` (`FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)

Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [Fl_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([Fl_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- [Fl_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

Gets the current tooltip text.
- void [tooltip](#) (const char *text)

Sets the current tooltip text.

- `FL_Window * top_window ()` const
Returns a pointer to the top-level window for the widget.
- `FL_Window * top_window_offset (int &xoff, int &yoff)` const
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type ()` const
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data ()` const
Gets the user data for this widget.
- `void user_data (FL_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible ()` const
Returns whether a widget is visible.
- `unsigned int visible_focus ()` const
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r ()` const
Returns whether a widget and all its parents are visible.
- `int w ()` const
Gets the widget width.
- `FL_When when ()` const
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `FL_Window * window ()` const
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x ()` const
Gets the widget position in its window.
- `int y ()` const
Gets the widget position in its window.
- `virtual ~FL_Widget ()`
Destroys the widget.

Static Public Member Functions

- `static int can_do (const int *m)`
Returns non-zero if the hardware supports the given OpenGL mode.
- `static int can_do (int m)`
Returns non-zero if the hardware supports the given OpenGL mode.

Static Public Member Functions inherited from [Fl_Window](#)

- static [Fl_Window](#) * [current](#) ()
Returns the last window that was made current.
- static void [default_callback](#) ([Fl_Window](#) *, void *v)
Back compatibility: Sets the default callback v for win to call on close event.
- static void [default_icon](#) (const [Fl_RGB_Image](#) *)
Sets a single default window icon.
- static void [default_icons](#) (const [Fl_RGB_Image](#) *[], int)
Sets the default window icons.
- static void [default_icons](#) (HICON big_icon, HICON small_icon)
Sets the default window icons (Windows platform only).
- static const char * [default_xclass](#) ()
Returns the default xclass.
- static void [default_xclass](#) (const char *)
Sets the default window xclass.
- static bool [is_a_rescale](#) ()
Returns true when a window is being rescaled.
- static char [show_next_window_iconic](#) ()
Returns the static flag whether the next window should be opened iconified.
- static void [show_next_window_iconic](#) (char stat)
Sets a static flag whether the next window should be opened iconified.

Static Public Member Functions inherited from [Fl_Group](#)

- static [Fl_Group](#) * [current](#) ()
Returns the currently active group.
- static void [current](#) ([Fl_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Member Functions

- void [draw](#) () [FL_OVERRIDE](#)
Draws the [Fl_Gl_Window](#).
- void [draw_begin](#) ()
Supports drawing to an [Fl_Gl_Window](#) with the FLTK 2D drawing API.
- void [draw_end](#) ()
To be used as a match for a previous call to [Fl_Gl_Window::draw_begin\(\)](#).

Protected Member Functions inherited from FI_Window

- void `default_size_range` ()
Protected method to calculate the default size range of a window.
- void `draw` () `FL_OVERRIDE`
Draws the widget.
- int `force_position` () const
Returns the internal state of the window's FORCE_POSITION flag.
- void `force_position` (int force)
Sets an internal flag that tells FLTK and the window manager to honor position requests.
- void `free_icons` ()
Deletes all icons previously attached to the window.
- int `is_resizable` ()
Protected method to determine whether a window is resizable.

Protected Member Functions inherited from FI_Group

- `FI_Rect` * `bounds` ()
Returns the internal array of widget sizes and positions.
- void `draw_child` (`FI_Widget` &widget) const
Forces a child to redraw.
- void `draw_children` ()
Draws all children of the group.
- void `draw_outside_label` (const `FI_Widget` &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int `on_insert` (`FI_Widget` *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int `on_move` (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void `on_remove` (int)
Allow derived groups to act when a child widget is removed from the group.
- int * `sizes` ()
Returns the internal array of widget sizes and positions.
- void `update_child` (`FI_Widget` &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from FI_Widget

- void `clear_flag` (unsigned int c)
Clears a flag in the flags mask.
- void `draw_backdrop` () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void `draw_box` () const
Draws the widget box according its box style.
- void `draw_box` (`FI_Boxtype` t, `FI_Color` c) const
Draws a box of type t, of color c at the widget's position and size.
- void `draw_box` (`FI_Boxtype` t, int x, int y, int w, int h, `FI_Color` c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void `draw_focus` () const
Draws a focus rectangle around the widget.
- void `draw_focus` (`FI_Boxtype` t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.

- void `draw_focus` (`FI_Boxtype` t, int x, int y, int w, int h, `FI_Color` bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `FI_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Friends

- class `FI_GI_Window_Driver`

Additional Inherited Members

Public Types inherited from `FI_Window`

- typedef struct HICON__ * `HICON`

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Static Protected Attributes inherited from `FI_Window`

- static `FI_Window` * `current_`
Stores the last window that was made current.

11.49.1 Detailed Description

The [FL_Gl_Window](#) widget sets things up so OpenGL works.

It also keeps an OpenGL "context" for that window, so that changes to the lighting and projection may be reused between redraws. [FL_Gl_Window](#) also flushes the OpenGL streams and swaps buffers after [draw\(\)](#) returns.

OpenGL hardware typically provides some overlay bit planes, which are very useful for drawing UI controls atop your 3D graphics. If the overlay hardware is not provided, FLTK tries to simulate the overlay. This works pretty well if your graphics are double buffered, but not very well for single-buffered.

Please note that the FLTK drawing and clipping functions will not work inside an [FL_Gl_Window](#). All drawing should be done using OpenGL calls exclusively.

See also

[OpenGL and support of HighDPI displays](#)

Note

FLTK 1.4 introduces a driver system for graphic calls. It is now possible to add a selection of widgets to an OpenGL window. The widgets will draw on top of any OpenGL rendering. The number of supported widgets will increase as the driver development improves. Program `test/cube.cxx` illustrates how to do that.

FLTK expects that when an [FL_Gl_Window](#) is a child of a parent [FL_Window](#), the child window lies entirely inside its parent window. If that's not the case, what happens to the part of the GL subwindow which leaks outside its parent is undefined and susceptible to be platform-specific.

11.49.2 Constructor & Destructor Documentation

11.49.2.1 FL_Gl_Window() [1/2]

```
Fl_Gl_Window::Fl_Gl_Window (
    int W,
    int H,
    const char * l = 0) [inline]
```

Creates a new [FL_Gl_Window](#) widget using the given size, and label string.

The default boxtype is `FL_NO_BOX`. The default mode is `FL_RGB|FL_DOUBLE|FL_DEPTH`.

11.49.2.2 FL_Gl_Window() [2/2]

```
Fl_Gl_Window::Fl_Gl_Window (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0) [inline]
```

Creates a new [FL_Gl_Window](#) widget using the given position, size, and label string.

The default boxtype is `FL_NO_BOX`. The default mode is `FL_RGB|FL_DOUBLE|FL_DEPTH`.

11.49.3 Member Function Documentation

11.49.3.1 as_gl_window() [1/2]

```
Fl_Gl_Window const * Fl_Gl_Window::as_gl_window () const [inline], [virtual]
```

Reimplemented from [FL_Widget](#).

11.49.3.2 as_gl_window() [2/2]

```
Fl_Gl_Window * Fl_Gl_Window::as_gl_window () [inline], [virtual]
```

Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).

Use this method if you have a widget (pointer) and need to know whether this widget is derived from [FL_Gl_Window](#).

If it returns non-NULL, then the widget in question is derived from [FL_Gl_Window](#).

Return values

<code>NULL</code>	if this widget is not derived from Fl_Gl_Window .

Note

This method is provided to avoid `dynamic_cast`.

See also

[Fl_Widget::as_group\(\)](#), [Fl_Widget::as_window\(\)](#)

Reimplemented from [Fl_Widget](#).

11.49.3.3 can_do()

```
static int Fl_Gl_Window::can_do (
    const int * m) [inline], [static]
```

Returns non-zero if the hardware supports the given OpenGL mode.

See also

[Fl_Gl_Window::mode\(const int *a\)](#)

11.49.3.4 can_do_overlay()

```
int Fl_Gl_Window::can_do_overlay ()
```

Returns true if the hardware overlay is possible.

If this is false, FLTK will try to simulate the overlay, with significant loss of update speed. Calling this will cause FLTK to open the display.

11.49.3.5 context() [1/2]

```
GLContext Fl_Gl_Window::context () const [inline]
```

Returns a pointer to the window's OpenGL rendering context.

See also

`void` [context\(GLContext c, int destroy_flag\)](#)

11.49.3.6 context() [2/2]

```
void Fl_Gl_Window::context (
    GLContext v,
    int destroy_flag = 0)
```

Sets a pointer to the GLContext that this window is using.

This is a system-dependent structure, but it is portable to copy the context from one window to another. You can also set it to `NULL`, which will force FLTK to recreate the context the next time [make_current\(\)](#) is called, this is useful for getting around bugs in OpenGL implementations.

If *destroy_flag* is true the context will be destroyed by fltk when the window is destroyed, or when the `mode()` is changed, or the next time `context(x)` is called.

11.49.3.7 context_valid()

```
char Fl_Gl_Window::context_valid () const [inline]
```

Will only be set if the OpenGL context is created or recreated.

It differs from [Fl_Gl_Window::valid\(\)](#) which is also set whenever the context changes size.

11.49.3.8 draw()

```
void Fl_Gl_Window::draw (
    void ) [protected], [virtual]
```

Draws the [Fl_Gl_Window](#).

You **must** subclass [Fl_Gl_Window](#) and provide an implementation for [draw\(\)](#). You may also provide an implementation of [draw_overlay\(\)](#) if you want to draw into the overlay planes. You can avoid reinitializing the viewport and lights and other things by checking [valid\(\)](#) at the start of [draw\(\)](#) and only doing the initialization if it is false.

The [draw\(\)](#) method can *only* use OpenGL calls. Do not attempt to call X, any of the functions in [<FL/fl_draw.H>](#), or glX directly. Do not call [gl_start\(\)](#) or [gl_finish\(\)](#).

If double-buffering is enabled in the window, the back and front buffers are swapped after this function is completed.

The following pseudo-code shows how to use "if (!valid())" to initialize the viewport:

```
void mywindow::draw() {
    if (!valid()) {
        glViewport(0,0,pixel_w(),pixel_h());
        glFrustum(...) or glOrtho(...)
        ...other initialization...
    }
    if (!context_valid()) {
        ...load textures, etc. ...
    }
    // clear screen
    glClearColor(...);
    glClear(...);
    ... draw your geometry here ...
}
```

Actual example code to clear screen to black and draw a 2D white "X":

```
void mywindow::draw() {
    if (!valid()) {
        glLoadIdentity();
        glViewport(0,0,pixel_w(),pixel_h());
        glOrtho(-w(),w(),-h(),h(),-1,1);
    }
    // Clear screen
    glClear(GL_COLOR_BUFFER_BIT);
    // Draw white 'X'
    glColor3f(1.0, 1.0, 1.0);
    glBegin(GL_LINE_STRIP); glVertex2f(w(), h()); glVertex2f(-w(),-h()); glEnd();
    glBegin(GL_LINE_STRIP); glVertex2f(w(),-h()); glVertex2f(-w(), h()); glEnd();
}
```

Regular FLTK widgets can be added as children to the [Fl_Gl_Window](#). To correctly overlay the widgets, [Fl_Gl_Window::draw\(\)](#) must be called after rendering the main scene.

```
void mywindow::draw() {
    // draw 3d graphics scene
    Fl_Gl_Window::draw();
    // -- or --
    draw_begin();
    Fl_Window::draw();
    // other 2d drawing calls, overlays, etc.
    draw_end();
}
```

Implements [Fl_Widget](#).

Reimplemented in [Fl_Glut_Window](#).

11.49.3.9 draw_begin()

```
void Fl_Gl_Window::draw_begin () [protected]
```

Supports drawing to an [Fl_Gl_Window](#) with the FLTK 2D drawing API.

See also

[Using FLTK widgets in OpenGL Windows](#)

11.49.3.10 draw_end()

```
void Fl_Gl_Window::draw_end () [protected]
```

To be used as a match for a previous call to [Fl_Gl_Window::draw_begin\(\)](#).

See also

[Using FLTK widgets in OpenGL Windows](#)

11.49.3.11 flush()

```
void Fl_Gl_Window::flush () [virtual]
```

Forces the window to be drawn, this window is also made current and calls [draw\(\)](#).

Reimplemented from [Fl_Window](#).

11.49.3.12 handle()

```
int Fl_Gl_Window::handle (
    int event) [virtual]
```

Handle some FLTK events as needed.

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Glut_Window](#).

11.49.3.13 hide()

```
void Fl_Gl_Window::hide () [virtual]
```

Hides the window and destroys the OpenGL context.

Reimplemented from [Fl_Widget](#).

11.49.3.14 make_current()

```
void Fl_Gl_Window::make_current ()
```

The [make_current\(\)](#) method selects the OpenGL context for the widget.

It is called automatically prior to the [draw\(\)](#) method being called and can also be used to implement feedback and/or selection within the [handle\(\)](#) method.

11.49.3.15 make_overlay_current()

```
void Fl_Gl_Window::make_overlay_current ()
```

Selects the OpenGL context for the widget's overlay.

This method is called automatically prior to the [draw_overlay\(\)](#) method being called and can also be used to implement feedback and/or selection within the [handle\(\)](#) method.

11.49.3.16 mode() [1/3]

```
Fl_Mode Fl_Gl_Window::mode () const [inline]
```

Returns the current OpenGL capabilities of the window.

Don't use this if capabilities were set through [Fl_Gl_Window::mode\(const int *a\)](#).

11.49.3.17 mode() [2/3]

```
int Fl_Gl_Window::mode (
    const int * a) [inline]
```

Set the OpenGL capabilities of the window using platform-specific data.

Parameters

<i>a</i>	zero-ending array of platform-specific attributes and attribute values

Unix/Linux platform: attributes are GLX attributes adequate for the 3rd argument of the `glXChooseVisual()` function (e.g., `GLX_DOUBLEBUFFER`, defined by including `<GL/glx.h>`).

Note

What attributes are adequate here is subject to change. The preferred, stable public API is [Fl_Gl_Window::mode\(int a\)](#).

Windows platform: this member function is of no use.

Mac OS X platform: attributes belong to the `kCGLPixelFormatAttribute` enumeration (defined by including `<OpenGL/OpenGL.h>`, e.g., `kCGLPFADoubleBuffer`) and may be followed by adequate attribute values.

11.49.3.18 mode() [3/3]

```
int Fl_Gl_Window::mode (
    int a) [inline]
```

Set or change the OpenGL capabilities of the window.

The value can be any of the following OR'd together:

- `FL_RGB` - RGB color (not indexed)
- `FL_RGB8` - RGB color with at least 8 bits of each color
- `FL_INDEX` - Indexed mode
- `FL_SINGLE` - not double buffered
- `FL_DOUBLE` - double buffered
- `FL_ACCUM` - accumulation buffer
- `FL_ALPHA` - alpha channel in color
- `FL_DEPTH` - depth buffer
- `FL_STENCIL` - stencil buffer
- `FL_MULTISAMPLE` - multisample antialiasing
- `FL_OPENGL3` - use OpenGL version 3.0 or more.

`FL_RGB` and `FL_SINGLE` have a value of zero, so they are "on" unless you give `FL_INDEX` or `FL_DOUBLE`.

If the desired combination cannot be done, FLTK will try turning off `FL_MULTISAMPLE`. If this also fails the `show()` will call `Fl::error()` and not show the window.

You can change the mode while the window is displayed. This is most useful for turning double-buffering on and off. Under X this will cause the old X window to be destroyed and a new one to be created. If this is a top-level window this will unfortunately also cause the window to blink, raise to the top, and be de-iconized, and the `xid()` will change, possibly breaking other code. It is best to make the GL window a child of another window if you wish to do this!

`mode()` must not be called within `draw()` since it changes the current context.

The `FL_OPENGL3` flag is recommended to use OpenGL version 3 or more. This flag is required (not just recommended) if GL 3.0 is in use and at least one of these conditions applies:

- the program runs on the macOS platform;
- the `Fl_Gl_Window` has child widgets.

See more details in [Using OpenGL 3.0 \(or higher versions\)](#).

Version

the `FL_OPENGL3` flag appeared in version 1.3.4

11.49.3.19 ortho()

```
void Fl_Gl_Window::ortho ()
```

Sets the projection so 0,0 is in the lower left of the window and each pixel is 1 unit wide/tall.

If you are drawing 2D images, your `draw()` method may want to call this if `valid()` is false.

11.49.3.20 pixel_h()

```
int Fl_Gl_Window::pixel_h () [inline]
```

Gives the window height in OpenGL pixels.

When an `Fl_Gl_Window` is mapped to a HighDPI display, the value given by `Fl_Gl_Window::h()` which is expressed in FLTK units, may differ from the window height in pixels. Calls to OpenGL functions expecting pixel values (e.g., `glViewport`) should therefore use `pixel_h()` rather than `h()`. Method `pixel_h()` detects when the GUI is rescaled or when the window has been moved between low and high resolution displays and automatically adjusts the returned value.

Version

1.3.4

11.49.3.21 pixel_w()

```
int Fl_Gl_Window::pixel_w () [inline]
```

Gives the window width in OpenGL pixels.

When an [Fl_Gl_Window](#) is mapped to a HighDPI display, the value given by [Fl_Gl_Window::w\(\)](#) which is expressed in FLTK units, may differ from the window width in pixels. Calls to OpenGL functions expecting pixel values (e.g., `glViewport`) should therefore use [pixel_w\(\)](#) rather than [w\(\)](#). Method [pixel_w\(\)](#) detects when the GUI is rescaled or when the window has been moved between low and high resolution displays and automatically adjusts the returned value.

Version

1.3.4

11.49.3.22 pixels_per_unit()

```
float Fl_Gl_Window::pixels_per_unit ()
```

The number of pixels per FLTK unit of length for the window.

This method dynamically adjusts its value when the GUI is rescaled or when the window is moved to/from displays of distinct resolutions. This method is useful, e.g., to convert, in a window's [handle\(\)](#) method, the FLTK units returned by [Fl::event_x\(\)](#) and [Fl::event_y\(\)](#) to the pixel units used by the OpenGL source code.

Version

1.3.4

11.49.3.23 redraw_overlay()

```
void Fl_Gl_Window::redraw_overlay ()
```

Causes [draw_overlay\(\)](#) to be called at a later time.

Initially the overlay is clear. If you want the window to display something in the overlay when it first appears, you must call this immediately after you [show\(\)](#) your window.

11.49.3.24 resize()

```
void Fl_Gl_Window::resize (
    int x,
    int y,
    int w,
    int h) [virtual]
```

Changes the size or position of the widget.

This is a virtual function so that the widget may implement its own handling of resizing. The default version does *not* call the [redraw\(\)](#) method, but instead relies on the parent widget to do so because the parent may know a faster way to update the display, such as scrolling from the old position.

Some window managers under X11 call [resize\(\)](#) a lot more often than needed. Please verify that the position or size of a widget did actually change before doing any extensive calculations.

`position(X, Y)` is a shortcut for `resize(X, Y, w\(\), h\(\))`, and `size(W, H)` is a shortcut for `resize(x\(\), y\(\), W, H)`.

Parameters

in	<i>x,y</i>	new position relative to the parent window
in	<i>w,h</i>	new size

See also

[position\(int,int\)](#), [size\(int,int\)](#)

Reimplemented from [Fl_Widget](#).

11.49.3.25 show()

```
void Fl_Gl_Window::show () [virtual]
```

Makes a widget visible.

An invisible widget never gets redrawn and does not get keyboard or mouse events, but can receive a few other events like FL_SHOW.

The [visible\(\)](#) method returns true if the widget is set to be visible. The [visible_r\(\)](#) method returns true if the widget and all of its parents are visible. A widget is only visible if [visible\(\)](#) is true on it *and all of its parents*.

Changing it will send FL_SHOW or FL_HIDE events to the widget. *Do not change it if the parent is not visible, as this will send false FL_SHOW or FL_HIDE events to the widget.* [redraw\(\)](#) is called if necessary on this or the parent.

See also

[hide\(\)](#), [visible\(\)](#), [visible_r\(\)](#)

Reimplemented from [Fl_Widget](#).

11.49.3.26 swap_buffers()

```
void Fl_Gl_Window::swap_buffers ()
```

The [swap_buffers\(\)](#) method swaps the back and front buffers.

It is called automatically after the [draw\(\)](#) method is called.

11.49.3.27 swap_interval() [1/2]

```
int Fl_Gl_Window::swap_interval () const
```

Gets the rate at which the GL windows swaps buffers.

This method can be called after the OpenGL context was created, typically within the user overridden [Fl_Gl_Window::draw\(\)](#) method that will be overridden by the user.

Note

This method depends highly on the underlying OpenGL contexts and driver implementation. Some drivers return no information, most drivers don't support intervals with multiple frames and return only 0 or 1.

Some drivers have the ability to set the swap interval but no way to query it, hence this method may return -1 even though the interval was set correctly. Conversely a return value greater zero does not guarantee that the driver actually honors the setting.

Returns

an integer greater zero if vertical blanking is taken into account when swapping OpenGL buffers

0 if the vertical blanking is ignored

-1 if the information can not be retrieved

11.49.3.28 swap_interval() [2/2]

```
void Fl_Gl_Window::swap_interval (
    int frames)
```

Sets the rate at which the GL windows swaps buffers.

This method can be called after the OpenGL context was created, typically within the user overridden [Fl_Gl_Window::draw\(\)](#) method that will be overridden by the user.

Note

This method depends highly on the underlying OpenGL contexts and driver implementation. Most driver seem to accept only 0 and 1 to swap buffer asynchronously or in sync with the vertical blank.

Parameters

<i>in</i>	<i>frames</i>	set the number of vertical frame blanks between OpenGL buffer swaps

11.49.3.29 valid()

```
char Fl_Gl_Window::valid () const [inline]
```

Is turned off when FLTK creates a new context for this window or when the window resizes, and is turned on *after* [draw\(\)](#) is called.

You can use this inside your [draw\(\)](#) method to avoid unnecessarily initializing the OpenGL context. Just do this:

```
void mywindow::draw() {
    if (!valid()) {
        glViewport(0,0,pixel_w(),pixel_h());
        glFrustum(...);
        ...other initialization...
    }
    if (!context_valid()) {
        ...load textures, etc. ...
    }
    ... draw your geometry here ...
}
```

You can turn [valid\(\)](#) on by calling [valid\(1\)](#). You should only do this after fixing the transformation inside a [draw\(\)](#) or after [make_current\(\)](#). This is done automatically after [draw\(\)](#) returns.

The documentation for this class was generated from the following files:

- [Fl_Gl_Window.H](#)
- [Fl_Gl_Overlay.cxx](#)
- [Fl_Gl_Window.cxx](#)

11.50 Fl_Glut_Bitmap_Font Struct Reference

fltk glut font/size attributes used in the glutXXX functions

```
#include <glut.H>
```

Public Attributes

- [Fl_Font](#) font
- [Fl_Fontsize](#) size

11.50.1 Detailed Description

fltk glut font/size attributes used in the glutXXX functions

The documentation for this struct was generated from the following file:

- [glut.H](#)

11.51 Fl_Glut_StrokeChar Struct Reference

Public Attributes

- int **Number**
- GLfloat **Right**
- const [Fl_Glut_StrokeStrip](#) * **Strips**

The documentation for this struct was generated from the following file:

- [glut.H](#)

11.52 Fl_Glut_StrokeFont Struct Reference

Public Attributes

- const [Fl_Glut_StrokeChar](#) ** **Characters**
- GLfloat **Height**
- char * **Name**
- int **Quantity**

The documentation for this struct was generated from the following file:

- glut.H

11.53 FI_Glut_StrokeStrip Struct Reference

Public Attributes

- int **Number**
- const [FI_Glut_StrokeVertex](#) * **Vertices**

The documentation for this struct was generated from the following file:

- glut.H

11.54 FI_Glut_StrokeVertex Struct Reference

Public Attributes

- GLfloat **X**
- GLfloat **Y**

The documentation for this struct was generated from the following file:

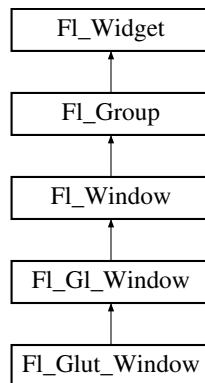
- glut.H

11.55 FI_Glut_Window Class Reference

GLUT is emulated using this window class and these static variables (plus several more static variables hidden in glut_compatibility.cxx):

```
#include <glut.H>
```

Inheritance diagram for FI_Glut_Window:



Public Member Functions

- **FI_Glut_Window** (int w, int h, const char *t=0)
Creates a glut window, registers to the glut windows list.
- **FI_Glut_Window** (int x, int y, int w, int h, const char *t=0)
Creates a glut window, registers to the glut windows list.
- void **make_current** ()
- **~FI_Glut_Window** ()
Destroys the glut window, first unregister it from the glut windows list.

Public Member Functions inherited from `FL_Gl_Window`

- `FL_Gl_Window` const * `as_gl_window` () const `FL_OVERRIDE`
- `FL_Gl_Window` * `as_gl_window` () `FL_OVERRIDE`
Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.
- int `can_do` ()
Returns non-zero if the hardware supports the current OpenGL mode.
- int `can_do_overlay` ()
Returns true if the hardware overlay is possible.
- `GLContext` `context` () const
Returns a pointer to the window's OpenGL rendering context.
- void `context` (`GLContext`, int `destroy_flag`=0)
Sets a pointer to the `GLContext` that this window is using.
- char `context_valid` () const
Will only be set if the OpenGL context is created or recreated.
- void `context_valid` (char v)
See char `FL_Gl_Window::context_valid()` const.
- `FL_Gl_Window` (int W, int H, const char *l=0)
Creates a new `FL_Gl_Window` widget using the given size, and label string.
- `FL_Gl_Window` (int X, int Y, int W, int H, const char *l=0)
Creates a new `FL_Gl_Window` widget using the given position, size, and label string.
- void `flush` () `FL_OVERRIDE`
Forces the window to be drawn, this window is also made current and calls `draw()`.
- void `hide` () `FL_OVERRIDE`
Hides the window and destroys the OpenGL context.
- void `hide_overlay` ()
Hides the window if it is not this window, does nothing in Windows.
- void `invalidate` ()
The `invalidate()` method turns off `valid()` and is equivalent to calling `value(0)`.
- void `make_current` ()
The `make_current()` method selects the OpenGL context for the widget.
- void `make_overlay_current` ()
Selects the OpenGL context for the widget's overlay.
- `FL_Mode` `mode` () const
Returns the current OpenGL capabilities of the window.
- int `mode` (const int *a)
Set the OpenGL capabilities of the window using platform-specific data.
- int `mode` (int a)
Set or change the OpenGL capabilities of the window.
- void `ortho` ()
Sets the projection so 0,0 is in the lower left of the window and each pixel is 1 unit wide/tall.
- int `pixel_h` ()
Gives the window height in OpenGL pixels.
- int `pixel_w` ()
Gives the window width in OpenGL pixels.
- float `pixels_per_unit` ()
The number of pixels per FLTK unit of length for the window.
- void `redraw_overlay` ()
Causes `draw_overlay()` to be called at a later time.
- void `resize` (int, int, int, int) `FL_OVERRIDE`
Changes the size or position of the widget.

- void `show` () `FL_OVERRIDE`
Makes a widget visible.
- void `show` (int a, char **b)
Same as `Fl_Window::show(int a, char **b)`
- void `swap_buffers` ()
The `swap_buffers()` method swaps the back and front buffers.
- int `swap_interval` () const
Gets the rate at which the GL windows swaps buffers.
- void `swap_interval` (int)
Sets the rate at which the GL windows swaps buffers.
- char `valid` () const
Is turned off when FLTK creates a new context for this window or when the window resizes, and is turned on after `draw()` is called.
- void `valid` (char v)
See char `Fl_Gl_Window::valid()` const.
- `~Fl_Gl_Window` ()
The destructor removes the widget and destroys the OpenGL context associated with it.

Public Member Functions inherited from `Fl_Window`

- void `allow_expand_outside_parent` ()
Allow this subwindow to expand outside the area of its parent window.
- virtual class `Fl_Double_Window * as_double_window` ()
Return non-null if this is an `Fl_Double_Window` object.
- virtual class `Fl_Overlay_Window * as_overlay_window` ()
Return non-null if this is an `Fl_Overlay_Window` object.
- `Fl_Window` const * `as_window` () const `FL_OVERRIDE`
- `Fl_Window * as_window` () `FL_OVERRIDE`
Returns an `Fl_Window` pointer if this widget is an `Fl_Window`.
- unsigned int `border` () const
Returns whether the window possesses a border.
- void `border` (int b)
Sets whether or not the window manager border is around the window.
- void `clear_border` ()
Fast inline function to turn the window manager border off.
- void `clear_modal_states` ()
Clears the "modal" flags and converts a "modal" or "non-modal" window back into a "normal" window.
- void `copy_label` (const char *a)
Sets the window titlebar label to a copy of a character string.
- void `cursor` (const `Fl_RGB_Image` *, int, int)
Changes the cursor for this window using the provided image as cursor's shape.
- void `cursor` (`Fl_Cursor` c, `Fl_Color`, `Fl_Color`=`FL_WHITE`)
For back compatibility only.
- void `cursor` (`Fl_Cursor`)
Changes the cursor for this window.
- int `decorated_h` () const
Returns the window height including any window title bar and any frame added by the window manager.
- int `decorated_w` () const
Returns the window width including any frame added by the window manager.
- void `default_cursor` (`Fl_Cursor` c, `Fl_Color`, `Fl_Color`=`FL_WHITE`)
For back compatibility only.

- void **default_cursor** ([FL_Cursor](#))
Sets the default window cursor.
- void **draw_backdrop** ()
Draw the background image if one is set and is aligned inside.
- [FL_Window](#) (int *w*, int *h*, const char **title*=0)
*Creates a window from the given width *w*, height *h*, and *title*.*
- [FL_Window](#) (int *x*, int *y*, int *w*, int *h*, const char **title*=0)
*Creates a window from the given position (*x*, *y*), size (*w*, *h*) and *title*.*
- void **free_position** ()
Undoes the effect of a previous [resize\(\)](#) or [show\(\)](#) so that the next time [show\(\)](#) is called the window manager is free to position the window.
- void **fullscreen** ()
Makes the window completely fill one or more screens, without any window manager border visible.
- unsigned int **fullscreen_active** () const
Returns non zero if FULLSCREEN flag is set, 0 otherwise.
- void **fullscreen_off** ()
Turns off any side effects of [fullscreen\(\)](#)
- void **fullscreen_off** (int *X*, int *Y*, int *W*, int *H*)
Turns off any side effects of [fullscreen\(\)](#) and does [resize\(x,y,w,h\)](#).
- void **fullscreen_screens** (int *top*, int *bottom*, int *left*, int *right*)
Sets which screens should be used when this window is in fullscreen mode.
- [uchar](#) **get_size_range** (int **minw*, int **minh*, int **maxw*=NULL, int **maxh*=NULL, int **dw*=NULL, int **dh*=NULL, int **aspect*=NULL)
Gets the allowable range to which the user can resize this window.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- void **hide** () [FL_OVERRIDE](#)
Removes the window from the screen.
- void **hotspot** (const [FL_Widget](#) &*p*, int *offscreen*=0)
*See void [FL_Window::hotspot](#)(int *x*, int *y*, int *offscreen* = 0)*
- void **hotspot** (const [FL_Widget](#) *, int *offscreen*=0)
*See void [FL_Window::hotspot](#)(int *x*, int *y*, int *offscreen* = 0)*
- void **hotspot** (int *x*, int *y*, int *offscreen*=0)
Positions the window so that the mouse is pointing at the given position, or at the center of the given widget, which may be the window itself.
- const void * **icon** () const
Gets the current icon window target dependent data.
- void **icon** (const [FL_RGB_Image](#) *)
Sets or resets a single window icon.
- void **icon** (const void **ic*)
Platform-specific method to set the window icon usable on Windows and X11 only.
- void **iconize** ()
Iconifies the window.
- const char * **iconlabel** () const
See void [FL_Window::iconlabel](#)(const char)*
- void **iconlabel** (const char *)
Sets the icon label.
- void **icons** (const [FL_RGB_Image](#) *[], int)
Sets the window icons.
- void **icons** (HICON *big_icon*, HICON *small_icon*)

- Sets the window icons using HICON handles (Windows platform only).*

 - const char * **label** () const

See void [Fl_Window::label\(const char\)](#)*
 - void **label** (const char *)

Sets the window title bar label.
 - void **label** (const char *label, const char *iconlabel)

Sets the icon label.
 - void [make_current](#) ()

Sets things up so that the drawing functions in <[FL/fl_draw.H](#)> will go into this window.
 - void [maximize](#) ()

Maximizes a top-level window to its current screen.
 - unsigned int **maximize_active** () const

Returns whether the window is currently maximized.
 - unsigned int **menu_window** () const

Returns true if this window is a menu window.
 - unsigned int [modal](#) () const

Returns true if this window is modal.
 - unsigned int **non_modal** () const

Returns true if this window is modal or non-modal.
 - [fl_uintptr_t os_id](#) ()

Returns a platform-specific identification of a shown window, or 0 if not shown.
 - unsigned int **override** () const

Returns non zero if OVERRIDE flag is set, 0 otherwise.
 - void [resize](#) (int X, int Y, int W, int H) [FL_OVERRIDE](#)

Changes the size and position of the window.
 - int [screen_num](#) ()

The number of the screen containing the mapped window.
 - void [screen_num](#) (int screen_num)

Set the number of the screen where to map the window.
 - void [set_menu_window](#) ()

Marks the window as a menu window.
 - void [set_modal](#) ()

A "modal" window, when [shown\(\)](#), will prevent any events from being delivered to other windows in the same program, and will also remain on top of the other windows (if the X window manager supports the "transient for" property).
 - void [set_non_modal](#) ()

A "non-modal" window (terminology borrowed from Microsoft Windows) acts like a [modal\(\)](#) one in that it remains on top, but it has no effect on event delivery.
 - void **set_override** ()

Activates the flags NOBORDER|OVERRIDE.
 - void [set_tooltip_window](#) ()

Marks the window as a tooltip window.
 - const [Fl_Image](#) * [shape](#) ()

Returns the image controlling the window shape or NULL.
 - void [shape](#) (const [Fl_Image](#) &b)

Set the window's shape with an [Fl_Image](#).
 - void [shape](#) (const [Fl_Image](#) *img)

Assigns a non-rectangular shape to the window.
 - void [show](#) () [FL_OVERRIDE](#)

Puts the window on the screen.
 - void [show](#) (int argc, char **argv)

- Puts the window on the screen with [show\(\)](#) and parses command-line arguments.*

 - `int shown ()`

Returns non-zero if [show\(\)](#) has been called (but not [hide\(\)](#)).
 - `void size\_range (int minw, int minh, int maxw=0, int maxh=0, int dw=0, int dh=0, int aspect=0)`

Sets the allowable range to which the user can resize this window.
 - `unsigned int tooltip\_window () const`

Returns true if this window is a tooltip window.
 - `void un\_maximize ()`

Returns a previously maximized top-level window to its previous size.
 - `void wait\_for\_expose ()`

Waits for the window to be displayed after calling [show\(\)](#).
 - `int x\_root () const`

Gets the x position of the window on the screen.
 - `const char * xclass () const`

Returns the xclass for this window, or a default.
 - `void xclass (const char *c)`

Sets the xclass for this window.
 - `int y\_root () const`

Gets the y position of the window on the screen.
 - `virtual ~FL\_Window ()`

The destructor also deletes all the children.

Public Member Functions inherited from [FL_Group](#)

- `FL\_Widget *& \_ddfdesign\_kludge ()`

This is for forms compatibility only.
- `void add (FL\_Widget &)`

The widget is removed from its current group (if any) and then added to the end of this group.
- `void add (FL\_Widget *o)`

See void [FL_Group::add\(FL_Widget &w\)](#)
- `void add\_resizable (FL\_Widget &o)`

Adds a widget to the group and makes it the resizable widget.
- `FL\_Widget *const * array () const`

Returns a pointer to the internal array of children.
- `FL\_Group const * as\_group () const FL\_OVERRIDE`
- `FL\_Group * as\_group () FL\_OVERRIDE`

Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- `void begin ()`

Sets the current group so you can build the widget tree by just constructing the widgets.
- `FL\_Widget * child (int n) const`

Returns the n'th child.
- `int children () const`

Returns how many child widgets the group has.
- `void clear ()`

Deletes all child widgets from memory recursively.
- `unsigned int clip\_children ()`

Returns the current clipping mode.
- `void clip\_children (int c)`

Controls whether the group widget clips the drawing of child widgets to its bounding box.
- `virtual int delete\_child (int n)`

*Removes the widget at *index* from the group and deletes it.*

- void **end** ()
Exactly the same as current(this->parent()).
- int **find** (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\) const](#).*
- int **find** (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char *s=0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void **focus** ([FL_Widget](#) *W)
- void **forms_end** ()
This is for forms compatibility only.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** ([FL_Widget](#) &o, [FL_Widget](#) *before)
This does insert(w, find(before)).
- void **remove** ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void **remove** ([FL_Widget](#) *o)
Removes the widget o from the group.
- void **remove** (int index)
Removes the widget at index from the group but does not delete it.
- [FL_Widget](#) * **resizable** () const
Returns the group's resizable widget.
- void **resizable** ([FL_Widget](#) &o)
Sets the group's resizable widget.
- void **resizable** ([FL_Widget](#) *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual [~FL_Group](#) ()
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- [FL_Align](#) **align** () const
Gets the label alignment.
- void **align** ([FL_Align](#) alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- void **bind_deimage** ([FL_Image](#) *img)

- Sets the image to use as part of the widget label when in the inactive state.*

 - void [bind_deimage](#) (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) (FI_Image *img)

Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype box](#) () const

Gets the box type of the widget.
- void [box](#) (FI_Boxtype new_box)

Sets the box type for the widget.
- [FI_Callback_p callback](#) () const

Gets the current callback function for the widget.
- void [callback](#) (FI_Callback *cb)

Sets the current callback function for the widget.
- void [callback](#) (FI_Callback *cb, FI_Callback_User_Data *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void [callback](#) (FI_Callback *cb, void *p)

Sets the current callback function and data for the widget.
- void [callback](#) (FI_Callback0 *cb)

Sets the current callback function for the widget.
- void [callback](#) (FI_Callback1 *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()

Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.
- void [clear_output](#) ()

Sets a widget to accept input.
- void [clear_visible](#) ()

Hides the widget.
- void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.
- [FI_Color color](#) () const

Gets the background color of the widget.
- void [color](#) (FI_Color bg)

Sets the background color of the widget.
- void [color](#) (FI_Color bg, FI_Color sel)

Sets the background and selection color of the widget.
- [FI_Color color2](#) () const

For back compatibility only.
- void [color2](#) (unsigned a)

For back compatibility only.
- int [contains](#) (const FI_Widget *w) const

Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)

Sets the current label.

- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) (uchar c)
Sets the damage bits for the widget.
- void [damage](#) (uchar c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [Fl_Image * deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [Fl_Image * deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([Fl_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([Fl_Widget](#) *widget, long arg, [Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([Fl_Widget](#) *widget, void *arg=0, [Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [Fl_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [Fl_Image * image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [Fl_Image * image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([Fl_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [Fl_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const

- Gets the current label text.*

 - void [label](#) (const char *text)
- Sets the current label pointer.*

 - void [label](#) (FI_Labeltype a, const char *b)
- Shortcut to set the label text and type in one call.*

 - int [label_image_spacing](#) ()
- Return the gap size between the label and the image.*

 - void [label_image_spacing](#) (int gap)
- Set the gap between the label and the image in pixels.*

 - [FI_Color](#) [labelcolor](#) () const
- Gets the label color.*

 - void [labelcolor](#) ([FI_Color](#) c)
- Sets the label color.*

 - [FI_Font](#) [labelfont](#) () const
- Gets the font to use.*

 - void [labelfont](#) ([FI_Font](#) f)
- Sets the font to use.*

 - [FI_Fonsize](#) [labelsize](#) () const
- Gets the font size in pixels.*

 - void [labelsize](#) ([FI_Fonsize](#) pix)
- Sets the font size in pixels.*

 - [FI_Labeltype](#) [labeltype](#) () const
- Gets the label type.*

 - void [labeltype](#) ([FI_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const
- Sets width ww and height hh accordingly with the label size.*

 - bool [needs_keyboard](#) () const
- Returns whether this widget needs a keyboard.*

 - void [needs_keyboard](#) (bool needs)
- Sets whether this widget needs a keyboard.*

 - unsigned int [output](#) () const
- Returns if a widget is used for output only.*

 - [FI_Group](#) * [parent](#) () const
- Returns a pointer to the parent widget.*

 - void [parent](#) ([FI_Group](#) *p)
- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
- Repositions the window or widget.*

 - void [redraw](#) ()
- Schedules the drawing of the widget.*

 - void [redraw_label](#) ()
- Schedules the drawing of the label.*

 - [FI_Color](#) [selection_color](#) () const
- Gets the selection color.*

 - void [selection_color](#) ([FI_Color](#) a)
- Sets the selection color.*

 - void [set_active](#) ()
- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const

- Gets the widget width.*
- [FL_When](#) [when](#) () const
 - Returns the conditions under which the callback is called.*
- void [when](#) (uchar i)
 - Sets the flags used to decide when a callback is called.*
- [FL_Window](#) * [window](#) () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
- int [x](#) () const
 - Gets the widget position in its window.*
- int [y](#) () const
 - Gets the widget position in its window.*
- virtual ~[FL_Widget](#) ()
 - Destroys the widget.*

Public Attributes

- void(* [display](#))()
- void(* [entry](#))(int)
- void(* [keyboard](#))(uchar, int x, int y)
- int [menu](#) [3]
- void(* [motion](#))(int x, int y)
- void(* [mouse](#))(int b, int state, int x, int y)
- int [number](#)
- void(* [overlaydisplay](#))()
- void(* [passivemotion](#))(int x, int y)
- void(* [reshape](#))(int w, int h)
- void(* [special](#))(int, int x, int y)
- void(* [visibility](#))(int)

Protected Member Functions

- void [draw](#) () [FL_OVERRIDE](#)
 - Draws the [FL_Gl_Window](#).*
- void [draw_overlay](#) () [FL_OVERRIDE](#)
 - You must implement this virtual function if you want to draw into the overlay.*
- int [handle](#) (int) [FL_OVERRIDE](#)
 - Handle some FLTK events as needed.*

Protected Member Functions inherited from [FL_Gl_Window](#)

- void [draw_begin](#) ()
 - Supports drawing to an [FL_Gl_Window](#) with the FLTK 2D drawing API.*
- void [draw_end](#) ()
 - To be used as a match for a previous call to [FL_Gl_Window::draw_begin\(\)](#).*

Protected Member Functions inherited from [FL_Window](#)

- void [default_size_range](#) ()
 - Protected method to calculate the default size range of a window.*
- void [draw](#) () [FL_OVERRIDE](#)
 - Draws the widget.*
- int [force_position](#) () const
 - Returns the internal state of the window's `FORCE_POSITION` flag.*
- void [force_position](#) (int force)

Sets an internal flag that tells FLTK and the window manager to honor position requests.

- void [free_icons](#) ()

Deletes all icons previously attached to the window.

- int [is_resizable](#) ()

Protected method to determine whether a window is resizable.

Protected Member Functions inherited from [Fl_Group](#)

- [Fl_Rect](#) * [bounds](#) ()

Returns the internal array of widget sizes and positions.

- void [draw_child](#) ([Fl_Widget](#) &widget) const

Forces a child to redraw.

- void [draw_children](#) ()

Draws all children of the group.

- void [draw_outside_label](#) (const [Fl_Widget](#) &widget) const

Parents normally call this to draw outside labels of child widgets.

- virtual int [on_insert](#) ([Fl_Widget](#) *, int)

Allow derived groups to act when a widget is added as a child.

- virtual int [on_move](#) (int, int)

Allow derived groups to act when a widget is moved within the group.

- virtual void [on_remove](#) (int)

Allow derived groups to act when a child widget is removed from the group.

- int * [sizes](#) ()

Returns the internal array of widget sizes and positions.

- void [update_child](#) ([Fl_Widget](#) &widget) const

Draws a child only if it needs it.

Protected Member Functions inherited from [Fl_Widget](#)

- void [clear_flag](#) (unsigned int c)

Clears a flag in the flags mask.

- void [draw_backdrop](#) () const

If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.

- void [draw_box](#) () const

Draws the widget box according its box style.

- void [draw_box](#) ([Fl_Boxtype](#) t, [Fl_Color](#) c) const

Draws a box of type t, of color c at the widget's position and size.

- void [draw_box](#) ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) c) const

Draws a box of type t, of color c at the position X,Y and size W,H.

- void [draw_focus](#) () const

Draws a focus rectangle around the widget.

- void [draw_focus](#) ([Fl_Boxtype](#) t, int X, int Y, int W, int H) const

Draws a focus rectangle around the widget.

- void [draw_focus](#) ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) bg) const

Draws a focus box for the widget at the given position and size.

- void [draw_label](#) () const

Draws the widget's label at the defined label position.

- void [draw_label](#) (int, int, int, int) const

Draws the label in an arbitrary bounding box.

- [Fl_Widget](#) (int x, int y, int w, int h, const char *label=0L)

Creates a widget at the given position and size.

- unsigned int [flags](#) () const

- Gets the widget flags mask.*
 - void **h** (int v)
 - Internal use only.*
 - void **set_flag** (unsigned int c)
 - Sets a flag in the flags mask.*
 - void **w** (int v)
 - Internal use only.*
 - void **x** (int v)
 - Internal use only.*
 - void **y** (int v)
 - Internal use only.*

Additional Inherited Members

Public Types inherited from [FI_Window](#)

- typedef struct HICON__ * **HICON**

Static Public Member Functions inherited from [FI_GI_Window](#)

- static int **can_do** (const int *m)
 - Returns non-zero if the hardware supports the given OpenGL mode.*
- static int **can_do** (int m)
 - Returns non-zero if the hardware supports the given OpenGL mode.*

Static Public Member Functions inherited from [FI_Window](#)

- static [FI_Window](#) * **current** ()
 - Returns the last window that was made current.*
- static void **default_callback** ([FI_Window](#) *, void *v)
 - Back compatibility: Sets the default callback v for win to call on close event.*
- static void **default_icon** (const [FI_RGB_Image](#) *)
 - Sets a single default window icon.*
- static void **default_icons** (const [FI_RGB_Image](#) *[], int)
 - Sets the default window icons.*
- static void **default_icons** (HICON big_icon, HICON small_icon)
 - Sets the default window icons (Windows platform only).*
- static const char * **default_xclass** ()
 - Returns the default xclass.*
- static void **default_xclass** (const char *)
 - Sets the default window xclass.*
- static bool **is_a_rescale** ()
 - Returns true when a window is being rescaled.*
- static char **show_next_window_iconic** ()
 - Returns the static flag whether the next window should be opened iconified.*
- static void **show_next_window_iconic** (char stat)
 - Sets a static flag whether the next window should be opened iconified.*

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * **current** ()
 - Returns the currently active group.*
- static void **current** ([FI_Group](#) *g)
 - Sets the current group.*

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Static Protected Attributes inherited from [Fl_Window](#)

- static [Fl_Window](#) * [current_](#)
Stores the last window that was made current.

11.55.1 Detailed Description

GLUT is emulated using this window class and these static variables (plus several more static variables hidden in `glut_compatibility.cxx`):

11.55.2 Member Function Documentation

11.55.2.1 `draw()`

```
void Fl_Glut_Window::draw (
    void ) [protected], [virtual]
```

Draws the [Fl_Gl_Window](#).

You **must** subclass [Fl_Gl_Window](#) and provide an implementation for `draw()`. You may also provide an implementation of `draw_overlay()` if you want to draw into the overlay planes. You can avoid reinitializing the viewport and lights and other things by checking `valid()` at the start of `draw()` and only doing the initialization if it is false.

The `draw()` method can *only* use OpenGL calls. Do not attempt to call X, any of the functions in `<FL/fl_draw.H>`, or glX directly. Do not call `gl_start()` or `gl_finish()`.

If double-buffering is enabled in the window, the back and front buffers are swapped after this function is completed. The following pseudo-code shows how to use "if (!valid())" to initialize the viewport:

```
void mywindow::draw() {
    if (!valid()) {
        glViewport(0,0,pixel_w(),pixel_h());
        glFrustum(...) or glOrtho(...)
        ...other initialization...
    }
    if (!context_valid()) {
        ...load textures, etc. ...
    }
    // clear screen
```

```

glClearColor(...);
glClear(...);
... draw your geometry here ...
}

```

Actual example code to clear screen to black and draw a 2D white "X":

```

void mywindow::draw() {
    if (!valid()) {
        glLoadIdentity();
        glViewport(0,0,pixel_w(),pixel_h());
        glOrtho(-w(),w(),-h(),h(),-1,1);
    }
    // Clear screen
    glClear(GL_COLOR_BUFFER_BIT);
    // Draw white 'X'
    glColor3f(1.0, 1.0, 1.0);
    glBegin(GL_LINE_STRIP); glVertex2f(w(), h()); glVertex2f(-w(),-h()); glEnd();
    glBegin(GL_LINE_STRIP); glVertex2f(w(),-h()); glVertex2f(-w(), h()); glEnd();
}

```

Regular FLTK widgets can be added as children to the [Fl_Gl_Window](#). To correctly overlay the widgets, [Fl_Gl_Window::draw\(\)](#) must be called after rendering the main scene.

```

void mywindow::draw() {
    // draw 3d graphics scene
    Fl_Gl_Window::draw();
    // -- or --
    draw_begin();
    Fl_Window::draw();
    // other 2d drawing calls, overlays, etc.
    draw_end();
}

```

Reimplemented from [Fl_Gl_Window](#).

11.55.2.2 draw_overlay()

```
void Fl_Glut_Window::draw_overlay () [protected], [virtual]
```

You must implement this virtual function if you want to draw into the overlay.

The overlay is cleared before this is called. You should draw anything that is not clear using OpenGL. You must use `gl_color(i)` to choose colors (it allocates them from the colormap using system-specific calls), and remember that you are in an indexed OpenGL mode and drawing anything other than flat-shaded will probably not work.

Both this function and [Fl_Gl_Window::draw\(\)](#) should check [Fl_Gl_Window::valid\(\)](#) and set the same transformation. If you don't your code may not work on other systems. Depending on the OS, and on whether overlays are real or simulated, the OpenGL context may be the same or different between the overlay and main window.

Reimplemented from [Fl_Gl_Window](#).

11.55.2.3 handle()

```
int Fl_Glut_Window::handle (
    int event) [protected], [virtual]
```

Handle some FLTK events as needed.

Reimplemented from [Fl_Gl_Window](#).

The documentation for this class was generated from the following files:

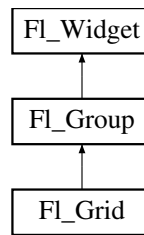
- glut.H
- glut_compatibility.cxx

11.56 Fl_Grid Class Reference

[Fl_Grid](#) is a container (layout) widget with multiple columns and rows.

```
#include <Fl_Grid.H>
```

Inheritance diagram for [Fl_Grid](#):



Classes

- class [Cell](#)

Public Member Functions

- [Fl_Grid::Cell * cell](#) ([Fl_Widget *widget](#)) const
Get the grid cell of widget `widget`.
- [Fl_Grid::Cell * cell](#) (int row, int col) const
Get the grid cell of row `row` and column `col`.
- virtual void [clear_layout](#) ()
Reset the layout w/o removing widgets.
- void [col_gap](#) (const int *value, [size_t size](#))
Set more than one column gaps at once.
- int [col_gap](#) (int col) const
- void [col_gap](#) (int col, int value)
Set the gap of column `col`.
- void [col_weight](#) (const int *value, [size_t size](#))
Set the weight of more than one column.
- int [col_weight](#) (int col) const
- void [col_weight](#) (int col, int value)
Set the weight of a column.
- void [col_width](#) (const int *value, [size_t size](#))
Set minimal widths of more than one column.
- int [col_width](#) (int col) const
- void [col_width](#) (int col, int value)
Set the minimal width of a column.
- short [cols](#) () const
- int [computed_col_width](#) (int col) const
- int [computed_row_height](#) (int row) const
- void [debug](#) (int level=127)
Output layout information of this [Fl_Grid](#) to stderr.
- [Fl_Grid](#) (int X, int Y, int W, int H, const char *L=0)
Create a new [Fl_Grid](#) widget.
- void [gap](#) (int *row_gap, int *col_gap) const
Get the default gaps for rows and columns.
- virtual void [gap](#) (int row_gap, int col_gap=-1)
Set default gaps for rows and columns.
- virtual void [layout](#) ()
Calculate the grid layout and resize and position all widgets.
- virtual void [layout](#) (int rows, int cols, int margin=-1, int gap=-1)
Set the basic layout parameters of the [Fl_Grid](#) widget.
- int [margin](#) (int *left, int *top, int *right, int *bottom) const
Returns all outside margin sizes of the grid.

- virtual void **margin** (int left, int top=-1, int right=-1, int bottom=-1)
Set all margins (left, top, right, bottom).
- bool **need_layout** () const
Return whether layout calculation is required.
- void **need_layout** (int set)
Request or reset the request to calculate the layout of children.
- virtual void **resize** (int X, int Y, int W, int H) **FL_OVERRIDE**
Recalculate the layout and position and resize all widgets.
- void **row_gap** (const int *value, size_t size)
Set more than one row gaps at once.
- int **row_gap** (int row) const
- void **row_gap** (int row, int value)
Set the gap of row row.
- void **row_height** (const int *value, size_t size)
Set the minimal row height of more than one row.
- int **row_height** (int row) const
- void **row_height** (int row, int value)
Set the minimal row height of row row.
- void **row_weight** (const int *value, size_t size)
Set the weight of more than one row.
- int **row_weight** (int row) const
- void **row_weight** (int row, int value)
Set the row weight of row row.
- short **rows** () const
- void **show_grid** (int set)
Enable or disable drawing of the grid helper lines for visualization.
- void **show_grid** (int set, **FL_Color** col)
Enable or disable drawing of the grid helper lines for visualization.
- **FL_Grid::Cell** * **widget** (**FL_Widget** *wi, int row, int col, **FL_Grid_Align** align=**FL_GRID_FILL**)
Assign a widget to a grid cell and set its alignment.
- **FL_Grid::Cell** * **widget** (**FL_Widget** *wi, int row, int col, int rowspan, int colspan, **FL_Grid_Align** align=**FL_GRID_FILL**)
Assign a widget to a grid cell and set cell spanning and alignment.

Public Member Functions inherited from **FL_Group**

- **FL_Widget** *& **_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** (**FL_Widget** &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** (**FL_Widget** *o)
*See void **FL_Group::add(FL_Widget &w)***
- void **add_resizable** (**FL_Widget** &o)
Adds a widget to the group and makes it the resizable widget.
- **FL_Widget** *const * **array** () const
Returns a pointer to the internal array of children.
- **FL_Group** const * **as_group** () const **FL_OVERRIDE**
- **FL_Group** * **as_group** () **FL_OVERRIDE**
*Returns an **FL_Group** pointer if this widget is an **FL_Group**.*
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- **FL_Widget** * **child** (int n) const

- Returns the n'th child.*

 - int **children** () const

Returns how many child widgets the group has.
- void **clear** ()

Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()

Returns the current clipping mode.
- void **clip_children** (int c)

Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)

*Removes the widget at *index* from the group and deletes it.*
- void **end** ()

*Exactly the same as *current(this->parent())*.*
- int **find** (const FL_Widget &o) const

*See int FL_Group::find(const FL_Widget *w) const.*
- int **find** (const FL_Widget *) const

Searches the children for the widget and returns the index.
- FL_Group (int, int, int, int, const char *s=0)

Creates a new FL_Group widget using the given position, size, and label string.
- void **focus** (FL_Widget *w)
- void **forms_end** ()

This is for forms compatibility only.
- int **handle** (int) FL_OVERRIDE

Handles the specified event.
- void **init_sizes** ()

Resets the internal array of widget sizes and positions.
- void **insert** (FL_Widget &, int i)

The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** (FL_Widget &o, FL_Widget *before)

This does insert(w, find(before)).
- void **remove** (FL_Widget &)

Removes a widget from the group but does not delete it.
- void **remove** (FL_Widget *o)

Removes the widget o from the group.
- void **remove** (int index)

*Removes the widget at *index* from the group but does not delete it.*
- FL_Widget * **resizable** () const

Returns the group's resizable widget.
- void **resizable** (FL_Widget &o)

Sets the group's resizable widget.
- void **resizable** (FL_Widget *o)

The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- void **resize** (int, int, int, int) FL_OVERRIDE

Resizes the FL_Group widget and all of its children.
- virtual ~FL_Group ()

The destructor also deletes all the children.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

- Marks the widget as inactive without sending events or changing focus.*

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*

 - void `clear_damage` (uchar c=0)
- Clears or sets the damage flags.*

 - void `clear_output` ()
- Sets a widget to accept input.*

 - void `clear_visible` ()
- Hides the widget.*

 - void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*

 - `FL_Color` `color` () const
- Gets the background color of the widget.*

 - void `color` (FL_Color bg)
- Sets the background color of the widget.*

 - void `color` (FL_Color bg, FL_Color sel)
- Sets the background and selection color of the widget.*

 - `FL_Color` `color2` () const
- For back compatibility only.*

 - void `color2` (unsigned a)
- For back compatibility only.*

 - int `contains` (const FL_Widget *w) const
- Checks if w is a child of this widget.*

 - void `copy_label` (const char *new_label)
- Sets the current label.*

 - void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*

 - uchar `damage` () const
- Returns non-zero if `draw()` needs to be called.*

 - void `damage` (uchar c)
- Sets the damage bits for the widget.*

 - void `damage` (uchar c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int `damage_resize` (int, int, int, int)
- Internal use only.*

 - void `deactivate` ()
- Deactivates the widget.*

 - `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*

 - const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*

 - void `deimage` (FL_Image &img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - void `deimage` (FL_Image *img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*

 - void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)
- Calls the widget callback function with default arguments.*

 - void `do_callback` (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)
- Calls the widget callback function with arbitrary arguments.*

- void `do_callback` (`FL_Widget *widget`, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image * image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image &img`)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image *img`)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget *wgt`) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize labelsz` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype labeltype` () const
Gets the label type.
- void `labeltype` (`FL_Labeltype` a)

- Sets the label type.*

 - void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FI_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FI_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- `FI_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`FI_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.
- void `set_visible` ()

Makes the widget visible.
- void `set_visible_focus` ()

Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()

Makes a widget visible.
- void `size` (int W, int H)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const

Gets the current tooltip text.
- void `tooltip` (const char *text)

Sets the current tooltip text.

- `Fl_Window * top_window ()` const
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff)` const
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type ()` const
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data ()` const
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible ()` const
Returns whether a widget is visible.
- `unsigned int visible_focus ()` const
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r ()` const
Returns whether a widget and all its parents are visible.
- `int w ()` const
Gets the widget width.
- `Fl_When when ()` const
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window ()` const
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x ()` const
Gets the widget position in its window.
- `int y ()` const
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- `Cell * add_cell (int row, int col)`
- `virtual void draw ()` `FL_OVERRIDE`
Draws the `Fl_Grid` widget and all children.
- `virtual void draw_grid ()`
Draws the grid helper lines for design and debugging purposes.
- `void init ()`
- `void on_remove (int)` `FL_OVERRIDE`
`Fl_Group` calls this method when a child widget is about to be removed.
- `void remove_cell (int row, int col)`

Protected Member Functions inherited from FL_Group

- `FL_Rect * bounds ()`
Returns the internal array of widget sizes and positions.
- `void draw () FL_OVERRIDE`
Draws the widget.
- `void draw_child (FL_Widget &widget) const`
Forces a child to redraw.
- `void draw_children ()`
Draws all children of the group.
- `void draw_outside_label (const FL_Widget &widget) const`
Parents normally call this to draw outside labels of child widgets.
- `virtual int on_insert (FL_Widget *, int)`
Allow derived groups to act when a widget is added as a child.
- `virtual int on_move (int, int)`
Allow derived groups to act when a widget is moved within the group.
- `int * sizes ()`
Returns the internal array of widget sizes and positions.
- `void update_child (FL_Widget &widget) const`
Draws a child only if it needs it.

Protected Member Functions inherited from FL_Widget

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const`
Draws a focus box for the widget at the given position and size.
- `void draw_label () const`
Draws the widget's label at the defined label position.
- `void draw_label (int, int, int, int) const`
Draws the label in an arbitrary bounding box.
- `FL_Widget (int x, int y, int w, int h, const char *label=0L)`
Creates a widget at the given position and size.
- `unsigned int flags () const`
Gets the widget flags mask.
- `void h (int v)`
Internal use only.
- `void set_flag (unsigned int c)`
Sets a flag in the flags mask.
- `void w (int v)`

Internal use only.

- void `x` (int v)

Internal use only.

- void `y` (int v)

Internal use only.

Protected Attributes

- bool `draw_grid_`
- `Fl_Color` `grid_color`

Friends

- class `Fl_Grid_Type`

Additional Inherited Members

Static Public Member Functions inherited from `Fl_Group`

- static `Fl_Group * current` ()
Returns the currently active group.
- static void `current` (`Fl_Group *g`)
Sets the current group.

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget *widget`, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `Fl_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

11.56.1 Detailed Description

[FL_Grid](#) is a container (layout) widget with multiple columns and rows.

This container widget features very flexible layouts in columns and rows w/o the need to position each child widget in x/y coordinates.

Widgets are assigned to grid cells (column, row) with their minimal sizes in `w()` and `h()`. The `x()` and `y()` positions are ignored and can be (0, 0). [FL_Grid](#) calculates widget positions and resizes the widgets to fit into the grid. It is possible to create a single row or column of widgets with [FL_Grid](#).

You should design your grid with the smallest possible sizes of all widgets in mind. [FL_Grid](#) will automatically assign additional space to cells according to some rules (described later) when resizing the [FL_Grid](#) widget.

Hint: You should set a minimum window size to make sure the [FL_Grid](#) is never resized below its minimal sizes. Resizing below the given widget sizes results in undefined behavior.

[FL_Grid](#) and other container widgets (e.g. [FL_Group](#)) can be nested. One main advantage of this usage is that widget coordinates in embedded [FL_Group](#) widgets become relative to the group and will be positioned as expected.

Todo This (relative group coordinates of nested groups of [FL_Grid](#)) needs explanation and maybe an example.

[FL_Grid](#) child widgets are handled by its base class [FL_Group](#) but [FL_Grid](#) stores additional data corresponding to each widget in internal grid cells.

[FL_Grid](#) children are allowed to span multiple columns and rows like HTML `<table>` cells. Individual children can have fixed sizes or be aligned inside their cells (left, right, top, bottom, and more) and/or follow their cell sizes when the [FL_Grid](#) container is resized.

Note to resizing: since [FL_Grid](#) uses its own layout algorithm the normal [FL_Group::resizable\(\)](#) widget is ignored (if set). Calling `init_sizes()` is not necessary.

Note

[FL_Grid](#) is, as of FLTK 1.4.x, still in experimental state and should be used with caution. The API can still be changed although it is assumed to be almost stable - as stable as possible for a first release.

Example: Simple 3x3 [FL_Grid](#) with five buttons:

```
#include <FL/Fl.H>
#include <FL/Fl_Double_Window.H>
#include <FL/Fl_Grid.H>
#include <FL/Fl_Button.H>

int main(int argc, char **argv) {
    Fl_Double_Window *win = new Fl_Double_Window(320, 180, "3x3 FL_Grid with Buttons");
    // create the FL_Grid container with five buttons
    Fl_Grid *grid = new Fl_Grid(0, 0, win->w(), win->h());
    grid->layout(3, 3, 10, 10);
    grid->color(FL_WHITE);
    Fl_Button *b0 = new Fl_Button(0, 0, 0, 0, "New");
    Fl_Button *b1 = new Fl_Button(0, 0, 0, 0, "Options");
    Fl_Button *b3 = new Fl_Button(0, 0, 0, 0, "About");
    Fl_Button *b4 = new Fl_Button(0, 0, 0, 0, "Help");
    Fl_Button *b6 = new Fl_Button(0, 0, 0, 0, "Quit");
    // assign buttons to grid positions
    grid->widget(b0, 0, 0);
    grid->widget(b1, 0, 2);
    grid->widget(b3, 1, 1);
    grid->widget(b4, 2, 0);
    grid->widget(b6, 2, 2);
    grid->show_grid(0); // 1 to display grid helper lines
    grid->end();
    win->end();
    win->resizable(grid);
    win->size_range(300, 100);
    win->show(argc, argv);
    return Fl::run();
}
```

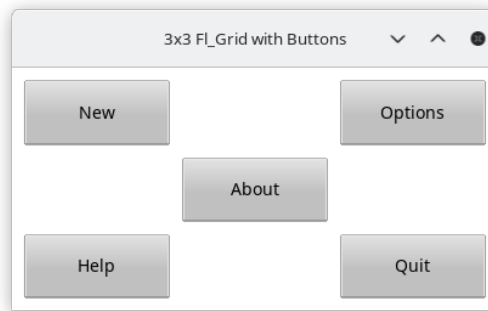


Figure 11.19 Simple 3x3 Fl_Grid



Figure 11.20 show_grid() set to 1

11.56.2 Constructor & Destructor Documentation

11.56.2.1 Fl_Grid()

```
Fl_Grid::Fl_Grid (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Create a new [Fl_Grid](#) widget.

Todo More documentation of [Fl_Grid](#) constructor?

11.56.3 Member Function Documentation

11.56.3.1 cell() [1/2]

```
Fl_Grid::Cell * Fl_Grid::cell (
    Fl_Widget * widget) const
```

Get the grid cell of widget `widget`.

The pointer to the cell can be used for further assignment of properties like alignment etc.

Hint: If you know the row and column index of the cell you should use `Fl_Grid::cell(int row, int col)` instead because it is **much** faster.

Please see `Fl_Grid::cell(int row, int col)` for details and the validity of cell pointers.

Parameters

in	<i>widget</i>	widget whose cell is requested

Return values

<code>NULL</code>	if <code>widget</code> is not assigned to a cell

11.56.3.2 `cell()` [2/2]

```
Fl_Grid::Cell * Fl_Grid::cell (
    int row,
    int col) const
```

Get the grid cell of row `row` and column `col`.

Widgets and other attributes are organized in cells ([Fl_Grid::Cell](#)).

This cell is an opaque structure (class) with some public methods. **Don't** assume anything about grid cell sizes and ordering in memory. These are implementation details that can be changed without notice.

The validity of an [Fl_Grid::Cell](#) pointer is limited. It will definitely be invalidated when the overall grid layout is changed, for instance by calling `layout(int, int)`.

Adding new cells beyond the current layout limits will also invalidate cell pointers but this is not (yet) implemented. Attempts to assign widgets to out-of-bounds cells are currently ignored.

The only well-defined usage of cell pointers is to set one or more properties like widget alignment of a cell after retrieving the cell pointer. Don't store cell pointers in your program for later reference.

Parameters

in	<i>row</i>	row index
in	<i>col</i>	column index

Returns

pointer to cell

Return values

<code>NULL</code>	if <code>row</code> or <code>col</code> is out of bounds or no widget was assigned

11.56.3.3 `clear_layout()`

```
void Fl_Grid::clear_layout () [virtual]
```

Reset the layout w/o removing widgets.

Removes all cells and sets rows and cols to zero. Existing widgets are kept as children of the [Fl_Group](#) (base class) but are hidden.

This method should be rarely used. You may want to call [Fl_Grid::clear\(\)](#) to remove all widgets and reset the layout to zero rows and columns.

You must call `layout(int rows, int cols, ...)` to set a new layout, allocate new cells, and assign widgets to new cells.

Todo [Fl_Grid::clear\(\)](#) needs to be implemented as documented above!

11.56.3.4 `col_gap()` [1/2]

```
void Fl_Grid::col_gap (
    const int * value,
    size_t size)
```

Set more than one column gaps at once.

See also

[Fl_Grid::col_weight\(const int *value, size_t size\)](#) for handling of the value `array` and `size`.

11.56.3.5 col_gap() [2/2]

```
void Fl_Grid::col_gap (
    int col,
    int value)
```

Set the gap of column `col`.

Note that the gap is right of each column except the last one which is ignored. Use [margin\(\)](#) for the right most column.

Parameters

in	<i>col</i>	column
in	<i>value</i>	gap size after the column

11.56.3.6 col_weight() [1/2]

```
void Fl_Grid::col_weight (
    const int * value,
    size_t size)
```

Set the weight of more than one column.

The values are taken from the array `value` and assigned sequentially to columns, starting from column 0. If the array `size` is too large extraneous values are ignored.

Negative values in the array are not assigned to their columns, i.e. the existing value for the corresponding column is not changed.

Example:

```
int val[] = { 0, 0, 50, -1, -1, 50, 0 };
grid->col_weight(val, sizeof(val)/sizeof(val[0]));
```

Parameters

in	<i>value</i>	an array of column weights
in	<i>size</i>	the size of the array (number of values)

11.56.3.7 col_weight() [2/2]

```
void Fl_Grid::col_weight (
    int col,
    int value)
```

Set the weight of a column.

Column and row weights are used to distribute additional space when the grid is resized beyond its defined (minimal) size. All weight values are relative and can be chosen freely. Suggested weights are in the range {0 .. 100}, 0 (zero) disables resizing of the column.

How does it work?

Whenever additional space (say: `SPACE` in pixels) is to be distributed to a set of columns the weights of all columns are added to a value `SUM`, then every single column width is increased by the value (in pseudo code):

```
col.width += SPACE * col.weight / SUM
```

Resulting pixel values are rounded to the next integer and rounding differences are added to or subtracted from the column with the highest weight. If more columns have the same weight one of them is chosen.

Note

If none of the columns considered for resizing have weights > 0 then `Fl_Grid` assigns the remaining space to an arbitrary column or to all considered columns evenly. This is implementation defined and can be changed without notice. You can avoid this situation by designing your grid with sensible sizes and weights.

Parameters

in	<i>col</i>	column number (counting from 0)

in	<i>value</i>	weight, must be ≥ 0

11.56.3.8 col_width() [1/2]

```
void Fl_Grid::col_width (
    const int * value,
    size_t size)
```

Set minimal widths of more than one column.

The values are taken from the array *value* and assigned sequentially to columns, starting from column 0. If the array *size* is too large extraneous values are ignored.

Negative values in the array are not assigned to their columns, i.e. the existing value for the corresponding column is not changed.

Example:

```
int widths[] = { 0, 0, 50, -1, -1, 50, 0 };
grid->col_width(widths, sizeof(width)/sizeof(width[0]));
```

Parameters

in	<i>value</i>	an array of column widths
in	<i>size</i>	the size of the array (number of values)

11.56.3.9 col_width() [2/2]

```
void Fl_Grid::col_width (
    int col,
    int value)
```

Set the minimal width of a column.

Column widths are calculated by using the maximum of all widget widths in that column and the given column width.

After calculating the width additional space is added when resizing according to the *weight* of the column.

You can set one or more column widths in one call by using [Fl_Grid::col_width\(const int *value, size_t size\)](#).

Parameters

in	<i>col</i>	column number (counting from 0)
in	<i>value</i>	minimal column width, must be ≥ 0

See also

[Fl_Grid::col_width\(const int *value, size_t size\)](#)

11.56.3.10 debug()

```
void Fl_Grid::debug (
    int level = 127)
```

Output layout information of this [Fl_Grid](#) to stderr.

Parameter *level* will be used to define the amount of output.

- 0 = nothing
- 127 = everything
- other values not yet defined

Note

It is not yet defined which kind of values `level` will have, either a numerical value (127 = maximum, 0 = nothing) or a bit mask that determines what to output.

Todo Add more information about cells and children.

Control output by using `level`.

Parameters

<code>in</code>	<code>level</code>	not yet used (0-127, default = 127)

11.56.3.11 draw()

```
void Fl_Grid::draw (
    void ) [protected], [virtual]
```

Draws the [Fl_Grid](#) widget and all children.

If the layout has been changed [layout\(\)](#) is called before the widget is drawn so all children are arranged as designed.

See also

[layout\(\)](#)

[need_layout\(\)](#)

Implements [Fl_Widget](#).

11.56.3.12 draw_grid()

```
void Fl_Grid::draw_grid () [protected], [virtual]
```

Draws the grid helper lines for design and debugging purposes.

This method is protected so it can be modified in subclasses.

11.56.3.13 gap() [1/2]

```
void Fl_Grid::gap (
    int * row_gap,
    int * col_gap) const
```

Get the default gaps for rows and columns.

Parameters

<code>out</code>	<code>row_gap</code>	pointer to int to receive column gap, may be NULL
<code>out</code>	<code>col_gap</code>	pointer to int to receive column gap, may be NULL

11.56.3.14 gap() [2/2]

```
void Fl_Grid::gap (
    int row_gap,
    int col_gap = -1) [virtual]
```

Set default gaps for rows and columns.

All gaps are positioned below the rows and right of their columns.

The bottom row and the right-most column don't have a gap, i.e. the gap sizes of these columns and rows are ignored. You can use a right or bottom margin instead.

You have to specify at least one argument, `col_gap` is optional. If you don't specify an argument or use a negative value (e.g. -1) then that margin is not affected.

You can also initialize the default gaps with [layout\(int, int, int, int\)](#).

Parameters

in	<i>row_gap</i>	default gap for all rows
in	<i>col_gap</i>	default gap for all columns

See also

[Fl_Grid::layout\(int rows, int cols, int margin, int gap\)](#)

11.56.3.15 layout() [1/2]

```
void Fl_Grid::layout () [virtual]
```

Calculate the grid layout and resize and position all widgets.

This is called automatically when the [Fl_Grid](#) is resized. You need to call it once after you added widgets or moved widgets between cells.

Calling it once after all modifications are completed is enough.

Todo Document when and why to call [layout\(\)](#) w/o args. See [Fl_Flex::layout\(\)](#)

See also

[Fl_Grid::layout\(int rows, int cols, int margin, int gap\)](#)

11.56.3.16 layout() [2/2]

```
void Fl_Grid::layout (
    int rows,
    int cols,
    int margin = -1,
    int gap = -1) [virtual]
```

Set the basic layout parameters of the [Fl_Grid](#) widget.

You need to specify at least `rows` and `cols` to define a layout before you can add widgets to the grid.

Parameters `margin` and `gap` are optional.

You can call [layout\(int rows, int cols, int margin, int gap\)](#) again to change the layout but this is inefficient since all cells are reallocated if the layout changed.

Calling this with the same values of `rows` and `cols` is fast and can be used to change `margin` and `gap` w/o reallocating the cells.

`margin` sets all margins (left, top, right, bottom) to the same value. Negative values (e.g. -1) don't change the established margins. The default value set by the constructor is 0.

`gap` sets row and column gaps to the same value. Negative values (e.g. -1) do not affect the established gaps. The default value set by the constructor is 0.

After you added all widgets you must call [layout\(\)](#) once without arguments to calculate the actual layout and to position and resize all widgets.

Todo Document when and why to call [layout\(\)](#) w/o args. See [Fl_Flex::layout\(\)](#)

Parameters

in	<i>rows</i>	number of rows
in	<i>cols</i>	number of columns
in	<i>margin</i>	margin size inside the Fl_Grid 's border
in	<i>gap</i>	gap size between cells

See also

[Fl_Grid::layout\(\)](#)

11.56.3.17 margin() [1/2]

```
int Fl_Grid::margin (
    int * left,
    int * top,
    int * right,
    int * bottom) const
```

Returns all outside margin sizes of the grid.

All margin sizes are returned in the given arguments. If any argument is `NULL` the respective value is not returned.

Parameters

out	<i>left</i>	returns left margin if not <code>NULL</code>
out	<i>top</i>	returns top margin if not <code>NULL</code>
out	<i>right</i>	returns right margin if not <code>NULL</code>
out	<i>bottom</i>	returns bottom margin if not <code>NULL</code>

Returns

whether all margins are equal

Return values

1	all margins have the same size
0	at least one margin has a different size

11.56.3.18 margin() [2/2]

```
void Fl_Grid::margin (
    int left,
    int top = -1,
    int right = -1,
    int bottom = -1) [virtual]
```

Set all margins (left, top, right, bottom).

All margins are measured in pixels inside the box borders. You need to specify at least one argument, all other arguments are optional. If you don't specify an argument or use a negative value (e.g. -1) then that particular margin is not affected.

Parameters

in	<i>left</i>	left margin
in	<i>top</i>	top margin
in	<i>right</i>	right margin
in	<i>bottom</i>	bottom margin

11.56.3.19 need_layout()

```
void Fl_Grid::need_layout (
    int set) [inline]
```

Request or reset the request to calculate the layout of children.

If called with `true` (1) this calls [redraw\(\)](#) to schedule a full [draw\(\)](#). When draw is eventually called, the layout is (re)calculated before actually drawing the widget.

Parameters

in	set	1 to request layout calculation, 0 to reset the request

11.56.3.20 on_remove()

```
void Fl_Grid::on_remove (
    int index) [protected], [virtual]
```

[Fl_Group](#) calls this method when a child widget is about to be removed.

Make sure that the widget is also removed from our internal list of children.

Reimplemented from [Fl_Group](#).

11.56.3.21 resize()

```
void Fl_Grid::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Recalculate the layout and position and resize all widgets.

This method overrides [Fl_Group::resize\(\)](#) and calculates all positions and sizes of its children according to its own rules.

Parameters

in	<i>X,Y</i>	new widget position
in	<i>W,H</i>	new widget size

Reimplemented from [Fl_Widget](#).

11.56.3.22 row_gap() [1/2]

```
void Fl_Grid::row_gap (
    const int * value,
    size_t size)
```

Set more than one row gaps at once.

See also

[Fl_Grid::col_weight\(const int *value, size_t size\)](#) for handling of the value [array](#) and [size](#).

11.56.3.23 row_gap() [2/2]

```
void Fl_Grid::row_gap (
    int row,
    int value)
```

Set the gap of row `row`.

Note that the gap is below each row except the last one which is ignored. Use [margin\(\)](#) for the bottom row.

Parameters

in	<i>row</i>	row
in	<i>value</i>	gap size below the row

11.56.3.24 row_height() [1/2]

```
void Fl_Grid::row_height (
    const int * value,
    size_t size)
```

Set the minimal row height of more than one row.

Parameters

in	<i>value</i>	array of height values
in	<i>size</i>	size of array <i>value</i>

See also

[Fl_Grid::col_weight\(const int *value, size_t size\)](#) for handling of the value [array](#) and [size](#).

11.56.3.25 row_height() [2/2]

```
void Fl_Grid::row_height (
    int row,
    int value)
```

Set the minimal row height of row *row*.

Parameters

in	<i>row</i>	<i>row</i>
in	<i>value</i>	minimal height of the row

11.56.3.26 row_weight() [1/2]

```
void Fl_Grid::row_weight (
    const int * value,
    size_t size)
```

Set the weight of more than one row.

Parameters

in	<i>value</i>	array of height values
in	<i>size</i>	size of array <i>value</i>

See also

[Fl_Grid::col_weight\(const int *value, size_t size\)](#) for handling of the value [array](#) and [size](#).

11.56.3.27 row_weight() [2/2]

```
void Fl_Grid::row_weight (
    int row,
    int value)
```

Set the row weight of row *row*.

Parameters

in	<i>row</i>	<i>row</i>
in	<i>value</i>	weight of the row

11.56.3.28 show_grid() [1/2]

```
void Fl_Grid::show_grid (
    int set) [inline]
```

Enable or disable drawing of the grid helper lines for visualization.

Use this method during the design stage of your [Fl_Grid](#) widget or for debugging if widgets are not positioned as intended.

The default is a light green color but you can change it for better contrast if needed, see [show_grid\(int set, Fl_Color col\)](#).

Note

You can define the environment variable `FLTK_GRID_DEBUG=1` to set `show_grid(1)` for all [Fl_Grid](#) widgets at construction time. This enables you to debug the grid layout w/o changing code.

Parameters

in	set	1 (true) = draw, 0 = don't draw the grid

See also

[show_grid\(int set, Fl_Color col\)](#)

11.56.3.29 show_grid() [2/2]

```
void Fl_Grid::show_grid (
    int set,
    Fl_Color col) [inline]
```

Enable or disable drawing of the grid helper lines for visualization.

This method also sets the color used for the helper lines.

The default is a light green color but you can change it to any color for better contrast if needed.

Parameters

in	set	1 (true) = draw, 0 = don't draw the grid
in	col	color to use for the grid helper lines

See also

[show_grid\(int set\)](#)

11.56.3.30 widget() [1/2]

```
Fl_Grid::Cell * Fl_Grid::widget (
    Fl_Widget * wi,
    int row,
    int col,
    Fl_Grid_Align align = FL_GRID_FILL)
```

Assign a widget to a grid cell and set its alignment.

This short form sets row and column spanning attributes to (1, 1).

For more information see [Fl_Grid::widget\(Fl_Widget *wi, int row, int col, int rowspan, int colspan, Fl_Grid_Align align\)](#)

Parameters

in	wi	widget to be assigned to the cell
in	row	row
in	col	column
in	align	widget alignment inside the cell

Returns

assigned cell

Return values

<code>NULL</code>	if <code>row</code> or <code>col</code> is out of bounds

See also

[Fl_Grid::widget\(Fl_Widget *wi, int row, int col, int rowspan, int colspan, Fl_Grid_Align align\)](#)

11.56.3.31 widget() [2/2]

```
Fl_Grid::Cell * Fl_Grid::widget (
    Fl_Widget * wi,
    int row,
    int col,
    int rowspan,
    int colspan,
    Fl_Grid_Align align = FL_GRID_FILL)
```

Assign a widget to a grid cell and set cell spanning and alignment.

Default alignment is `FL_GRID_FILL` which stretches the widget in horizontal and vertical directions to fill the whole cell(s) given by `colspan` and `rowspan`.

You can use this method to move a widget from one cell to another; it is automatically removed from its old cell. If the new cell is already assigned to another widget that widget is deassigned but kept as a child of the group.

Before you can assign a widget to a cell it must have been created as a child of the [Fl_Grid](#) widget (i.e. its [Fl_Group](#)).

Parameters

in	<i>wi</i>	widget to be assigned to the cell
in	<i>row</i>	row
in	<i>col</i>	column
in	<i>rowspan</i>	vertical span in cells, default 1
in	<i>colspan</i>	horizontal span in cells, default 1
in	<i>align</i>	widget alignment inside the cell

Returns

assigned cell

Return values

<code>NULL</code>	if <code>row</code> or <code>col</code> is out of bounds or <code>wi</code> is not a child

The documentation for this class was generated from the following files:

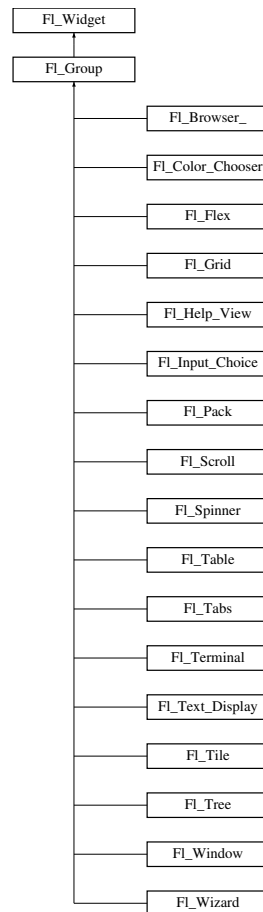
- [Fl_Grid.H](#)
- [Fl_Grid.cxx](#)

11.57 Fl_Group Class Reference

The [Fl_Group](#) class is the main FLTK container widget.

```
#include <Fl_Group.H>
```

Inheritance diagram for [Fl_Group](#):



Public Member Functions

- [FL_Widget](#) * & **_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void **add_resizable** ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * **array** () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * **as_group** () const [FL_OVERRIDE](#)
- [FL_Group](#) * **as_group** () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.

- void `clip_children` (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int `delete_child` (int n)
Removes the widget at `index` from the group and deletes it.
- void `end` ()
Exactly the same as `current(this->parent())`.
- int `find` (const `FL_Widget` &o) const
See int `FL_Group::find(const FL_Widget *w)` const.
- int `find` (const `FL_Widget` *) const
Searches the children for the widget and returns the index.
- `FL_Group` (int, int, int, int, const char *s=0)
Creates a new `FL_Group` widget using the given position, size, and label string.
- void `focus` (`FL_Widget` *W)
- void `forms_end` ()
This is for forms compatibility only.
- int `handle` (int) `FL_OVERRIDE`
Handles the specified event.
- void `init_sizes` ()
Resets the internal array of widget sizes and positions.
- void `insert` (`FL_Widget` &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void `insert` (`FL_Widget` &o, `FL_Widget` *before)
This does `insert(w, find(before))`.
- void `remove` (`FL_Widget` &)
Removes a widget from the group but does not delete it.
- void `remove` (`FL_Widget` *o)
Removes the widget o from the group.
- void `remove` (int index)
Removes the widget at `index` from the group but does not delete it.
- `FL_Widget` * `resizable` () const
Returns the group's resizable widget.
- void `resizable` (`FL_Widget` &o)
Sets the group's resizable widget.
- void `resizable` (`FL_Widget` *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- void `resize` (int, int, int, int) `FL_OVERRIDE`
Resizes the `FL_Group` widget and all of its children.
- virtual `~FL_Group` ()
The destructor also deletes all the children.

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
Activates the widget.
- unsigned int `active` () const
Returns whether the widget is active.
- int `active_r` () const
Returns whether the widget and all of its parents are active.
- `FL_Align` `align` () const

- Gets the label alignment.*
- void [align](#) ([FI_Align](#) alignment)
- Sets the label alignment.*
- long [argument](#) () const
- Gets the current user data (long) argument that is passed to the callback function.*
- void [argument](#) (long v)
- Sets the current user data (long) argument that is passed to the callback function.*
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
- Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).*
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void [bind_deimage](#) (int f)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- void [bind_image](#) ([FI_Image](#) *img)
- Sets the image to use as part of the widget label when in the active state.*
- void [bind_image](#) (int f)
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- [FI_Boxtype](#) [box](#) () const
- Gets the box type of the widget.*
- void [box](#) ([FI_Boxtype](#) new_box)
- Sets the box type for the widget.*
- [FI_Callback_p](#) [callback](#) () const
- Gets the current callback function for the widget.*
- void [callback](#) ([FI_Callback](#) *cb)
- Sets the current callback function for the widget.*
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*
- void [callback](#) ([FI_Callback](#) *cb, void *p)
- Sets the current callback function and data for the widget.*
- void [callback](#) ([FI_Callback0](#) *cb)
- Sets the current callback function for the widget.*
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
- Sets the current callback function for the widget.*
- unsigned int [changed](#) () const
- Checks if the widget value changed since the last callback.*
- void [clear_active](#) ()
- Marks the widget as inactive without sending events or changing focus.*
- void [clear_changed](#) ()
- Marks the value of the widget as unchanged.*
- void [clear_damage](#) (uchar c=0)
- Clears or sets the damage flags.*
- void [clear_output](#) ()
- Sets a widget to accept input.*
- void [clear_visible](#) ()
- Hides the widget.*
- void [clear_visible_focus](#) ()
- Disables keyboard focus navigation with this widget.*

- `FL_Color color () const`
Gets the background color of the widget.
- `void color (FL_Color bg)`
Sets the background color of the widget.
- `void color (FL_Color bg, FL_Color sel)`
Sets the background and selection color of the widget.
- `FL_Color color2 () const`
For back compatibility only.
- `void color2 (unsigned a)`
For back compatibility only.
- `int contains (const FL_Widget *w) const`
Checks if w is a child of this widget.
- `void copy_label (const char *new_label)`
Sets the current label.
- `void copy_tooltip (const char *text)`
Sets the current tooltip text.
- `uchar damage () const`
Returns non-zero if `draw()` needs to be called.
- `void damage (uchar c)`
Sets the damage bits for the widget.
- `void damage (uchar c, int x, int y, int w, int h)`
Sets the damage bits for an area inside the widget.
- `int damage_resize (int, int, int, int)`
Internal use only.
- `void deactivate ()`
Deactivates the widget.
- `FL_Image * deimage ()`
Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`
Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`
Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`
Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`
Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FL_Align) const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`
Gets the widget height.
- `virtual void hide ()`
Makes a widget invisible.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`

- Set the spacing between the label and the horizontal edge of the widget.*

 - `FL_Image * image ()`

Gets the image that is used as part of the widget label when in the active state.

 - `const FL_Image * image () const`

Gets the image that is used as part of the widget label when in the active state.

 - `void image (FL_Image &img)`

Sets the image to use as part of the widget label when in the active state.

 - `void image (FL_Image *img)`

Sets the image to use as part of the widget label when in the active state.

 - `int image_bound () const`

Returns whether the image is managed by the widget.

 - `int inside (const FL_Widget *wgt) const`

Checks if this widget is a child of wgt.

 - `int is_label_copied () const`

Returns whether the current label was assigned with `copy_label()`.

 - `const char * label () const`

Gets the current label text.

 - `void label (const char *text)`

Sets the current label pointer.

 - `void label (FL_Labeltype a, const char *b)`

Shortcut to set the label text and type in one call.

 - `int label_image_spacing ()`

Return the gap size between the label and the image.

 - `void label_image_spacing (int gap)`

Set the gap between the label and the image in pixels.

 - `FL_Color labelcolor () const`

Gets the label color.

 - `void labelcolor (FL_Color c)`

Sets the label color.

 - `FL_Font labelfont () const`

Gets the font to use.

 - `void labelfont (FL_Font f)`

Sets the font to use.

 - `FL_Fonsize labelsize () const`

Gets the font size in pixels.

 - `void labelsize (FL_Fonsize pix)`

Sets the font size in pixels.

 - `FL_Labeltype labeltype () const`

Gets the label type.

 - `void labeltype (FL_Labeltype a)`

Sets the label type.

 - `void measure_label (int &ww, int &hh) const`

Sets width ww and height hh accordingly with the label size.

 - `bool needs_keyboard () const`

Returns whether this widget needs a keyboard.

 - `void needs_keyboard (bool needs)`

Sets whether this widget needs a keyboard.

 - `unsigned int output () const`

Returns if a widget is used for output only.

 - `FL_Group * parent () const`

Returns a pointer to the parent widget.

- void [parent](#) ([Fl_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [Fl_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const

- Gets the user data for this widget.*
- void **user_data** ([FL_Callback_User_Data](#) *v, bool auto_free)
- Sets the user data for this widget.*
- void **user_data** (void *v)
- Sets the user data for this widget.*
- int [vertical_label_margin](#) ()
- Get the spacing between the label and the vertical edge of the widget.*
- void [vertical_label_margin](#) (int px)
- Set the spacing between the label and the vertical edge of the widget.*
- unsigned int [visible](#) () const
- Returns whether a widget is visible.*
- unsigned int [visible_focus](#) () const
- Checks whether this widget has a visible focus.*
- void [visible_focus](#) (int v)
- Modifies keyboard focus navigation.*
- int [visible_r](#) () const
- Returns whether a widget and all its parents are visible.*
- int [w](#) () const
- Gets the widget width.*
- [FL_When](#) [when](#) () const
- Returns the conditions under which the callback is called.*
- void [when](#) (uchar i)
- Sets the flags used to decide when a callback is called.*
- [FL_Window](#) * [window](#) () const
- Returns a pointer to the nearest parent window up the widget hierarchy.*
- int [x](#) () const
- Gets the widget position in its window.*
- int [y](#) () const
- Gets the widget position in its window.*
- virtual [~FL_Widget](#) ()
- Destroys the widget.*

Static Public Member Functions

- static [FL_Group](#) * [current](#) ()
- Returns the currently active group.*
- static void [current](#) ([FL_Group](#) *g)
- Sets the current group.*

Static Public Member Functions inherited from [FL_Widget](#)

- static void [default_callback](#) ([FL_Widget](#) *widget, void *data)
- The default callback for all widgets that don't set a callback.*
- static unsigned int [label_shortcut](#) (const char *t)
- Returns the Unicode value of the '&x' shortcut in a given text.*
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
- Returns true if the given text t contains the entered '&x' shortcut.*

Protected Member Functions

- [FI_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void [draw_child](#) ([FI_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [FI_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) ([FI_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int [on_move](#) (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void [on_remove](#) (int)
Allow derived groups to act when a child widget is removed from the group.
- int * [sizes](#) ()
Returns the internal array of widget sizes and positions.
- void [update_child](#) ([FI_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FI_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.
- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int [flags](#) () const
Gets the widget flags mask.
- void [h](#) (int v)
Internal use only.
- void [set_flag](#) (unsigned int c)

- *Sets a flag in the flags mask.*
- void [w](#) (int v)
Internal use only.
- void [x](#) (int v)
Internal use only.
- void [y](#) (int v)
Internal use only.

Additional Inherited Members

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.57.1 Detailed Description

The [Fl_Group](#) class is the main FLTK container widget.

It maintains an ordered list of child widgets. These children can themselves be any widget including [Fl_Group](#). The most important subclass of [Fl_Group](#) is [Fl_Window](#), however groups can also be used to control radio buttons, to enforce resize behavior, or to manage layout of widgets, see [Fl_Pack](#), [Fl_Flex](#), [Fl_Grid](#).

The tab and arrow keys are used to move the focus between widgets of this group, and to other groups. The only modifier grabbed is shift (for shift-tab), so that ctrl-tab, alt-up, and such are free for the app to use as shortcuts.

To remove a widget from the group and destroy it, in 1.3.x (and up) you can simply use:

```
delete some_widget;
```

..and this will trigger proper scheduling of the widget's removal from its parent group.

If used as a child of [Fl_Tabs](#), setting `when (FL_WHEN_CLOSED)` will enable the Close button in the corresponding tab. If the user clicks the Close button, the callback of this group will be called with the callback reason `FL_REASON_CLOSED`.

11.57.2 Constructor & Destructor Documentation

11.57.2.1 [Fl_Group](#)()

```
Fl_Group::Fl_Group (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Group](#) widget using the given position, size, and label string.

The default boxtype is `FL_NO_BOX`.

11.57.2.2 ~Fl_Group()

```
Fl_Group::~~Fl_Group () [virtual]
```

The destructor *also deletes all the children*.

This allows a whole tree to be deleted at once, without having to keep a pointer to all the children in the user code. It is allowed that the [Fl_Group](#) and all of its children are automatic (local) variables, but you must declare the [Fl_Group](#) *first*, so that it is destroyed last.

If you add static or automatic (local) variables to an [Fl_Group](#), then it is your responsibility to remove (or delete) all such static or automatic child widgets *before destroying* the group - otherwise the group will attempt to call delete operator on them leading to undefined behavior!

11.57.3 Member Function Documentation

11.57.3.1 array()

```
Fl_Widget *const * Fl_Group::array () const
```

Returns a pointer to the internal array of children.

Note

This pointer is only valid until the next time a child is added or removed.

11.57.3.2 as_group() [1/2]

```
Fl_Group const * Fl_Group::as_group () const [inline], [virtual]
```

Reimplemented from [Fl_Widget](#).

11.57.3.3 as_group() [2/2]

```
Fl_Group * Fl_Group::as_group () [inline], [virtual]
```

Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).

Use this method if you have a widget (pointer) and need to know whether this widget is derived from [Fl_Group](#). If it returns non-NULL, then the widget in question is derived from [Fl_Group](#), and you can use the returned pointer to access its children or other [Fl_Group](#)-specific methods.

Example:

```
void my_callback (Fl_Widget *w, void *) {
    Fl_Group *g = w->as_group();
    if (g)
        printf ("This group has %d children\n", g->children());
    else
        printf ("This widget is not a group!\n");
}
```

Return values

NULL	if this widget is not derived from Fl_Group .

Note

This method is provided to avoid `dynamic_cast`.

See also

[Fl_Widget::as_window\(\)](#), [Fl_Widget::as_gl_window\(\)](#)

Reimplemented from [Fl_Widget](#).

11.57.3.4 begin()

```
void Fl_Group::begin ()
```

Sets the current group so you can build the widget tree by just constructing the widgets.

[begin\(\)](#) is automatically called by the constructor for [Fl_Group](#) (and thus for [Fl_Window](#) as well). [begin\(\)](#) is *exactly the same as* [current\(this\)](#). *Don't forget to [end\(\)](#) the group or window!*

11.57.3.5 bounds()

```
Fl_Rect * Fl_Group::bounds () [protected]
```

Returns the internal array of widget sizes and positions.

If the `bounds()` array does not exist, it will be allocated and filled with the current widget sizes and positions.

The `bounds()` array stores the initial positions of widgets as `Fl_Rect`'s. The size of the array is `children() + 2`.

- The first `Fl_Rect` is the group,
- the second is the resizable (clipped to the group),
- the rest are the children.

This is a convenient order for the resize algorithm.

If the group and/or the `resizable()` is a `Fl_Window` (or subclass) then the `x()` and `y()` coordinates of their respective `Fl_Rect`'s are zero.

Note

You should never need to use this *protected* method directly, unless you have special needs to rearrange the children of a `Fl_Group`. `Fl_Tile` uses this to rearrange its widget positions. The returned array should be considered read-only. Do not change its contents. If you need to rearrange children in a group, do so by resizing the children and call `init_sizes()`.

`#include <FL/Fl_Rect.H>` if you want to access the `bounds()` array in your derived class. `Fl_Rect.H` is intentionally not included by `Fl_Group.H` to avoid unnecessary dependencies.

Returns

Array of `Fl_Rect`'s with widget positions and sizes. The returned array is only valid until `init_sizes()` is called or widgets are added to or removed from the group.

See also

[init_sizes\(\)](#)

Since

FLTK 1.4.0

11.57.3.6 child()

```
Fl_Widget * Fl_Group::child (
    int n) const [inline]
```

Returns the n'th child.

Returns `nullptr` if `n` is out of range (since FLTK 1.4.0).

No range checking was done in FLTK 1.3 and older versions!

Parameters

in	<i>n</i>	index of child (0 .. children() - 1)

Returns

pointer to the n'th child or `nullptr` if out of range

11.57.3.7 clear()

```
void Fl_Group::clear (
    void )
```

Deletes all child widgets from memory recursively.

This method differs from the `remove()` method in that it affects all child widgets and deletes them from memory.

The `resizable()` widget of the `Fl_Group` is set to the `Fl_Group` itself.

11.57.3.8 clip_children() [1/2]

```
unsigned int Fl_Group::clip_children () [inline]
```

Returns the current clipping mode.

Returns

true, if clipping is enabled, false otherwise.

See also

void [Fl_Group::clip_children\(int c\)](#)

11.57.3.9 clip_children() [2/2]

```
void Fl_Group::clip_children (
    int c) [inline]
```

Controls whether the group widget clips the drawing of child widgets to its bounding box.

Set *c* to 1 if you want to clip the child widgets to the bounding box.

The default is to not clip (0) the drawing of child widgets.

11.57.3.10 current() [1/2]

```
Fl_Group * Fl_Group::current () [static]
```

Returns the currently active group.

The [Fl_Widget](#) constructor automatically does [current\(\)](#)->[add\(widget\)](#) if this is not null. To prevent new widgets from being added to a group, call [Fl_Group::current\(0\)](#).

11.57.3.11 current() [2/2]

```
void Fl_Group::current (
    Fl_Group * g) [static]
```

Sets the current group.

See also

[Fl_Group::current\(\)](#)

11.57.3.12 delete_child()

```
int Fl_Group::delete_child (
    int index) [virtual]
```

Removes the widget at *index* from the group and deletes it.

This method does nothing if *index* is out of bounds.

This method differs from the [remove\(\)](#) method in that it deletes the widget from memory. Since this method is virtual it can be reimplemented in subclasses with additional requirements and consequences. See the documentation of subclasses.

Many subclasses don't need to reimplement this method.

Note

This method **may** refuse to remove and delete the widget if it is an essential part of the [Fl_Group](#), for instance a scrollbar in an [Fl_Scroll](#) group. In this case the widget is neither removed nor deleted.

This method does not call [init_sizes\(\)](#) or [redraw\(\)](#). This is left to user code if necessary.

Returns 0 if the widget was removed and deleted. Return values > 0 are reserved for use by FLTK core widgets.

Return values < 0 are free to be used by user defined widgets.

Todo Reimplementation of [Fl_Group::delete_child\(int\)](#) in more FLTK subclasses. This is not yet complete.

Parameters

in	<i>index</i>	index of child to be removed

Returns

success (0) or error code

Return values

0	success
1	index out of range
2	widget not allowed to be removed (see note)
>2	reserved for FLTK use

Since

FLTK 1.4.0

Reimplemented in [Fl_Scroll](#).

11.57.3.13 draw()

```
void Fl_Group::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

Reimplemented in [Fl_Help_View](#), [Fl_Input_Choice](#), [Fl_Pack](#), [Fl_Scroll](#), [Fl_Spinner](#), [Fl_Table](#), [Fl_Tabs](#), [Fl_Terminal](#), [Fl_Text_Display](#), [Fl_Tree](#), [Fl_Window](#), and [Fl_Wizard](#).

11.57.3.14 draw_child()

```
void Fl_Group::draw_child (
    Fl_Widget & widget) const [protected]
```

Forces a child to redraw.

This draws a child widget, if it is not clipped. The damage bits are cleared after drawing.

11.57.3.15 draw_children()

```
void Fl_Group::draw_children () [protected]
```

Draws all children of the group.

This is useful, if you derived a widget from [Fl_Group](#) and want to draw a special border or background. You can call [draw_children\(\)](#) from the derived [draw\(\)](#) method after drawing the box, border, or background.

11.57.3.16 end()

```
void Fl_Group::end ()
```

Exactly the same as [current\(this->parent\(\)\)](#).

Any new widgets added to the widget tree will be added to the parent of the group.

11.57.3.17 find()

```
int Fl_Group::find (
    const Fl_Widget * o) const
```

Searches the children for the widget and returns the index.
Returns [children\(\)](#) if the widget is NULL or not found.

11.57.3.18 focus()

```
void Fl_Group::focus (
    Fl_Widget * W) [inline]
```

Deprecated This is for backwards compatibility only. You should use *W->take_focus()* instead.

See also

[Fl_Widget::take_focus\(\)](#);

11.57.3.19 handle()

```
int Fl_Group::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Help_View](#), [Fl_Scroll](#), [Fl_Spinner](#), [Fl_Table](#), [Fl_Table_Row](#), [Fl_Tabs](#), [Fl_Terminal](#), [Fl_Text_Display](#), [Fl_Text_Editor](#), [Fl_Tile](#), [Fl_Tree](#), and [Fl_Window](#).

11.57.3.20 init_sizes()

```
void Fl_Group::init_sizes ()
```

Resets the internal array of widget sizes and positions.

The [Fl_Group](#) widget keeps track of the original widget sizes and positions when resizing occurs so that if you resize a window back to its original size the widgets will be in the correct places. If you rearrange the widgets in your group, call this method to register the new arrangement with the [Fl_Group](#) that contains them.

If you add or remove widgets, this will be done automatically.

Note

The internal array of widget sizes and positions will be allocated and filled when the next `resize()` occurs. For more information on the contents and structure of the `bounds()` array see `bounds()`.

See also

`bounds()`

`sizes()` (deprecated)

11.57.3.21 insert() [1/2]

```
void Fl_Group::insert (
    Fl_Widget & o,
    int index)
```

The widget is removed from its current group (if any) and then inserted into this group.

It is put at index `n` - or at the end, if `n >= children()`. This can also be used to rearrange the widgets inside a group.

11.57.3.22 insert() [2/2]

```
void Fl_Group::insert (
    Fl_Widget & o,
    Fl_Widget * before) [inline]
```

This does `insert(w, find(before))`.

This will append the widget if `before` is not in the group.

11.57.3.23 on_insert()

```
int Fl_Group::on_insert (
    Fl_Widget * candidate,
    int index) [protected], [virtual]
```

Allow derived groups to act when a widget is added as a child.

Widgets derived from `Fl_Group` may store additional data for their children. Overriding this method will allow derived classes to generate these data structures just before the child is added.

This method usually returns the same index that was given in the parameters. By setting a new index, the position of other widgets in the list of children can be preserved (e.g. `Fl_Scroll` keeps its scroll bars as the last two children). By returning -1, `Fl_Group::insert` will not add the child to the group. This is not recommended, but `Fl_Table` does something similar to forward children to a hidden group.

Parameters

<i>candidate</i>	the candidate will be added to the child vector after this method returns.
<i>index</i>	add the child at this position in the list of children

Returns

index to position the child as planned

a new index to force the child to a different position

-1 to keep the group from adding the candidate

Reimplemented in `Fl_Scroll`, `Fl_Tabs`, and `Fl_Tile`.

11.57.3.24 on_move()

```
int Fl_Group::on_move (
    int oldIndex,
    int newIndex) [protected], [virtual]
```

Allow derived groups to act when a widget is moved within the group.

Widgets derived from [Fl_Group](#) may store additional data for their children. Overriding this method will allow derived classes to move these data structures just before the child itself is moved.

This method usually returns the new index that was given in the parameters. By setting a different destination index, the position of other widgets in the list of children can be preserved.

By returning -1, [Fl_Group::insert](#) will not move the child.

Parameters

<i>oldIndex</i>	the current index of the child that will be moved
<i>newIndex</i>	the new index of the child

Returns

newIndex to position the child as planned

a different index to force the child to a different position

-1 to keep the group from moving the child

Reimplemented in [Fl_Scroll](#), [Fl_Tabs](#), and [Fl_Tile](#).

11.57.3.25 on_remove()

```
void Fl_Group::on_remove (
    int index) [protected], [virtual]
```

Allow derived groups to act when a child widget is removed from the group.

Widgets derived from [Fl_Group](#) may store additional data for their children. Overriding this method will allow derived classes to remove these data structures just before the child is removed.

Parameters

<i>index</i>	remove the child at this position

Reimplemented in [Fl_Flex](#), [Fl_Grid](#), [Fl_Tabs](#), and [Fl_Tile](#).

11.57.3.26 remove() [1/3]

```
void Fl_Group::remove (
    Fl_Widget & o)
```

Removes a widget from the group but does not delete it.

This method does nothing if the widget is not a child of the group.

This method differs from the [clear\(\)](#) method in that it only affects a single widget and does not delete it from memory.

Note

If you have the child's index anyway, use [remove\(int index\)](#) instead, because this doesn't need a child lookup in the group's table of children. This can be much faster, if there are lots of children.

11.57.3.27 remove() [2/3]

```
void Fl_Group::remove (
    Fl_Widget * o) [inline]
```

Removes the widget *o* from the group.

See also

void [remove\(Fl_Widget&\)](#)

11.57.3.28 remove() [3/3]

```
void Fl_Group::remove (
    int index)
```

Removes the widget at `index` from the group but does not delete it.

This method does nothing if `index` is out of bounds.

This method differs from the [clear\(\)](#) method in that it only affects a single widget and does not delete it from memory.

Since

FLTK 1.3.0

11.57.3.29 resizable() [1/3]

```
Fl_Widget * Fl_Group::resizable () const [inline]
```

Returns the group's resizable widget.

See void [Fl_Group::resizable\(Fl_Widget *o\)](#)

11.57.3.30 resizable() [2/3]

```
void Fl_Group::resizable (
    Fl_Widget & o) [inline]
```

Sets the group's resizable widget.

See void [Fl_Group::resizable\(Fl_Widget *o\)](#)

11.57.3.31 resizable() [3/3]

```
void Fl_Group::resizable (
    Fl_Widget * o) [inline]
```

The resizable widget defines both the resizing box and the resizing behavior of the group and its children.

If the resizable is NULL the group's size is fixed and all of the widgets in the group remain a fixed size and distance from the top-left corner. This is the default for groups derived from [Fl_Window](#) and [Fl_Pack](#).

The resizable may be set to the group itself, in which case all of the widgets that are its direct children are resized proportionally. This is the default value for [Fl_Group](#).

The resizable widget defines the resizing box for the group, which could be the group itself or one of the group's direct children. When the group is resized it calculates a new size and position for all of its children. Widgets that are horizontally or vertically inside the dimensions of the box are scaled to the new size. Widgets outside the box are moved.

Note

The resizable of a group **must** be one of

- NULL
- the group itself
- a direct child of the group.

If you set any other widget that is not a direct child of the group as its resizable then the behavior is undefined. This is **not** checked by [Fl_Group](#) for historical reasons.

In these examples the gray area is the resizable:

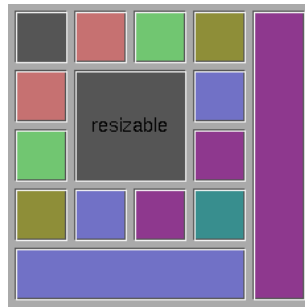


Figure 11.21 before resize



Figure 11.22 after resize

It is possible to achieve any type of resize behavior by using an invisible [Fl_Box](#) as the resizable and/or by using a hierarchy of [Fl_Group](#) widgets, each with their own resizing strategies.

See the [How Does Resizing Work?](#) chapter for more examples and detailed explanation.

Note

The [resizable\(\)](#) widget of a window can also affect the window's resizing behavior if [Fl_Window::size_range\(\)](#) is not called. Please see [Fl_Window::default_size_range\(\)](#) for more information on how the default size range is calculated.

See also

[Fl_Window::size_range\(\)](#)

[Fl_Window::default_size_range\(\)](#)

11.57.3.32 `resize()`

```
void Fl_Group::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Resizes the [Fl_Group](#) widget and all of its children.

The [Fl_Group](#) widget first resizes itself, and then it moves and resizes all its children according to the rules documented for [Fl_Group::resizable\(Fl_Widget*\)](#)

See also

[Fl_Group::resizable\(Fl_Widget*\)](#)

[Fl_Group::resizable\(\)](#)

[Fl_Widget::resize\(int,int,int,int\)](#)

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Help_View](#), [Fl_Input_Choice](#), [Fl_Overlay_Window](#), [Fl_Pack](#), [Fl_Scroll](#), [Fl_Spinner](#), [Fl_Table](#), [Fl_Tabs](#), [Fl_Terminal](#), [Fl_Text_Display](#), [Fl_Tile](#), [Fl_Tree](#), and [Fl_Window](#).

11.57.3.33 sizes()

```
int * Fl_Group::sizes () [protected]
```

Returns the internal array of widget sizes and positions.

For backward compatibility with FLTK versions before 1.4.

The [sizes\(\)](#) array stores the initial positions of widgets as (left, right, top, bottom) quads. The first quad is the group, the second is the resizable (clipped to the group), and the rest are the children. If the group and/or the [resizable\(\)](#) is a [Fl_Window](#), then the first (left) and third (top) entries of their respective quads (x,y) are zero.

Deprecated Deprecated since 1.4.0. Please use [bounds\(\)](#) instead.

Note

This method will be removed in a future FLTK version (1.5.0 or higher).

Returns

Array of int's with widget positions and sizes. The returned array is only valid until [init_sizes\(\)](#) is called or widgets are added to or removed from the group.

Note

Since FLTK 1.4.0 the returned array is a **read-only** and re-ordered copy of the internal [bounds\(\)](#) array. Do not change its contents. If you need to rearrange children in a group, do so by resizing the children and call [init_sizes\(\)](#).

See also

[bounds\(\)](#)

11.57.3.34 update_child()

```
void Fl_Group::update_child (
    Fl_Widget & widget) const [protected]
```

Draws a child only if it needs it.

This draws a child widget, if it is not clipped *and* if any [damage\(\)](#) bits are set. The damage bits are cleared after drawing.

See also

[Fl_Group::draw_child\(Fl_Widget& widget\) const](#)

The documentation for this class was generated from the following files:

- [Fl_Group.H](#)
- [Fl_Group.cxx](#)
- [forms_compatibility.cxx](#)

11.58 FI_Help_Block Struct Reference

Public Attributes

- [FI_Color](#) **bgcolor**
- [uchar](#) **border**
- const char * **end**
- int **h**
- int **line** [32]
- int **ol**
- int **ol_num**
- const char * **start**
- int **w**
- int **x**
- int **y**

The documentation for this struct was generated from the following file:

- [FI_Help_View.H](#)

11.59 FI_Help_Dialog Class Reference

The [FI_Help_Dialog](#) widget displays a standard help dialog window using the [FI_Help_View](#) widget.

Public Member Functions

- **FI_Help_Dialog** ()
The constructor creates the dialog pictured above.
- int **h** ()
Returns the position and size of the help dialog.
- void **hide** ()
Hides the [FI_Help_Dialog](#) window.
- int **load** (const char *f)
Loads the specified HTML file into the [FI_Help_View](#) widget.
- void **position** (int xx, int yy)
Set the screen position of the dialog.
- void **resize** (int xx, int yy, int ww, int hh)
Change the position and size of the dialog.
- void **show** ()
Shows the [FI_Help_Dialog](#) window.
- void **show** (int argc, char **argv)
*Shows the main Help Dialog Window Delegates call to encapsulated window_ void [FI_Window::show\(int argc, char **argv\)](#) instance method.*
- [FI_Fontsize](#) **textsize** ()
Sets or gets the default text size for the help view.
- void **textsize** ([FI_Fontsize](#) s)
Sets or gets the default text size for the help view.
- void **topline** (const char *n)
Sets the top line in the [FI_Help_View](#) widget to the named or numbered line.
- void **topline** (int n)
Sets the top line in the [FI_Help_View](#) widget to the named or numbered line.
- const char * **value** () const
The first form sets the current buffer to the string provided and reformats the text.
- void **value** (const char *f)

The first form sets the current buffer to the string provided and reformats the text.

- `int visible ()`
Returns 1 if the [Fl_Help_Dialog](#) window is visible.
- `int w ()`
Returns the position and size of the help dialog.
- `int x ()`
Returns the position and size of the help dialog.
- `int y ()`
Returns the position and size of the help dialog.
- `~Fl_Help_Dialog ()`
The destructor destroys the widget and frees all memory that has been allocated for the current file.

11.59.1 Detailed Description

The [Fl_Help_Dialog](#) widget displays a standard help dialog window using the [Fl_Help_View](#) widget.

The [Fl_Help_Dialog](#) class is not part of the FLTK core library, but instead of [fltk_images](#). Use `--use-images` when compiling with `fltk-config`.

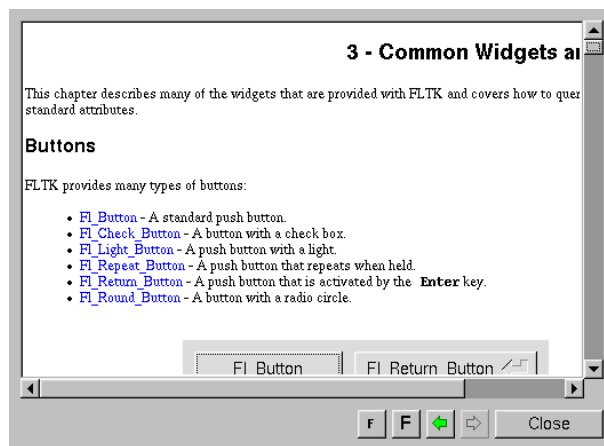


Figure 11.23 [Fl_Help_Dialog](#)

11.59.2 Member Function Documentation

11.59.2.1 load()

```
int Fl_Help_Dialog::load (
    const char * f)
```

Loads the specified HTML file into the [Fl_Help_View](#) widget.

The filename can also contain a target name ("filename.html#target"). Always use forward slashes as path delimiters, Windows-style backslashes are not supported here.

Parameters

<code>in</code>	<code>f</code>	the name and path of an HTML file

Returns

0 on success, -1 on error

See also

[Fl_Help_View::load\(\)](#), [fl_load_uri\(\)](#)

11.59.2.2 show()

```
void Fl_Help_Dialog::show ()
```

Shows the [Fl_Help_Dialog](#) window.

Shows the main Help Dialog Window Delegates call to encapsulated window_ void [Fl_Window::show\(\)](#) method.

11.59.2.3 textsize()

```
void Fl_Help_Dialog::textsize (
    Fl_Fontsize s)
```

Sets or gets the default text size for the help view.

Sets the internal [Fl_Help_View](#) instance text size.

Delegates call to encapsulated view_ void [Fl_Help_View::textsize\(Fl_Fontsize s\)](#) instance method

11.59.2.4 value() [1/2]

```
const char * Fl_Help_Dialog::value () const
```

The first form sets the current buffer to the string provided and reformats the text.

It also clears the history of the "back" and "forward" buttons. The second form returns the current buffer contents.

11.59.2.5 value() [2/2]

```
void Fl_Help_Dialog::value (
    const char * v)
```

The first form sets the current buffer to the string provided and reformats the text.

It also clears the history of the "back" and "forward" buttons. The second form returns the current buffer contents.

The documentation for this class was generated from the following files:

- [Fl_Help_Dialog.H](#)
- [Fl_Help_Dialog.cxx](#)
- [Fl_Help_Dialog_Dox.cxx](#)

11.60 Fl_Help_Font_Stack Struct Reference

Public Member Functions

- `size_t count () const`
Gets the current count of font style elements in the stack.
- `Fl_Help_Font_Stack ()`
font stack construction, initialize attributes.
- `void init (Fl_Font f, Fl_Fontsize s, Fl_Color c)`
- `void pop (Fl_Font &f, Fl_Fontsize &s, Fl_Color &c)`
Pops from the stack the font style triplet and calls [fl_font\(\)](#) & [fl_color\(\)](#) adequately.
- `void push (Fl_Font f, Fl_Fontsize s, Fl_Color c)`
Pushes the font style triplet on the stack, also calls [fl_font\(\)](#) & [fl_color\(\)](#) adequately.
- `void top (Fl_Font &f, Fl_Fontsize &s, Fl_Color &c)`
Gets the top (current) element on the stack.

Protected Attributes

- `Fl_Help_Font_Style elts_ [MAX_FL_HELP_FS_ELTS]`
font elements
- `size_t nfonts_`
current number of fonts in stack

The documentation for this struct was generated from the following file:

- [Fl_Help_View.H](#)

11.61 Fl_Help_Font_Style Struct Reference

[Fl_Help_View](#) font stack element definition.

```
#include <Fl_Help_View.H>
```

Public Member Functions

- **Fl_Help_Font_Style** ([Fl_Font](#) afont, [Fl_Fontsize](#) asize, [Fl_Color](#) acolor)
- void **get** ([Fl_Font](#) &afont, [Fl_Fontsize](#) &asize, [Fl_Color](#) &acolor)
Gets current font attributes.
- void **set** ([Fl_Font](#) afont, [Fl_Fontsize](#) asize, [Fl_Color](#) acolor)
Sets current font attributes.

Public Attributes

- [Fl_Color](#) **c**
Font Color.
- [Fl_Font](#) **f**
Font.
- [Fl_Fontsize](#) **s**
Font Size.

11.61.1 Detailed Description

[Fl_Help_View](#) font stack element definition.

The documentation for this struct was generated from the following file:

- Fl_Help_View.H

11.62 Fl_Help_Link Struct Reference

Definition of a link for the html viewer.

```
#include <Fl_Help_View.H>
```

Public Attributes

- char **filename** [192]
Reference filename.
- int **h**
Height of link text.
- char **name** [32]
Link target (blank if none)
- int **w**
Width of link text.
- int **x**
X offset of link text.
- int **y**
Y offset of link text.

11.62.1 Detailed Description

Definition of a link for the html viewer.

The documentation for this struct was generated from the following file:

- Fl_Help_View.H

11.63 `Fl_Help_Target` Struct Reference

`Fl_Help_Target` structure.

```
#include <Fl_Help_View.H>
```

Public Attributes

- char **name** [32]
Target name.
- int **y**
Y offset of target.

11.63.1 Detailed Description

`Fl_Help_Target` structure.

The documentation for this struct was generated from the following file:

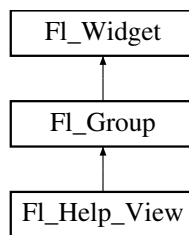
- `Fl_Help_View.H`

11.64 `Fl_Help_View` Class Reference

The `Fl_Help_View` widget displays HTML text.

```
#include <Fl_Help_View.H>
```

Inheritance diagram for `Fl_Help_View`:



Public Member Functions

- void **clear_selection** ()
Removes the current text selection.
- int **copy** (int clipboard=1)
If text is selected in this view, copy it to a clipboard.
- const char * **directory** () const
Returns the current directory for the text in the buffer.
- const char * **filename** () const
Returns the current filename for the text in the buffer.
- int **find** (const char *s, int p=0)
Finds the specified string s at starting position p.
- **Fl_Help_View** (int xx, int yy, int ww, int hh, const char *l=0)
The constructor creates the `Fl_Help_View` widget at the specified position and size.
- int **handle** (int) **FL_OVERRIDE**
Handles events in the widget.
- int **leftline** () const
Gets the left position in pixels.
- void **leftline** (int)
Scrolls the text to the indicated position, given a pixel column.
- void **link** (Fl_Help_Func *fn)

This method assigns a callback function to use when a link is followed or a file is loaded (via `Fl_Help_View::load()`) that requires a different file or path.

- `int load` (`const char *f`)
Loads the specified file.
- `void resize` (`int, int, int, int`) `FL_OVERRIDE`
Resizes the help widget.
- `int scrollbar_size` () `const`
Gets the current size of the scrollbars' troughs, in pixels.
- `void scrollbar_size` (`int newSize`)
Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.
- `void select_all` ()
Selects all the text in the view.
- `int size` () `const`
Gets the size of the help view.
- `void size` (`int W, int H`)
- `int text_selected` ()
Check if the user selected text in this view.
- `Fl_Color textcolor` () `const`
Returns the current default text color.
- `void textcolor` (`Fl_Color c`)
Sets the default text color.
- `Fl_Font textfont` () `const`
Returns the current default text font.
- `void textfont` (`Fl_Font f`)
Sets the default text font.
- `Fl_Fonsize textsize` () `const`
Gets the default text size.
- `void textsize` (`Fl_Fonsize s`)
Sets the default text size.
- `const char * title` ()
Returns the current document title, or NULL if there is no title.
- `int topline` () `const`
Returns the current top line in pixels.
- `void topline` (`const char *n`)
Scrolls the text to the indicated position, given a named destination.
- `void topline` (`int`)
Scrolls the text to the indicated position, given a pixel line.
- `const char * value` () `const`
Returns the current buffer contents.
- `void value` (`const char *val`)
Sets the current help text buffer to the string provided and reformats the text.
- `~Fl_Help_View` ()
Destroys the `Fl_Help_View` widget.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *[_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void [add](#) ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void [add](#) ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void [add_resizable](#) ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * [array](#) () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * [as_group](#) () const [FL_OVERRIDE](#)
- [FL_Group](#) * [as_group](#) () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void [begin](#) ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * [child](#) (int n) const
Returns the n'th child.
- int [children](#) () const
Returns how many child widgets the group has.
- void [clear](#) ()
Deletes all child widgets from memory recursively.
- unsigned int [clip_children](#) ()
Returns the current clipping mode.
- void [clip_children](#) (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int [delete_child](#) (int n)
*Removes the widget at *index* from the group and deletes it.*
- void [end](#) ()
*Exactly the same as *current(this->parent())*.*
- int [find](#) (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int [find](#) (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char *==0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void [focus](#) ([FL_Widget](#) *W)
- void [forms_end](#) ()
This is for forms compatibility only.
- void [init_sizes](#) ()
Resets the internal array of widget sizes and positions.
- void [insert](#) ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void [insert](#) ([FL_Widget](#) &o, [FL_Widget](#) *before)
*This does *insert(w, find(before))*.*
- void [remove](#) ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void [remove](#) ([FL_Widget](#) *o)
Removes the widget o from the group.
- void [remove](#) (int index)

- Removes the widget at `index` from the group but does not delete it.*

 - `FI_Widget * resizable ()` const

Returns the group's resizable widget.

 - void `resizable (FI_Widget &o)`
- Sets the group's resizable widget.*
- void `resizable (FI_Widget *o)`
- The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
- virtual `~FI_Group ()`
- The destructor also deletes all the children.*

Public Member Functions inherited from FI_Widget

- void `_clear_fullscreen ()`
 - void `_set_fullscreen ()`
 - void `activate ()`
- Activates the widget.*
- unsigned int `active ()` const
- Returns whether the widget is active.*
- int `active_r ()` const
- Returns whether the widget and all of its parents are active.*
- `FI_Align align ()` const
- Gets the label alignment.*
- void `align (FI_Align alignment)`
- Sets the label alignment.*
- long `argument ()` const
- Gets the current user data (long) argument that is passed to the callback function.*
- void `argument (long v)`
- Sets the current user data (long) argument that is passed to the callback function.*
- virtual class `FI_Gl_Window * as_gl_window ()`
- Returns an `FI_Gl_Window` pointer if this widget is an `FI_Gl_Window`.*
- virtual class `FI_Gl_Window` const * `as_gl_window ()` const
 - virtual `FI_Window * as_window ()`
- Returns an `FI_Window` pointer if this widget is an `FI_Window`.*
- virtual `FI_Window` const * `as_window ()` const
 - void `bind_deimage (FI_Image *img)`
- Sets the image to use as part of the widget label when in the inactive state.*
- void `bind_deimage (int f)`
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- void `bind_image (FI_Image *img)`
- Sets the image to use as part of the widget label when in the active state.*
- void `bind_image (int f)`
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- `FI_Boxtype box ()` const
- Gets the box type of the widget.*
- void `box (FI_Boxtype new_box)`
- Sets the box type for the widget.*
- `FI_Callback_p callback ()` const
- Gets the current callback function for the widget.*
- void `callback (FI_Callback *cb)`
- Sets the current callback function for the widget.*
- void `callback (FI_Callback *cb, FI_Callback_User_Data *p, bool auto_free)`

- Sets the current callback function and managed user data for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.

 - void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.

 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.

 - unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.

 - void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.

 - void [clear_changed](#) ()

Marks the value of the widget as unchanged.

 - void [clear_damage](#) ([uchar](#) c=0)

Clears or sets the damage flags.

 - void [clear_output](#) ()

Sets a widget to accept input.

 - void [clear_visible](#) ()

Hides the widget.

 - void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.

 - [FI_Color](#) [color](#) () const

Gets the background color of the widget.

 - void [color](#) ([FI_Color](#) bg)

Sets the background color of the widget.

 - void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)

Sets the background and selection color of the widget.

 - [FI_Color](#) [color2](#) () const

For back compatibility only.

 - void [color2](#) (unsigned a)

For back compatibility only.

 - int [contains](#) (const [FI_Widget](#) *w) const

Checks if w is a child of this widget.

 - void [copy_label](#) (const char *new_label)

Sets the current label.

 - void [copy_tooltip](#) (const char *text)

Sets the current tooltip text.

 - [uchar](#) [damage](#) () const

Returns non-zero if [draw\(\)](#) needs to be called.

 - void [damage](#) ([uchar](#) c)

Sets the damage bits for the widget.

 - void [damage](#) ([uchar](#) c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.

 - int [damage_resize](#) (int, int, int, int)

Internal use only.

 - void [deactivate](#) ()

Deactivates the widget.

 - [FI_Image](#) * [deimage](#) ()

Gets the image that is used as part of the widget label when in the inactive state.

 - const [FI_Image](#) * [deimage](#) () const

Gets the image that is used as part of the widget label when in the inactive state.

- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const

- Gets the font to use.*

 - void [labelfont](#) ([FI_Font](#) f)

Sets the font to use.
- [FI_Fontsize](#) [labelsize](#) () const

Gets the font size in pixels.
- void [labelsize](#) ([FI_Fontsize](#) pix)

Sets the font size in pixels.
- [FI_Labeltype](#) [labeltype](#) () const

Gets the label type.
- void [labeltype](#) ([FI_Labeltype](#) a)

Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FI_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FI_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- [FI_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.

- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- uchar [type](#) () const
Gets the widget type.
- void [type](#) (uchar t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) (uchar i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual [~Fl_Widget](#) ()
Destroys the widget.

Static Public Attributes

- static const char * **copy_menu_text** = "Copy"
[this text may be customized at run-time]

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
Draws the [FL_Help_View](#) widget.

Protected Member Functions inherited from [FI_Group](#)

- [FI_Rect](#) * **bounds** ()
Returns the internal array of widget sizes and positions.
- void **draw_child** ([FI_Widget](#) &widget) const
Forces a child to redraw.
- void **draw_children** ()
Draws all children of the group.
- void **draw_outside_label** (const [FI_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int **on_insert** ([FI_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int **on_move** (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void **on_remove** (int)
Allow derived groups to act when a child widget is removed from the group.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** ([FI_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If [FL_ALIGN_IMAGE_BACKDROP](#) is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.

- `FI_Widget` (int `x`, int `y`, int `w`, int `h`, const char *`label`=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int `v`)
Internal use only.
- void `set_flag` (unsigned int `c`)
Sets a flag in the flags mask.
- void `w` (int `v`)
Internal use only.
- void `x` (int `v`)
Internal use only.
- void `y` (int `v`)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from `FI_Group`

- static `FI_Group` * `current` ()
Returns the currently active group.
- static void `current` (`FI_Group` *`g`)
Sets the current group.

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *`widget`, void *`data`)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *`t`)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *`s`, const bool `require_alt`=false)
Returns true if the given text `t` contains the entered '&x' shortcut.

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

11.64.1 Detailed Description

The [Fl_Help_View](#) widget displays HTML text.

Most HTML 2.0 elements are supported, as well as a primitive implementation of tables. GIF, JPEG, and PNG images are displayed inline.

Supported HTML tags:

- A: HREF/NAME
- B
- BODY: BGCOLOR/TEXT/LINK
- BR
- CENTER
- CODE
- DD
- DL
- DT
- EM
- FONT: COLOR/SIZE/FACE=(helvetica/arial/sans/times/serif/symbol/courier)
- H1/H2/H3/H4/H5/H6
- HEAD
- HR
- I
- IMG: SRC/WIDTH/HEIGHT/ALT
- KBD
- LI
- OL
- P
- PRE
- STRONG
- TABLE: TH/TD/TR/BORDER/BGCOLOR/COLSPAN/ALIGN=CENTER|RIGHT|LEFT
- TITLE
- TT
- U
- UL
- VAR

Supported color names:

- black,red,green,yellow,blue,magenta,fuchsia,cyan,aqua,white,gray,gray,lime,maroon,navy,olive,purple,silver,teal.

Supported urls:

- Internal: file:

- External: http: ftp: https: ipp: mailto: news:

Quoted char names:

- Aacute aacute Acirc acirc acute AElig aelig Agrave agrave amp Aring aring Atilde atilde Auml auml
- brvbar bull
- Ccedil ccedil cedil cent copy curren
- dagger deg divide
- Eacute eacute Ecirc ecirc Egrave egrave ETH eth Euml euml euro
- frac12 frac14 frac34
- gt
- Iacute iacute Icirc icirc Iexcl lgrave lgrave lquest luml iuml
- laquo lt
- macr micro middot
- nbsp not Ntilde ntilde
- Oacute oacute Ocirc ocirc Ograve ograve ordf ordm Oslash oslash Otilde otilde Ouml ouml
- para permil plusmn pound
- quot
- raquo reg
- sect shy sup1 sup2 sup3 szlig
- THORN thorn times trade
- Uacute uacute Ucirc ucirc Ugrave ugrave uml Uuml uuml
- Yacute yacute
- yen Yuml yuml

Note

You can't effectively set the `box()` to `FL_NO_BOX`, this would result in `FL_DOWN_BOX` being used as the boxtype of the widget. This is unexpected but can't be changed for backwards compatibility. If you don't want a frame around the widget you can use `FL_FLAT_BOX` instead.

11.64.2 Constructor & Destructor Documentation

11.64.2.1 ~Fl_Help_View()

```
Fl_Help_View::~Fl_Help_View ()
```

Destroys the [Fl_Help_View](#) widget.

The destructor destroys the widget and frees all memory that has been allocated for the current document.

11.64.3 Member Function Documentation

11.64.3.1 copy()

```
int Fl_Help_View::copy (
    int clipboard = 1)
```

If text is selected in this view, copy it to a clipboard.

Parameters

in	<i>clipboard</i>	for x11 only, 0=selection buffer, 1=clipboard, 2=both

Returns

1 if text is selected, 0 if no text is selected

11.64.3.2 draw()

```
void Fl_Help_View::draw (
    void ) [protected], [virtual]
```

Draws the [Fl_Help_View](#) widget.

Reimplemented from [Fl_Group](#).

11.64.3.3 find()

```
int Fl_Help_View::find (
    const char * s,
    int p = 0)
```

Finds the specified string *s* at starting position *p*.

The argument *p* and the return value are offsets in [Fl_Help_View::value\(\)](#), counting from 0. If *p* is out of range, 0 is used.

The string comparison is simple but honors some special cases:

- the specified string *s* must be in UTF-8 encoding
- HTML tags in [value\(\)](#) are filtered (not compared as such, they never match)
- HTML entities like '<' or '&#20ac;' are converted to Unicode (UTF-8)
- ASCII characters (7-bit, < 0x80) are compared case insensitive
- every newline (LF, '\n') in [value\(\)](#) is treated like a single space
- all other strings are compared as-is (byte by byte)

Parameters

in	<i>s</i>	search string in UTF-8 encoding
in	<i>p</i>	starting position for search (0,...), Default = 0

Returns

the matching position or -1 if not found

11.64.3.4 handle()

```
int Fl_Help_View::handle (
    int event) [virtual]
```

Handles events in the widget.

Reimplemented from [Fl_Group](#).

11.64.3.5 leftline()

```
void Fl_Help_View::leftline (
    int left)
```

Scrolls the text to the indicated position, given a pixel column.

If the given pixel value *left* is out of range, then the text is scrolled to the left or right side of the document, resp.

Parameters

in	left	left column number in pixels (0 = left side)

11.64.3.6 link()

```
void Fl_Help_View::link (
    Fl_Help_Func * fn) [inline]
```

This method assigns a callback function to use when a link is followed or a file is loaded (via [Fl_Help_View::load\(\)](#)) that requires a different file or path.

The callback function receives a pointer to the [Fl_Help_View](#) widget and the URI or full pathname for the file in question. It must return a pathname that can be opened as a local file or NULL:

```
const char *fn(Fl_Widget *w, const char *uri);
```

The link function can be used to retrieve remote or virtual documents, returning a temporary file that contains the actual data. If the link function returns NULL, the value of the [Fl_Help_View](#) widget will remain unchanged.

If the link callback cannot handle the URI scheme, it should return the uri value unchanged or set the [value\(\)](#) of the widget before returning NULL.

11.64.3.7 load()

```
int Fl_Help_View::load (
    const char * f)
```

Loads the specified file.

This method loads the specified file or URL. The filename may end in a `#name` style target.

If the URL starts with *ftp*, *http*, *https*, *ipp*, *mailto*, or *news*, followed by a colon, FLTK will use [fl_open_uri\(\)](#) to show the requested page in an external browser.

In all other cases, the URL is interpreted as a filename. The file is read and displayed in this browser. Note that Windows style backslashes are not supported in the file name.

Parameters

in	f	filename or URL

Returns

0 on success, -1 on error

See also

[fl_open_uri\(\)](#)

11.64.3.8 resize()

```
void Fl_Help_View::resize (
    int xx,
    int yy,
    int ww,
    int hh) [virtual]
```

Resizes the help widget.

Reimplemented from [Fl_Group](#).

11.64.3.9 scrollbar_size() [1/2]

```
int Fl_Help_View::scrollbar_size () const [inline]
```

Gets the current size of the scrollbars' troughs, in pixels.

If this value is zero (default), this widget will use the [Fl::scrollbar_size\(\)](#) value as the scrollbar's width.

Returns

Scrollbar size in pixels, or 0 if the global [Fl::scrollbar_size\(\)](#) is being used.

See also

[Fl::scrollbar_size\(int\)](#)

11.64.3.10 scrollbar_size() [2/2]

```
void Fl_Help_View::scrollbar_size (
    int newSize) [inline]
```

Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.

Normally you should not need this method, and should use [Fl::scrollbar_size\(int\)](#) instead to manage the size of ALL your widgets' scrollbars. This ensures your application has a consistent UI, is the default behavior, and is normally what you want.

Only use THIS method if you really need to override the global scrollbar size. The need for this should be rare.

Setting `newSize` to the special value of 0 causes the widget to track the global [Fl::scrollbar_size\(\)](#), which is the default.

Parameters

in	<i>newSize</i>	Sets the scrollbar size in pixels. If 0 (default), scrollbar size tracks the global Fl::scrollbar_size()

See also

[Fl::scrollbar_size\(\)](#)

11.64.3.11 text_selected()

```
int Fl_Help_View::text_selected ()
```

Check if the user selected text in this view.

Returns

1 if text is selected, 0 if no text is selected

11.64.3.12 topline() [1/2]

```
void Fl_Help_View::topline (
    const char * n)
```

Scrolls the text to the indicated position, given a named destination.

Parameters

in	<i>n</i>	target name

11.64.3.13 topline() [2/2]

```
void Fl_Help_View::topline (
    int top)
```

Scrolls the text to the indicated position, given a pixel line.

If the given pixel value `top` is out of range, then the text is scrolled to the top or bottom of the document, resp.

Parameters

in	<i>top</i>	top line number in pixels (0 = start of document)

11.64.3.14 value()

```
void Fl_Help_View::value (
    const char * val)
```

Sets the current help text buffer to the string provided and reformats the text.

The provided character string `val` is copied internally and will be freed when `value()` is called again, or when the widget is destroyed.

If `val` is NULL, then the widget is cleared.

The documentation for this class was generated from the following files:

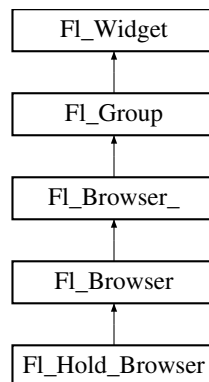
- Fl_Help_View.H
- Fl_Help_View.cxx

11.65 Fl_Hold_Browser Class Reference

The `Fl_Hold_Browser` is a subclass of `Fl_Browser` which lets the user select a single item, or no items by clicking on the empty space.

```
#include <Fl_Hold_Browser.H>
```

Inheritance diagram for `Fl_Hold_Browser`:



Public Member Functions

- `Fl_Hold_Browser` (int X, int Y, int W, int H, const char *L=0)
Creates a new `Fl_Hold_Browser` widget using the given position, size, and label string.

Public Member Functions inherited from `Fl_Browser`

- void `add` (const char *newtext, void *d=0)
Adds a new line to the end of the browser.
- void `bottomline` (int line)
Scrolls the browser so the bottom item in the browser is showing the specified `line`.
- void `clear` ()
Removes all the lines in the browser.
- char `column_char` () const
Gets the current column separator character.
- void `column_char` (char c)
Sets the column separator to `c`.
- const int * `column_widths` () const
Gets the current column width array.
- void `column_widths` (const int *arr)
Sets the current array to `arr`.
- void * `data` (int line) const

- Returns the user [data\(\)](#) for specified `line`.*

 - void [data](#) (int line, void *d)

Sets the user data for specified `line` to `d`.
- void [display](#) (int line, int val=1)

For back compatibility.
- int [displayed](#) (int line) const

Returns non-zero if `line` has been scrolled to a position where it is being displayed.
- [FL_Browser](#) (int X, int Y, int W, int H, const char *L=0)

The constructor makes an empty browser.
- char [format_char](#) () const

Gets the current format code prefix character, which by default is '@'.
- void [format_char](#) (char c)

Sets the current format code prefix character to `c`.
- void [hide](#) () [FL_OVERRIDE](#)

Hides the entire [FL_Browser](#) widget – opposite of [show\(\)](#).
- void [hide](#) (int line)

Makes `line` invisible, preventing selection by the user.
- [FL_Image](#) * [icon](#) (int line) const

Returns the icon currently defined for `line`.
- void [icon](#) (int line, [FL_Image](#) *icon)

Set the image icon for `line` to the value `icon`.
- void [insert](#) (int line, const char *newtext, void *d=0)

Insert a new entry whose label is `newtext` above given `line`, optional data `d`.
- void [lineposition](#) (int line, [FL_Line_Position](#) pos)

Updates the browser so that `line` is shown at position `pos`.
- int [load](#) (const char *filename)

Clears the browser and reads the file, adding each line from the file to the browser.
- void [make_visible](#) (int line)

Make the item at the specified `line` [visible\(\)](#).
- void [middleline](#) (int line)

Scrolls the browser so the middle item in the browser is showing the specified `line`.
- void [move](#) (int to, int from)

Line `from` is removed and reinserted at `to`.
- void [remove](#) (int line)

Remove entry for given `line` number, making the browser one line shorter.
- void [remove_icon](#) (int line)

Removes the icon for `line`.
- void [replace](#) (int a, const char *b)

For back compatibility only.
- int [select](#) (int line, int val=1)

Sets the selection state of the item at `line` to the value `val`.
- int [selected](#) (int line) const

Returns 1 if specified `line` is selected, 0 if not.
- void [show](#) () [FL_OVERRIDE](#)

Shows the entire [FL_Browser](#) widget – opposite of [hide\(\)](#).
- void [show](#) (int line)

Makes `line` visible, and available for selection by user.
- int [size](#) () const

Returns how many lines are in the browser.
- void [size](#) (int W, int H)
- void [swap](#) (int a, int b)

- Swaps two browser lines `a` and `b`.*
- `const char * text` (int line) const
 - Returns the label text for the specified `line`.*
- `void text` (int line, const char *newtext)
 - Sets the text for the specified `line` to `newtext`.*
- `FI_Fontsize textsize` () const
 - Gets the default text size (in pixels) for the lines in the browser.*
- `void textsize` (FI_Fontsize newSize)
 - Sets the default text size (in pixels) for the lines in the browser to `newSize`.*
- `int topline` () const
 - Returns the line that is currently visible at the top of the browser.*
- `void topline` (int line)
 - Scrolls the browser so the top item in the browser is showing the specified `line`.*
- `int value` () const
 - Returns the line number of the currently selected line, or 0 if none selected.*
- `void value` (int line)
 - Sets the browser's `value()`, which selects the specified `line`.*
- `int visible` (int line) const
 - Returns non-zero if the specified `line` is visible, 0 if hidden.*
- `~FI_Browser` ()
 - The destructor deletes all list items and destroys the browser.*

Public Member Functions inherited from FI_Browser_

- `int deselect` (int docallbacks=0)
 - Deselects all items in the list and returns 1 if the state changed or 0 if it did not.*
- `void display` (void *item)
 - Displays the `item`, scrolling the list as necessary.*
- `int handle` (int event) FL_OVERRIDE
 - Handles the `event` within the normal widget bounding box.*
- `uchar has_scrollbar` () const
 - Returns the current scrollbar mode, see `FI_Browser_::has_scrollbar(uchar)`*
- `void has_scrollbar` (uchar mode)
 - Sets whether the widget should have scrollbars or not (default `FI_Browser_::BOTH`).*
- `int hposition` () const
 - Gets the horizontal scroll position of the list as a pixel position `pos`.*
- `void hposition` (int)
 - Sets the horizontal scroll position of the list to pixel position `pos`.*
- `int linespacing` () const
 - Return the height of additional spacing between browser lines.*
- `void linespacing` (int pixels)
 - Add some space between browser lines.*
- `int position` () const
- `void position` (int pos)
- `void position` (int x, int y)
- `void resize` (int X, int Y, int W, int H) FL_OVERRIDE
 - Repositions and/or resizes the browser.*
- `void scrollbar_left` ()
 - Moves the vertical scrollbar to the lefthand side of the list.*
- `void scrollbar_right` ()
 - Moves the vertical scrollbar to the righthand side of the list.*

- `int scrollbar_size () const`
Gets the current size of the scrollbars' troughs, in pixels.
- `void scrollbar_size (int newSize)`
Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.
- `int scrollbar_width () const`
Returns the global value `Fl::scrollbar_size()`.
- `void scrollbar_width (int width)`
Sets the global `Fl::scrollbar_size()`, and forces this instance of the widget to use it.
- `int select (void *item, int val=1, int docallbacks=0)`
Sets the selection state of `item` to `val`, and returns 1 if the state changed or 0 if it did not.
- `int select_only (void *item, int docallbacks=0)`
Selects `item` and returns 1 if the state changed or 0 if it did not.
- `void sort (int flags=0)`
Sort the items in the browser based on `flags`.
- `Fl_Color textcolor () const`
Gets the default text color for the lines in the browser.
- `void textcolor (Fl_Color col)`
Sets the default text color for the lines in the browser to color `col`.
- `Fl_Font textfont () const`
Gets the default text font for the lines in the browser.
- `void textfont (Fl_Font font)`
Sets the default text font for the lines in the browser to `font`.
- `Fl_Fontsize textsize () const`
Gets the default text size (in pixels) for the lines in the browser.
- `void textsize (Fl_Fontsize newSize)`
Sets the default text size (in pixels) for the lines in the browser to `size`.
- `int vposition () const`
Gets the vertical scroll position of the list as a pixel position `pos`.
- `void vposition (int pos)`
Sets the vertical scroll position of the list to pixel position `pos`.

Public Member Functions inherited from `Fl_Group`

- `Fl_Widget *&_ddfdesign_kludge ()`
This is for forms compatibility only.
- `void add (Fl_Widget &)`
The widget is removed from its current group (if any) and then added to the end of this group.
- `void add (Fl_Widget *o)`
See void `Fl_Group::add(Fl_Widget &w)`
- `void add_resizable (Fl_Widget &o)`
Adds a widget to the group and makes it the resizable widget.
- `Fl_Widget *const * array () const`
Returns a pointer to the internal array of children.
- `Fl_Group const * as_group () const FL_OVERRIDE`
- `Fl_Group * as_group () FL_OVERRIDE`
Returns an `Fl_Group` pointer if this widget is an `Fl_Group`.
- `void begin ()`
Sets the current group so you can build the widget tree by just constructing the widgets.
- `Fl_Widget * child (int n) const`
Returns the `n`'th child.
- `int children () const`

- Returns how many child widgets the group has.*

 - void `clear` ()

Deletes all child widgets from memory recursively.
- unsigned int `clip_children` ()

Returns the current clipping mode.
- void `clip_children` (int c)

Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int `delete_child` (int n)

Removes the widget at `index` from the group and deletes it.
- void `end` ()

Exactly the same as `current(this->parent())`.
- int `find` (const `FL_Widget` &o) const

See int `FL_Group::find(const FL_Widget *w)` const.
- int `find` (const `FL_Widget` *) const

Searches the children for the widget and returns the index.
- `FL_Group` (int, int, int, int, const char *==0)

Creates a new `FL_Group` widget using the given position, size, and label string.
- void `focus` (`FL_Widget` *W)
- void `forms_end` ()

This is for forms compatibility only.
- int `handle` (int) `FL_OVERRIDE`

Handles the specified event.
- void `init_sizes` ()

Resets the internal array of widget sizes and positions.
- void `insert` (`FL_Widget` &, int i)

The widget is removed from its current group (if any) and then inserted into this group.
- void `insert` (`FL_Widget` &o, `FL_Widget` *before)

This does `insert(w, find(before))`.
- void `remove` (`FL_Widget` &)

Removes a widget from the group but does not delete it.
- void `remove` (`FL_Widget` *o)

Removes the widget o from the group.
- void `remove` (int index)

Removes the widget at `index` from the group but does not delete it.
- `FL_Widget` * `resizable` () const

Returns the group's resizable widget.
- void `resizable` (`FL_Widget` &o)

Sets the group's resizable widget.
- void `resizable` (`FL_Widget` *o)

The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- void `resize` (int, int, int, int) `FL_OVERRIDE`

Resizes the `FL_Group` widget and all of its children.
- virtual `~FL_Group` ()

The destructor also deletes all the children.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

- Marks the widget as inactive without sending events or changing focus.*

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*

 - void `clear_damage` (uchar c=0)
- Clears or sets the damage flags.*

 - void `clear_output` ()
- Sets a widget to accept input.*

 - void `clear_visible` ()
- Hides the widget.*

 - void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*

 - `FL_Color` `color` () const
- Gets the background color of the widget.*

 - void `color` (FL_Color bg)
- Sets the background color of the widget.*

 - void `color` (FL_Color bg, FL_Color sel)
- Sets the background and selection color of the widget.*

 - `FL_Color` `color2` () const
- For back compatibility only.*

 - void `color2` (unsigned a)
- For back compatibility only.*

 - int `contains` (const FL_Widget *w) const
- Checks if w is a child of this widget.*

 - void `copy_label` (const char *new_label)
- Sets the current label.*

 - void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*

 - uchar `damage` () const
- Returns non-zero if `draw()` needs to be called.*

 - void `damage` (uchar c)
- Sets the damage bits for the widget.*

 - void `damage` (uchar c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int `damage_resize` (int, int, int, int)
- Internal use only.*

 - void `deactivate` ()
- Deactivates the widget.*

 - `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*

 - const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*

 - void `deimage` (FL_Image &img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - void `deimage` (FL_Image *img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*

 - void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)
- Calls the widget callback function with default arguments.*

 - void `do_callback` (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)
- Calls the widget callback function with arbitrary arguments.*

- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const
Gets the label type.
- void `labeltype` (`FL_Labeltype` a)
Sets the label type.
- void `measure_label` (int &ww, int &hh) const

- Sets width ww and height hh accordingly with the label size.*

 - bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `Fl_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`Fl_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- `Fl_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`Fl_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.
- void `set_visible` ()

Makes the widget visible.
- void `set_visible_focus` ()

Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- void `size` (int W, int H)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const

Gets the current tooltip text.
- void `tooltip` (const char *text)

Sets the current tooltip text.
- `Fl_Window` * `top_window` () const

Returns a pointer to the top-level window for the widget.
- `Fl_Window` * `top_window_offset` (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.

- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Additional Inherited Members

Public Types inherited from `Fl_Browser`

- `enum Fl_Line_Position { TOP , BOTTOM , MIDDLE }`
For internal use only?

Public Types inherited from `Fl_Browser_`

- `enum {`
`HORIZONTAL = 1 , VERTICAL = 2 , BOTH = 3 , ALWAYS_ON = 4 ,`
`HORIZONTAL_ALWAYS = 5 , VERTICAL_ALWAYS = 6 , BOTH_ALWAYS = 7 }`
Values for `has_scrollbar()`.

Static Public Member Functions inherited from FI_Group

- static [FI_Group * current](#) ()
Returns the currently active group.
- static void [current](#) ([FI_Group *g](#))
Sets the current group.

Static Public Member Functions inherited from FI_Widget

- static void [default_callback](#) ([FI_Widget *widget](#), void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Public Attributes inherited from FI_Browser_

- [FI_Scrollbar hscrollbar](#)
Horizontal scrollbar.
- [FI_Scrollbar scrollbar](#)
Vertical scrollbar.

Protected Types inherited from FI_Widget

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from FI_Browser

- [FL_BLINE * _remove](#) (int line)
Removes the item at the specified line.
- [FL_BLINE * find_line](#) (int line) const
Returns the item for specified line.
- int [full_height](#) () const [FL_OVERRIDE](#)
The height of the entire list of all [visible\(\)](#) items in pixels.
- int [incr_height](#) () const [FL_OVERRIDE](#)
The default 'average' item height (including inter-item spacing) in pixels.
- void [insert](#) (int line, [FL_BLINE *item](#))
Insert specified item above line.
- void * [item_at](#) (int line) const [FL_OVERRIDE](#)
Return the item at specified line.

- void [item_draw](#) (void *item, int X, int Y, int W, int H) const [FL_OVERRIDE](#)

Draws `item` at the position specified by `X Y W H`.

- void * [item_first](#) () const [FL_OVERRIDE](#)

Returns the very first item in the list.

- int [item_height](#) (void *item) const [FL_OVERRIDE](#)

Returns height of `item` in pixels.

- void * [item_last](#) () const [FL_OVERRIDE](#)

Returns the very last item in the list.

- void * [item_next](#) (void *item) const [FL_OVERRIDE](#)

Returns the next item after `item`.

- void * [item_prev](#) (void *item) const [FL_OVERRIDE](#)

Returns the previous item before `item`.

- void [item_select](#) (void *item, int val) [FL_OVERRIDE](#)

Change the selection state of `item` to the value `val`.

- int [item_selected](#) (void *item) const [FL_OVERRIDE](#)

See if `item` is selected.

- void [item_swap](#) (void *a, void *b) [FL_OVERRIDE](#)

Swap the items `a` and `b`.

- const char * [item_text](#) (void *item) const [FL_OVERRIDE](#)

Returns the label text for `item`.

- int [item_width](#) (void *item) const [FL_OVERRIDE](#)

Returns width of `item` in pixels.

- int [lineno](#) (void *item) const

Returns line number corresponding to `item`, or zero if not found.

- void [swap](#) (FL_BLINE *a, FL_BLINE *b)

Swap the two items `a` and `b`.

Protected Member Functions inherited from [Fl_Browser_](#)

- void [bbox](#) (int &X, int &Y, int &W, int &H) const

Returns the bounding box for the interior of the list's display window, inside the scrollbars.

- void [deleting](#) (void *item)

This method should be used when `item` is being deleted from the list.

- int [displayed](#) (void *item) const

Returns non-zero if `item` has been scrolled to a position where it is being displayed.

- void [draw](#) () [FL_OVERRIDE](#)

Draws the list within the normal widget bounding box.

- void * [find_item](#) (int ypos)

This method returns the item under mouse y position `ypos`.

- [Fl_Browser_](#) (int X, int Y, int W, int H, const char *L=0)

The constructor makes an empty browser.

- virtual int [full_width](#) () const

This method may be provided by the subclass to indicate the full width of the item list, in pixels.

- void [inserting](#) (void *a, void *b)

This method should be used when an item is in the process of being inserted into the list.

- virtual int [item_quick_height](#) (void *item) const

This method may be provided by the subclass to return the height of the `item`, in pixels.

- int [leftedge](#) () const

This method returns the X position of the left edge of the list area after adjusting for the scrollbar and border, if any.

- void [new_list](#) ()

This method should be called when the list data is completely replaced or cleared.

- void [redraw_line](#) (void *item)
This method should be called when the contents of `item` has changed, but not its height.
- void [redraw_lines](#) ()
This method will cause the entire list to be redrawn.
- void [replacing](#) (void *a, void *b)
This method should be used when item `a` is being replaced by item `b`.
- void * [selection](#) () const
Returns the item currently selected, or NULL if there is no selection.
- void [swapping](#) (void *a, void *b)
This method should be used when two items `a` and `b` are being swapped.
- void * [top](#) () const
Returns the item that appears at the top of the list.

Protected Member Functions inherited from [FL_Group](#)

- [FL_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void [draw_child](#) ([FL_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [FL_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) ([FL_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int [on_move](#) (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void [on_remove](#) (int)
Allow derived groups to act when a child widget is removed from the group.
- int * [sizes](#) ()
Returns the internal array of widget sizes and positions.
- void [update_child](#) ([FL_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FL_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [drawBackdrop](#) () const
If `FL_ALIGN_IMAGE_BACKDROP` is set, the image or deimage will be drawn.
- void [drawBox](#) () const
Draws the widget box according its box style.
- void [drawBox](#) ([FL_Boxtype](#) t, [FL_Color](#) c) const
Draws a box of type `t`, of color `c` at the widget's position and size.
- void [drawBox](#) ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) c) const
Draws a box of type `t`, of color `c` at the position `X,Y` and size `W,H`.
- void [drawFocus](#) () const
Draws a focus rectangle around the widget.
- void [drawFocus](#) ([FL_Boxtype](#) t, int X, int Y, int W, int H) const

- Draws a focus rectangle around the widget.*
- void `draw_focus` (`Fl_Boxtype` t, int x, int y, int w, int h, `Fl_Color` bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `Fl_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

11.65.1 Detailed Description

The `Fl_Hold_Browser` is a subclass of `Fl_Browser` which lets the user select a single item, or no items by clicking on the empty space.



Figure 11.24 `Fl_Hold_Browser`

As long as the mouse button is held down the item pointed to by it is highlighted, and this highlighting remains on when the mouse button is released. Normally the callback is done when the user releases the mouse, but you can change this with `when()`.

See `Fl_Browser` for methods to add and remove lines from the browser.

11.65.2 Constructor & Destructor Documentation

11.65.2.1 `Fl_Hold_Browser()`

```
Fl_Hold_Browser::Fl_Hold_Browser (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new `Fl_Hold_Browser` widget using the given position, size, and label string.

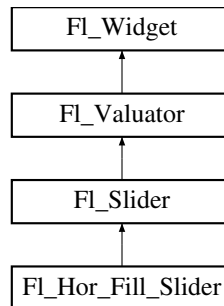
The default boxttype is `FL_DOWN_BOX`. The constructor specializes `Fl_Browser()` by setting the type to `FL_HOLD_BROWSER`. The destructor destroys the widget and frees all memory that has been allocated.

The documentation for this class was generated from the following files:

- `Fl_Hold_Browser.H`
- `Fl_Browser.cxx`

11.66 FI_Hor_Fill_Slider Class Reference

Inheritance diagram for FI_Hor_Fill_Slider:



Public Member Functions

- **FI_Hor_Fill_Slider** (int X, int Y, int W, int H, const char *L=0)

Public Member Functions inherited from FI_Slider

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- **FI_Slider** (int X, int Y, int W, int H, const char *L=0)
Creates a new FI_Slider widget using the given position, size, and label string.
- **FI_Slider** (uchar t, int X, int Y, int W, int H, const char *L)
Creates a new FI_Slider widget using the given type, position, size, and label string.
- int **handle** (int) **FL_OVERRIDE**
Handles the specified event.
- int **scrollvalue** (int pos, int size, int first, int total)
Sets the size and position of the sliding knob in the box.
- **FI_Boxtype slider** () const
Gets the slider box type.
- void **slider** (FI_Boxtype c)
Sets the slider box type.
- float **slider_size** () const
Get the dimensions of the moving piece of slider.
- void **slider_size** (double v)
Set the dimensions of the moving piece of slider.

Public Member Functions inherited from FI_Valuator

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)

- Sets the maximum value for the valuator.*

 - double **minimum** () const

Gets the minimum value for the valuator.

 - void **minimum** (double a)

Sets the minimum value for the valuator.

 - void **precision** (int digits)

Sets the step value to $1.0 / 10^{\text{digits}}$.

 - void **range** (double a, double b)

Sets the minimum and maximum values for the valuator.

 - double **round** (double)

Round the passed value to the nearest step increment.

 - double **step** () const

Gets or sets the step value.

 - void **step** (double a, int b)

See double [FI_Valuator::step\(\)](#) const
- void **step** (double s)
- See double [FI_Valuator::step\(\)](#) const.*

 - void **step** (int a)

See double [FI_Valuator::step\(\)](#) const
- double **value** () const
- Gets the floating point(double) value.*

 - int **value** (double)

Sets the current value.

 - **~FI_Valuator** () **FL_OVERRIDE**

Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
- Activates the widget.*

 - unsigned int **active** () const

Returns whether the widget is active.
- int **active_r** () const
- Returns whether the widget and all of its parents are active.*

 - **FI_Align align** () const

Gets the label alignment.
- void **align** (**FI_Align** alignment)
- Sets the label alignment.*

 - long **argument** () const

Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
- Sets the current user data (long) argument that is passed to the callback function.*

 - virtual class **FI_Gl_Window** * **as_gl_window** ()

Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class **FI_Gl_Window** const * **as_gl_window** () const
- virtual **FI_Group** * **as_group** ()
- Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).*

 - virtual **FI_Group** const * **as_group** () const

- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.

- int [contains](#) (const [Fl_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [Fl_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [Fl_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([Fl_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([Fl_Widget](#) *widget, long arg, [Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([Fl_Widget](#) *widget, void *arg=0, [Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [Fl_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [Fl_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [Fl_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([Fl_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const

- Returns whether the image is managed by the widget.*
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const
Gets the label type.
- void `labeltype` (`FL_Labeltype` a)
Sets the label type.
- void `measure_label` (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const
Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int `output` () const
Returns if a widget is used for output only.
- `FL_Group` * `parent` () const
Returns a pointer to the parent widget.
- void `parent` (`FL_Group` *p)
Internal use only - "for hacks only".
- void `position` (int X, int Y)
Repositions the window or widget.
- void `redraw` ()
Schedules the drawing of the widget.
- void `redraw_label` ()
Schedules the drawing of the label.
- virtual void `resize` (int x, int y, int w, int h)
Changes the size or position of the widget.

- [FI_Color selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)

- Set the spacing between the label and the vertical edge of the widget.*

 - unsigned int `visible` () const
 - Returns whether a widget is visible.*
 - unsigned int `visible_focus` () const
 - Checks whether this widget has a visible focus.*
 - void `visible_focus` (int v)
 - Modifies keyboard focus navigation.*
 - int `visible_r` () const
 - Returns whether a widget and all its parents are visible.*
 - int `w` () const
 - Gets the widget width.*
 - `FI_When` when () const
 - Returns the conditions under which the callback is called.*
 - void `when` (uchar i)
 - Sets the flags used to decide when a callback is called.*
 - `FI_Window` * `window` () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
 - int `x` () const
 - Gets the widget position in its window.*
 - int `y` () const
 - Gets the widget position in its window.*
 - virtual `~FI_Widget` ()
 - Destroys the widget.*

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *widget, void *data)
- The default callback for all widgets that don't set a callback.*
- static unsigned int `label_shortcut` (const char *t)
- Returns the Unicode value of the '&x' shortcut in a given text.*
- static int `test_shortcut` (const char *, const bool require_alt=false)
- Returns true if the given text t contains the entered '&x' shortcut.*

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from [FI_Slider](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void **draw** (int, int, int, int)
- int **handle** (int, int, int, int, int)

Protected Member Functions inherited from [FI_Valuator](#)

- [FI_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FI_Valuator](#) widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an [FL_WHEN_CHANGED](#) event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an [FL_WHEN_RELEASE](#) event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an [FL_HORIZONTAL](#) one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void [value_damage](#) ()
Asks for partial redraw.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If [FL_ALIGN_IMAGE_BACKDROP](#) is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.

- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

The documentation for this class was generated from the following files:

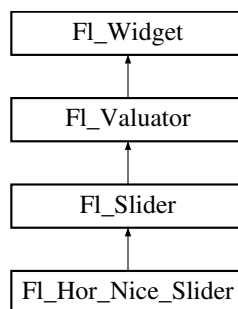
- Fl_Hor_Fill_Slider.H
- Fl_Slider.cxx

11.67 Fl_Hor_Nice_Slider Class Reference

Single thumb tab slider.

```
#include <Fl_Hor_Nice_Slider.H>
```

Inheritance diagram for Fl_Hor_Nice_Slider:



Public Member Functions

- **Fl_Hor_Nice_Slider** (int X, int Y, int W, int H, const char *L=0)

Public Member Functions inherited from **Fl_Slider**

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- **Fl_Slider** (int X, int Y, int W, int H, const char *L=0)
*Creates a new **Fl_Slider** widget using the given position, size, and label string.*
- **Fl_Slider** (uchar t, int X, int Y, int W, int H, const char *L)
*Creates a new **Fl_Slider** widget using the given type, position, size, and label string.*
- int **handle** (int) **FL_OVERRIDE**
Handles the specified event.
- int **scrollvalue** (int pos, int **size**, int first, int total)
Sets the size and position of the sliding knob in the box.
- **Fl_Boxtype slider** () const
Gets the slider box type.
- void **slider** (**Fl_Boxtype** c)

- *Sets the slider box type.*
- float **slider_size** () const
Get the dimensions of the moving piece of slider.
- void **slider_size** (double v)
Set the dimensions of the moving piece of slider.

Public Member Functions inherited from **FI_Valuator**

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)
Sets the maximum value for the valuator.
- double **minimum** () const
Gets the minimum value for the valuator.
- void **minimum** (double a)
Sets the minimum value for the valuator.
- void **precision** (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void **range** (double a, double b)
Sets the minimum and maximum values for the valuator.
- double **round** (double)
Round the passed value to the nearest step increment.
- double **step** () const
Gets or sets the step value.
- void **step** (double a, int b)
See double [FI_Valuator::step\(\)](#) const
- void **step** (double s)
See double [FI_Valuator::step\(\)](#) const.
- void **step** (int a)
See double [FI_Valuator::step\(\)](#) const
- double **value** () const
Gets the floating point(double) value.
- int **value** (double)
Sets the current value.
- **~FI_Valuator** () **FL_OVERRIDE**
Destructor is accessible despite protected constructor.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (**FI_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FI_Gl_Window** * **as_gl_window** ()
Returns an FI_Gl_Window pointer if this widget is an FI_Gl_Window.
- virtual class **FI_Gl_Window** const * **as_gl_window** () const
- virtual **FI_Group** * **as_group** ()
Returns an FI_Group pointer if this widget is an FI_Group.
- virtual **FI_Group** const * **as_group** () const
- virtual **FI_Window** * **as_window** ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual **FI_Window** const * **as_window** () const
- void **bind_deimage** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- **FI_Boxtype box** () const
Gets the box type of the widget.
- void **box** (**FI_Boxtype** new_box)
Sets the box type for the widget.
- **FI_Callback_p callback** () const
Gets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb, **FI_Callback_User_Data** *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void **callback** (**FI_Callback** *cb, void *p)
Sets the current callback function and data for the widget.
- void **callback** (**FI_Callback0** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback1** *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int **changed** () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FL_Color` bg)

Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)

Sets the background and selection color of the widget.
- `FL_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const
- Sets width ww and height hh accordingly with the label size.*

 - bool [needs_keyboard](#) () const
- Returns whether this widget needs a keyboard.*

 - void [needs_keyboard](#) (bool needs)
- Sets whether this widget needs a keyboard.*

 - unsigned int [output](#) () const
- Returns if a widget is used for output only.*

 - [Fl_Group](#) * [parent](#) () const
- Returns a pointer to the parent widget.*

 - void [parent](#) ([Fl_Group](#) *p)
- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
- Repositions the window or widget.*

 - void [redraw](#) ()
- Schedules the drawing of the widget.*

 - void [redraw_label](#) ()
- Schedules the drawing of the label.*

 - virtual void [resize](#) (int x, int y, int w, int h)
- Changes the size or position of the widget.*

 - [Fl_Color](#) [selection_color](#) () const
- Gets the selection color.*

 - void [selection_color](#) ([Fl_Color](#) a)
- Sets the selection color.*

 - void [set_active](#) ()
- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

 - void [set_output](#) ()
- Sets a widget to output only.*

 - void [set_visible](#) ()
- Makes the widget visible.*

 - void [set_visible_focus](#) ()
- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void [shortcut_label](#) (int value)
- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - virtual void [show](#) ()
- Makes a widget visible.*

 - void [size](#) (int W, int H)
- Changes the size of the widget.*

 - int [take_focus](#) ()
- Gives the widget the keyboard focus.*

 - unsigned int [takeevents](#) () const
- Returns if the widget is able to take events.*

 - int [test_shortcut](#) ()
- Returns true if the widget's label contains the entered '&x' shortcut.*

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from [FL_Widget](#)

- static void [default_callback](#) ([FL_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [FL_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from [FL_Slider](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void **draw** (int, int, int, int)
- int **handle** (int, int, int, int, int)

Protected Member Functions inherited from [FL_Valuator](#)

- [FL_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FL_Valuator](#) widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void [value_damage](#) ()
Asks for partial redraw.

Protected Member Functions inherited from [Fl_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (Fl_Boxtype t, Fl_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **Fl_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.67.1 Detailed Description

Single thumb tab slider.

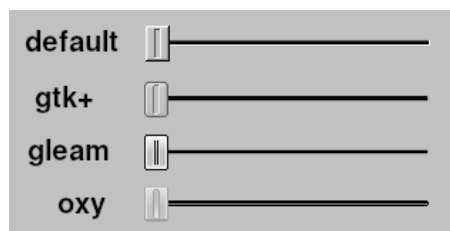


Figure 11.25 `Fl_Hor_Nice_Slider` with various `Fl::scheme()` values

The documentation for this class was generated from the following files:

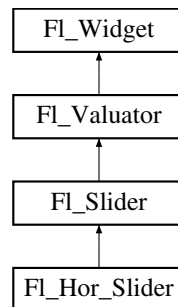
- `Fl_Hor_Nice_Slider.H`
- `Fl_Slider.cxx`

11.68 FI_Hor_Slider Class Reference

Horizontal Slider class.

```
#include <Fl_Hor_Slider.H>
```

Inheritance diagram for `FI_Hor_Slider`:



Public Member Functions

- **FI_Hor_Slider** (int X, int Y, int W, int H, const char *l=0)
Creates a new [FI_Hor_Slider](#) widget using the given position, size, and label string.

Public Member Functions inherited from [FI_Slider](#)

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- [FI_Slider](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [FI_Slider](#) widget using the given position, size, and label string.
- **FI_Slider** (uchar t, int X, int Y, int W, int H, const char *L)
Creates a new [FI_Slider](#) widget using the given type, position, size, and label string.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- int **scrollvalue** (int pos, int [size](#), int first, int total)
Sets the size and position of the sliding knob in the box.
- [FI_Boxtype](#) **slider** () const
Gets the slider box type.
- void **slider** ([FI_Boxtype](#) c)
Sets the slider box type.
- float **slider_size** () const
Get the dimensions of the moving piece of slider.
- void **slider_size** (double v)
Set the dimensions of the moving piece of slider.

Public Member Functions inherited from [FI_Valuator](#)

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const

- Gets the maximum value for the valuator.*

 - void **maximum** (double a)

Sets the maximum value for the valuator.

 - double **minimum** () const

Gets the minimum value for the valuator.

 - void **minimum** (double a)

Sets the minimum value for the valuator.

 - void **precision** (int digits)

Sets the step value to $1.0 / 10^{\text{digits}}$.

 - void **range** (double a, double b)

Sets the minimum and maximum values for the valuator.

 - double **round** (double)

Round the passed value to the nearest step increment.

 - double **step** () const

Gets or sets the step value.

 - void **step** (double a, int b)

See double [FI_Valuator::step\(\)](#) const

 - void **step** (double s)

See double [FI_Valuator::step\(\)](#) const.

 - void **step** (int a)

See double [FI_Valuator::step\(\)](#) const

 - double **value** () const

Gets the floating point(double) value.

 - int **value** (double)

Sets the current value.

 - **~FI_Valuator** () **FL_OVERRIDE**

Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
 - void **_set_fullscreen** ()
 - void **activate** ()
- Activates the widget.*
- unsigned int **active** () const
- Returns whether the widget is active.*
- int **active_r** () const
- Returns whether the widget and all of its parents are active.*
- [FI_Align](#) **align** () const
- Gets the label alignment.*
- void **align** ([FI_Align](#) alignment)
- Sets the label alignment.*
- long **argument** () const
- Gets the current user data (long) argument that is passed to the callback function.*
- void **argument** (long v)
- Sets the current user data (long) argument that is passed to the callback function.*
- virtual class [FI_Gl_Window](#) * **as_gl_window** ()
- Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).*
- virtual class [FI_Gl_Window](#) const * **as_gl_window** () const
- virtual [FI_Group](#) * **as_group** ()

- Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).*

 - virtual [FL_Group](#) const * **as_group** () const
 - virtual [FL_Window](#) * **as_window** ()
- Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).*

 - virtual [FL_Window](#) const * **as_window** () const
 - void **bind_deimage** ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.

 - void **bind_deimage** (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.

 - void **bind_image** ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.

 - void **bind_image** (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.

 - [FL_Boxtype](#) **box** () const

Gets the box type of the widget.

 - void **box** ([FL_Boxtype](#) new_box)

Sets the box type for the widget.

 - [FL_Callback_p](#) **callback** () const

Gets the current callback function for the widget.

 - void **callback** ([FL_Callback](#) *cb)

Sets the current callback function for the widget.

 - void **callback** ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.

 - void **callback** ([FL_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.

 - void **callback** ([FL_Callback0](#) *cb)

Sets the current callback function for the widget.

 - void **callback** ([FL_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.

 - unsigned int **changed** () const

Checks if the widget value changed since the last callback.

 - void **clear_active** ()

Marks the widget as inactive without sending events or changing focus.

 - void **clear_changed** ()

Marks the value of the widget as unchanged.

 - void **clear_damage** (uchar c=0)

Clears or sets the damage flags.

 - void **clear_output** ()

Sets a widget to accept input.

 - void **clear_visible** ()

Hides the widget.

 - void **clear_visible_focus** ()

Disables keyboard focus navigation with this widget.

 - [FL_Color](#) **color** () const

Gets the background color of the widget.

 - void **color** ([FL_Color](#) bg)

Sets the background color of the widget.

 - void **color** ([FL_Color](#) bg, [FL_Color](#) sel)

Sets the background and selection color of the widget.

 - [FL_Color](#) **color2** () const

For back compatibility only.

- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FL_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FL_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)

- Sets the image to use as part of the widget label when in the active state.*

 - int `image_bound` () const

Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const

Checks if this widget is a child of wgt.
- int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const

Gets the current label text.
- void `label` (const char *text)

Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const

Gets the label color.
- void `labelcolor` (`FL_Color` c)

Sets the label color.
- `FL_Font` `labelfont` () const

Gets the font to use.
- void `labelfont` (`FL_Font` f)

Sets the font to use.
- `FL_Fonsize` `labelsize` () const

Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)

Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const

Gets the label type.
- void `labeltype` (`FL_Labeltype` a)

Sets the label type.
- void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FL_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FL_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.

- virtual void **resize** (int x, int y, int w, int h)
Changes the size or position of the widget.
- **FL_Color selection_color** () const
Gets the selection color.
- void **selection_color** (FL_Color a)
Sets the selection color.
- void **set_active** ()
Marks the widget as active without sending events or changing focus.
- void **set_changed** ()
Marks the value of the widget as changed.
- void **set_output** ()
Sets a widget to output only.
- void **set_visible** ()
Makes the widget visible.
- void **set_visible_focus** ()
Enables keyboard focus navigation with this widget.
- int **shortcut_label** () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void **shortcut_label** (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void **show** ()
Makes a widget visible.
- void **size** (int W, int H)
Changes the size of the widget.
- int **take_focus** ()
Gives the widget the keyboard focus.
- unsigned int **takeevents** () const
Returns if the widget is able to take events.
- int **test_shortcut** ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * **tooltip** () const
Gets the current tooltip text.
- void **tooltip** (const char *text)
Sets the current tooltip text.
- FL_Window * **top_window** () const
Returns a pointer to the top-level window for the widget.
- FL_Window * **top_window_offset** (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- uchar **type** () const
Gets the widget type.
- void **type** (uchar t)
Sets the widget type.
- int **use_accents_menu** ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const
Gets the user data for this widget.
- void **user_data** (FL_Callback_User_Data *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()

- Get the spacing between the label and the vertical edge of the widget.*

 - void `vertical_label_margin` (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int `visible` () const

Returns whether a widget is visible.
- unsigned int `visible_focus` () const

Checks whether this widget has a visible focus.
- void `visible_focus` (int v)

Modifies keyboard focus navigation.
- int `visible_r` () const

Returns whether a widget and all its parents are visible.
- int `w` () const

Gets the widget width.
- `Fl_When` `when` () const

Returns the conditions under which the callback is called.
- void `when` (uchar i)

Sets the flags used to decide when a callback is called.
- `Fl_Window` * `window` () const

Returns a pointer to the nearest parent window up the widget hierarchy.
- int `x` () const

Gets the widget position in its window.
- int `y` () const

Gets the widget position in its window.
- virtual `~Fl_Widget` ()

Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget` *widget, void *data)

The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)

Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)

Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `Fl_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
,
`OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
,
`MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
,
`GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from [FL_Slider](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void **draw** (int, int, int, int)
- int **handle** (int, int, int, int, int)

Protected Member Functions inherited from [FL_Valuator](#)

- [FL_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FL_Valuator](#) widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void [value_damage](#) ()
Asks for partial redraw.

Protected Member Functions inherited from [FL_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FL_Boxtype](#) t, [FL_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FL_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FL_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.

- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.68.1 Detailed Description

Horizontal Slider class.

See also

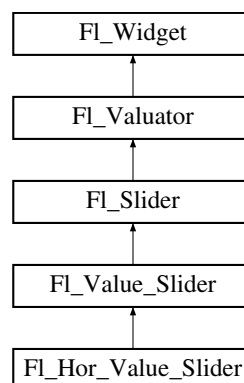
class [Fl_Slider](#).

The documentation for this class was generated from the following files:

- [Fl_Hor_Slider.H](#)
- [Fl_Slider.cxx](#)

11.69 Fl_Hor_Value_Slider Class Reference

Inheritance diagram for [Fl_Hor_Value_Slider](#):



Public Member Functions

- **Fl_Hor_Value_Slider** (int X, int Y, int W, int H, const char *l=0)

Public Member Functions inherited from [Fl_Value_Slider](#)

- [Fl_Value_Slider](#) (int x, int y, int w, int h, const char *l=0)
Creates a new [Fl_Value_Slider](#) widget using the given position, size, and label string.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- [Fl_Color](#) **textcolor** () const
Gets the color of the text in the value box.
- void **textcolor** ([Fl_Color](#) s)

- Sets the color of the text in the value box.*
- **FI_Font** **textfont** () const
Gets the typeface of the text in the value box.
- void **textfont** (FI_Font s)
Sets the typeface of the text in the value box.
- **FI_Fontsize** **textsize** () const
Gets the size of the text in the value box.
- void **textsize** (FI_Fontsize s)
Sets the size of the text in the value box.
- int **value_height** () const
Gets the height of the value box in pixels (vertical mode only).
- void **value_height** (int s)
Sets the height of the value box in pixels (vertical mode only).
- int **value_width** () const
Gets the width of the value box in pixels (horizontal mode only).
- void **value_width** (int s)
Sets the width of the value box in pixels (horizontal mode only).

Public Member Functions inherited from FI_Slider

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- **FI_Slider** (int X, int Y, int W, int H, const char *L=0)
Creates a new FI_Slider widget using the given position, size, and label string.
- **FI_Slider** (uchar t, int X, int Y, int W, int H, const char *L)
Creates a new FI_Slider widget using the given type, position, size, and label string.
- int **scrollvalue** (int pos, int size, int first, int total)
Sets the size and position of the sliding knob in the box.
- **FI_Boxtype** **slider** () const
Gets the slider box type.
- void **slider** (FI_Boxtype c)
Sets the slider box type.
- float **slider_size** () const
Get the dimensions of the moving piece of slider.
- void **slider_size** (double v)
Set the dimensions of the moving piece of slider.

Public Member Functions inherited from FI_Valuator

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)
Sets the maximum value for the valuator.
- double **minimum** () const

- Gets the minimum value for the valuator.*
- void **minimum** (double a)
 - Sets the minimum value for the valuator.*
- void **precision** (int digits)
 - Sets the step value to $1.0 / 10^{\text{digits}}$.*
- void **range** (double a, double b)
 - Sets the minimum and maximum values for the valuator.*
- double **round** (double)
 - Round the passed value to the nearest step increment.*
- double **step** () const
 - Gets or sets the step value.*
- void **step** (double a, int b)
 - See double [FI_Valuator::step\(\)](#) const*
- void **step** (double s)
 - See double [FI_Valuator::step\(\)](#) const.*
- void **step** (int a)
 - See double [FI_Valuator::step\(\)](#) const*
- double **value** () const
 - Gets the floating point(double) value.*
- int **value** (double)
 - Sets the current value.*
- ~**FI_Valuator** () **FL_OVERRIDE**
 - Destructor is accessible despite protected constructor.*

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
 - Activates the widget.*
- unsigned int **active** () const
 - Returns whether the widget is active.*
- int **active_r** () const
 - Returns whether the widget and all of its parents are active.*
- [FI_Align](#) **align** () const
 - Gets the label alignment.*
- void **align** ([FI_Align](#) alignment)
 - Sets the label alignment.*
- long **argument** () const
 - Gets the current user data (long) argument that is passed to the callback function.*
- void **argument** (long v)
 - Sets the current user data (long) argument that is passed to the callback function.*
- virtual class [FI_Gl_Window](#) * **as_gl_window** ()
 - Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).*
- virtual class [FI_Gl_Window](#) const * **as_gl_window** () const
- virtual [FI_Group](#) * **as_group** ()
 - Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).*
- virtual [FI_Group](#) const * **as_group** () const
- virtual [FI_Window](#) * **as_window** ()
 - Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*

- virtual [FI_Window](#) const * **as_window** () const
- void [bind_deimage](#) ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) **box** () const

Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)

Sets the box type for the widget.
- [FI_Callback_p](#) **callback** () const

Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()

Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.
- void [clear_output](#) ()

Sets a widget to accept input.
- void [clear_visible](#) ()

Hides the widget.
- void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.
- [FI_Color](#) **color** () const

Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)

Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)

Sets the background and selection color of the widget.
- [FI_Color](#) **color2** () const

For back compatibility only.
- void [color2](#) (unsigned a)

For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const

Checks if w is a child of this widget.

- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)
Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FL_Image * deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)
Calls the widget callback function with default arguments.
- void `do_callback` (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, FL_Align) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image * image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (FL_Image &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (FL_Image *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const FL_Widget *wgt) const

- Checks if this widget is a child of wgt.*

 - int [is_label_copied](#) () const

Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const

Gets the current label text.
- void [label](#) (const char *text)

Sets the current label pointer.
- void [label](#) (FL_Labeltype a, const char *b)

Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()

Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)

Set the gap between the label and the image in pixels.
- FL_Color [labelcolor](#) () const

Gets the label color.
- void [labelcolor](#) (FL_Color c)

Sets the label color.
- FL_Font [labelfont](#) () const

Gets the font to use.
- void [labelfont](#) (FL_Font f)

Sets the font to use.
- FL_Fonsize [labelsize](#) () const

Gets the font size in pixels.
- void [labelsize](#) (FL_Fonsize pix)

Sets the font size in pixels.
- FL_Labeltype [labeltype](#) () const

Gets the label type.
- void [labeltype](#) (FL_Labeltype a)

Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- FL_Group * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) (FL_Group *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- FL_Color [selection_color](#) () const

Gets the selection color.

- void `selection_color` (`Fl_Color` a)
Sets the selection color.
- void `set_active` ()
Marks the widget as active without sending events or changing focus.
- void `set_changed` ()
Marks the value of the widget as changed.
- void `set_output` ()
Sets a widget to output only.
- void `set_visible` ()
Makes the widget visible.
- void `set_visible_focus` ()
Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()
Makes a widget visible.
- void `size` (int W, int H)
Changes the size of the widget.
- int `take_focus` ()
Gives the widget the keyboard focus.
- unsigned int `takesevents` () const
Returns if the widget is able to take events.
- int `test_shortcut` ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const
Gets the current tooltip text.
- void `tooltip` (const char *text)
Sets the current tooltip text.
- `Fl_Window` * `top_window` () const
Returns a pointer to the top-level window for the widget.
- `Fl_Window` * `top_window_offset` (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar` `type` () const
Gets the widget type.
- void `type` (`uchar` t)
Sets the widget type.
- int `use_accents_menu` ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * `user_data` () const
Gets the user data for this widget.
- void `user_data` (`Fl_Callback_User_Data` *v, bool auto_free)
Sets the user data for this widget.
- void `user_data` (void *v)
Sets the user data for this widget.
- int `vertical_label_margin` ()
Get the spacing between the label and the vertical edge of the widget.
- void `vertical_label_margin` (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int `visible` () const

- Returns whether a widget is visible.*
- unsigned int `visible_focus` () const
Checks whether this widget has a visible focus.
- void `visible_focus` (int v)
Modifies keyboard focus navigation.
- int `visible_r` () const
Returns whether a widget and all its parents are visible.
- int `w` () const
Gets the widget width.
- `Fl_When` when () const
Returns the conditions under which the callback is called.
- void `when` (uchar i)
Sets the flags used to decide when a callback is called.
- `Fl_Window` * `window` () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual `~Fl_Widget` ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget` *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `Fl_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from `Fl_Value_Slider`

- void `draw` () `FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from [FI_Slider](#)

- void **draw** (int, int, int, int)
- int **handle** (int, int, int, int, int)

Protected Member Functions inherited from [FI_Valuator](#)

- [FI_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FI_Valuator](#) widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void **value_damage** ()
Asks for partial redraw.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.

- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

The documentation for this class was generated from the following files:

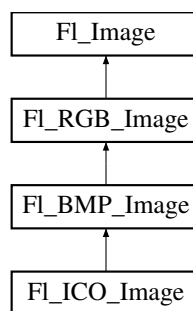
- FI_Hor_Value_Slider.H
- FI_Value_Slider.cxx

11.70 FI_ICO_Image Class Reference

The **FI_ICO_Image** class supports loading, caching, and drawing of Windows icon (.ico) files.

```
#include <FI_ICO_Image.H>
```

Inheritance diagram for FI_ICO_Image:



Classes

- struct **IconDirEntry**
Windows ICONDIRENTRY structure

Public Member Functions

- **FI_ICO_Image** (const char *filename, int id=-1, const unsigned char *data=NULL, const size_t datasize=0)
Loads the named icon image from the given .ico filename or from memory.
- const **IconDirEntry** * **icondirentry** () const
*Returns the array of **idcount**() loaded **IconDirEntry** structures.*
- int **idcount** () const
Returns the number of icons of various resolutions present in the ICO object.
- **~FI_ICO_Image** ()
Destructor.

Public Member Functions inherited from FI_BMP_Image

- **FI_BMP_Image** (const char *filename)
This constructor loads the named BMP image from the given BMP filename.
- **FI_BMP_Image** (const char *imagenname, const unsigned char *data, const long length=-1)
This constructor loads a BMP image from memory.

Public Member Functions inherited from [FI_RGB_Image](#)

- virtual [FI_SVG_Image](#) * [as_svg_image](#) ()
Returns whether an image is an [FI_SVG_Image](#) or not.
- int [cache_h](#) ()
- int [cache_w](#) ()
- void [color_average](#) ([FI_Color](#) c, float i) [FL_OVERRIDE](#)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [FI_Image](#) * [copy](#) () const
- [FI_Image](#) * [copy](#) (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void [desaturate](#) () [FL_OVERRIDE](#)
The [desaturate\(\)](#) method converts an image to grayscale.
- void [draw](#) (int X, int Y)
- void [draw](#) (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [FI_RGB_Image](#) (const [FI_Pixmap](#) *pxm, [FI_Color](#) bg=[FL_GRAY](#))
The constructor creates a new RGBA image from the specified [FI_Pixmap](#).
- [FI_RGB_Image](#) (const [uchar](#) *bits, int bits_length, int W, int H, int D, int LD)
The constructor creates a new image from the specified data.
- [FI_RGB_Image](#) (const [uchar](#) *bits, int W, int H, int D=3, int LD=0)
The constructor creates a new image from the specified data.
- void [label](#) ([FI_Menu_Item](#) *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.
- void [label](#) ([FI_Widget](#) *w) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a widget or menu item.
- virtual void [normalize](#) ()
Makes sure the object is fully initialized.
- void [uncache](#) () [FL_OVERRIDE](#)
If the image has been cached for display, delete the cache data.
- ~[FI_RGB_Image](#) () [FL_OVERRIDE](#)
The destructor frees all memory and server resources that are used by the image.

Public Member Functions inherited from [FI_Image](#)

- virtual class [FI_Shared_Image](#) * [as_shared_image](#) ()
Returns whether an image is an [FI_Shared_Image](#) or not.
- [FI_Image](#) * [copy](#) () const
Creates a copy of the image in the same size.
- int [count](#) () const
Returns the number of data values associated with the image.
- int [d](#) () const
Returns the image depth.
- const char *const * [data](#) () const
Returns a pointer to the current image data array.
- int [data_h](#) () const
Returns the height of the image data.
- int [data_w](#) () const
Returns the width of the image data.
- void [draw](#) (int X, int Y)
Draws the image to the current drawing surface.
- int [fail](#) () const

- Returns a value that is not 0 if there is currently no image available.*
- [FI_Image](#) (int W, int H, int D)
 - The constructor creates an empty image with the specified width, height, and depth.*
- int [h](#) () const
 - Returns the current image drawing height in FLTK units.*
- void [inactive](#) ()
 - The [inactive\(\)](#) method calls [color_average\(FI_BACKGROUND_COLOR, 0.33f\)](#) to produce an image that appears grayed out.*
- int [ld](#) () const
 - Returns the current line data size in bytes.*
- virtual void [release](#) ()
 - Releases an [FI_Image](#) - the same as 'delete this'.*
- virtual void [scale](#) (int width, int height, int proportional=1, int can_expand=0)
 - Sets the drawing size of the image.*
- int [w](#) () const
 - Returns the current image drawing width in FLTK units.*
- virtual [~FI_Image](#) ()
 - The destructor is a virtual method that frees all memory used by the image.*

Additional Inherited Members

Static Public Member Functions inherited from [FI_RGB_Image](#)

- static size_t [max_size](#) ()
 - Returns the maximum allowed image size in bytes when creating an [FI_RGB_Image](#) object.*
- static void [max_size](#) (size_t size)
 - Sets the maximum allowed image size in bytes when creating an [FI_RGB_Image](#) object.*

Static Public Member Functions inherited from [FI_Image](#)

- static [FI_Labeltype](#) [define_FL_IMAGE_LABEL](#) ()
- static [FI_RGB_Scaling](#) [RGB_scaling](#) ()
 - Returns the currently used RGB image scaling method.*
- static void [RGB_scaling](#) ([FI_RGB_Scaling](#))
 - Sets the RGB image scaling method used for copy(int, int).*
- static [FI_RGB_Scaling](#) [scaling_algorithm](#) ()
 - Gets what algorithm is used when resizing a source image to draw it.*
- static void [scaling_algorithm](#) ([FI_RGB_Scaling](#) algorithm)
 - Sets what algorithm is used when resizing a source image to draw it.*

Public Attributes inherited from [FI_RGB_Image](#)

- int [alloc_array](#)
 - If non-zero, the object's data array is delete[]'d when deleting the object.*
- const [uchar](#) * [array](#)
 - Points to the start of the object's data array.*

Static Public Attributes inherited from [FI_Image](#)

- static const int [ERR_FILE_ACCESS](#) = -2
- static const int [ERR_FORMAT](#) = -3
- static const int [ERR_MEMORY_ACCESS](#) = -4
- static const int [ERR_NO_IMAGE](#) = -1
- static bool [register_images_done](#) = false
 - True after [fl_register_images\(\)](#) was called, false before.*

Protected Member Functions inherited from [FI_BMP_Image](#)

- void **load_bmp_** (class [FI_Image_Reader](#) &rdr, int ico_height=0, int ico_width=0)

Protected Member Functions inherited from [FI_Image](#)

- void **d** (int D)
Sets the current image depth.
- void **data** (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void **draw_empty** (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int **draw_scaled** (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void **h** (int H)
Sets the height of the image data.
- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Static Protected Member Functions inherited from [FI_Image](#)

- static void **labeltype** (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void **measure** (const [FI_Label](#) *lo, int &lw, int &lh)

11.70.1 Detailed Description

The [FI_ICO_Image](#) class supports loading, caching, and drawing of Windows icon (.ico) files.

11.70.2 Constructor & Destructor Documentation

11.70.2.1 [FI_ICO_Image](#)()

```
FI_ICO_Image::FI_ICO_Image (
    const char * filename,
    int id = -1,
    const unsigned char * data = NULL,
    const size_t datasize = 0)
```

Loads the named icon image from the given .ico filename or from memory.

Parameters

<i>filename</i>	Name of a .ico file, or of the in-memory image
<i>id</i>	When id is -1 (default), the highest-resolution icon is loaded; when id 0, load the icon with this ID; when id = -2, load all IconDirEntry structures but no image.
<i>data</i>	NULL, or in-memory icon data
<i>datasize</i>	Size in bytes of the <i>data</i> byte array (used when <i>data</i> is not NULL)

The documentation for this class was generated from the following files:

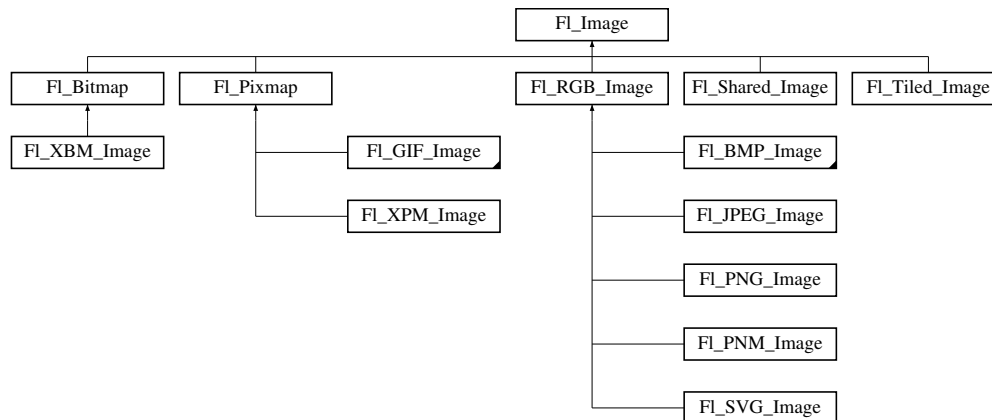
- [FI_ICO_Image.H](#)
- [FI_ICO_Image.cxx](#)

11.71 Fl_Image Class Reference

Base class for image caching, scaling and drawing.

```
#include <Fl_Image.H>
```

Inheritance diagram for Fl_Image:



Public Member Functions

- virtual class `Fl_Shared_Image *` `as_shared_image ()`
Returns whether an image is an `Fl_Shared_Image` or not.
- virtual void `color_average (Fl_Color c, float i)`
The `color_average()` method averages the colors in the image with the provided FLTK color value.
- `Fl_Image *` `copy ()` const
Creates a copy of the image in the same size.
- virtual `Fl_Image *` `copy (int W, int H)` const
Creates a resized copy of the image.
- int `count ()` const
Returns the number of data values associated with the image.
- int `d ()` const
Returns the image depth.
- const char *const * `data ()` const
Returns a pointer to the current image data array.
- int `data_h ()` const
Returns the height of the image data.
- int `data_w ()` const
Returns the width of the image data.
- virtual void `desaturate ()`
The `desaturate()` method converts an image to grayscale.
- void `draw (int X, int Y)`
Draws the image to the current drawing surface.
- virtual void `draw (int X, int Y, int W, int H, int cx=0, int cy=0)`
Draws the image to the current drawing surface with a bounding box.
- int `fail ()` const
Returns a value that is not 0 if there is currently no image available.
- `Fl_Image` (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int `h ()` const
Returns the current image drawing height in FLTK units.
- void `inactive ()`

The `inactive()` method calls `color_average(FL_BACKGROUND_COLOR, 0.33f)` to produce an image that appears grayed out.

- virtual void `label` (`FL_Menu_Item` *m)
This method is an obsolete way to set the image attribute of a menu item.
- virtual void `label` (`FL_Widget` *w)
This method is an obsolete way to set the image attribute of a widget or menu item.
- int `ld` () const
Returns the current line data size in bytes.
- virtual void `release` ()
Releases an `FL_Image` - the same as 'delete this'.
- virtual void `scale` (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- virtual void `uncache` ()
If the image has been cached for display, delete the cache data.
- int `w` () const
Returns the current image drawing width in FLTK units.
- virtual `~FI_Image` ()
The destructor is a virtual method that frees all memory used by the image.

Static Public Member Functions

- static `FL_Labeltype` `define_FL_IMAGE_LABEL` ()
- static `FL_RGB_Scaling` `RGB_scaling` ()
Returns the currently used RGB image scaling method.
- static void `RGB_scaling` (`FL_RGB_Scaling`)
Sets the RGB image scaling method used for copy(int, int).
- static `FL_RGB_Scaling` `scaling_algorithm` ()
Gets what algorithm is used when resizing a source image to draw it.
- static void `scaling_algorithm` (`FL_RGB_Scaling` algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Static Public Attributes

- static const int `ERR_FILE_ACCESS` = -2
- static const int `ERR_FORMAT` = -3
- static const int `ERR_MEMORY_ACCESS` = -4
- static const int `ERR_NO_IMAGE` = -1
- static bool `register_images_done` = false
True after `fl_register_images()` was called, false before.

Protected Member Functions

- void `d` (int D)
Sets the current image depth.
- void `data` (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void `draw_empty` (int X, int Y)
The protected method `draw_empty()` draws a box with an X in it.
- int `draw_scaled` (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void `h` (int H)
Sets the height of the image data.
- void `ld` (int LD)

Sets the current line data size in bytes.

- void [w](#) (int W)

Sets the width of the image data.

Static Protected Member Functions

- static void **labeltype** (const [Fl_Label](#) *lo, int lx, int ly, int lw, int lh, [Fl_Align](#) la)
- static void **measure** (const [Fl_Label](#) *lo, int &lw, int &lh)

Friends

- class [Fl_Graphics_Driver](#)

11.71.1 Detailed Description

Base class for image caching, scaling and drawing.

[Fl_Image](#) is the base class used for caching, scaling and drawing all kinds of images in FLTK. This class keeps track of common image data such as the pixels, colormap, width, height, and depth. Virtual methods are used to provide type-specific image handling.

Each image possesses two (width, height) pairs:

1. The width and height of the raw image data are returned by [data_w\(\)](#) and [data_h\(\)](#). These values are set when the image is created and remain unchanged.
2. The width and height of the area filled by the image when it gets drawn are returned by [w\(\)](#) and [h\(\)](#). These values are equal to [data_w\(\)](#) and [data_h\(\)](#) when the image is created and can be changed by the [scale\(\)](#) member function.

Since the [Fl_Image](#) class does not support image drawing by itself, calling the [Fl_Image::draw\(\)](#) method results in a box with an X in it being drawn instead.

11.71.2 Constructor & Destructor Documentation

11.71.2.1 Fl_Image()

```
Fl_Image::Fl_Image (
    int W,
    int H,
    int D)
```

The constructor creates an empty image with the specified width, height, and depth.

The width and height are in pixels. The depth is 0 for bitmaps, 1 for pixmap (colormap) images, and 1 to 4 for color images.

11.71.3 Member Function Documentation

11.71.3.1 as_shared_image()

```
virtual class Fl\_Shared\_Image * Fl_Image::as_shared_image () [inline], [virtual]
```

Returns whether an image is an [Fl_Shared_Image](#) or not.

This virtual method returns a pointer to an [Fl_Shared_Image](#) if this object is an instance of [Fl_Shared_Image](#) or NULL if not. This can be used to detect if a given [Fl_Image](#) object is a shared image, i.e. derived from [Fl_Shared_Image](#).

Since

1.4.0

Reimplemented in [Fl_Shared_Image](#).

11.71.3.2 color_average()

```
void Fl_Image::color_average (
    Fl_Color c,
    float i) [virtual]
```

The `color_average()` method averages the colors in the image with the provided FLTK color value.

The first argument specifies the FLTK color to be used.

The second argument specifies the amount of the original image to combine with the color, so a value of 1.0 results in no color blend, and a value of 0.0 results in a constant image of the specified color.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented in [Fl_Anim_GIF_Image](#), [Fl_Pixmap](#), [Fl_RGB_Image](#), [Fl_Shared_Image](#), [Fl_SVG_Image](#), and [Fl_Tiled_Image](#).

11.71.3.3 copy() [1/2]

```
Fl_Image * Fl_Image::copy () const [inline]
```

Creates a copy of the image in the same size.

The new image should be released when you are done with it.

This does exactly the same as '[Fl_Image::copy\(int W, int H\) const](#)' where `W` and `H` are the width and height of the source image, respectively. This applies also to all subclasses of [Fl_Image](#) in the FLTK library.

The following two `copy()` calls are equivalent:

```
Fl_Image *img1 = new Fl_Image(...);
// ...
Fl_Image *img2 = img1->copy();
Fl_Image *img3 = img1->copy(img1->w(), img1->h());
```

For details see '[Fl_Image::copy\(int w, int h\) const](#)'.

See also

[Fl_Image::release\(\)](#)

Note

Since FLTK 1.4.0 this method is 'const'. If you derive your own class from [Fl_Image](#) or any subclass your overridden methods of '[Fl_Image::copy\(\) const](#)' and '[Fl_Image::copy\(int, int\) const](#)' **must** also be 'const' for inheritance to work properly. This is different than in FLTK 1.3.x and earlier where these methods have not been 'const'.

11.71.3.4 copy() [2/2]

```
Fl_Image * Fl_Image::copy (
    int W,
    int H) const [virtual]
```

Creates a resized copy of the image.

It is recommended not to call this member function to reduce the size of an image to the size of the area where this image will be drawn, and to use [Fl_Image::scale\(\)](#) instead.

The new image should be released when you are done with it.

Note: since FLTK 1.4.0 you can use [Fl_Image::release\(\)](#) for all types of images (i.e. all subclasses of [Fl_Image](#)) instead of operator `delete` for [Fl_Image](#)'s and [Fl_Image::release\(\)](#) for [Fl_Shared_Image](#)'s.

The new image data will be converted to the requested size. RGB images are resized using the algorithm set by [Fl_Image::RGB_scaling\(\)](#).

For the new image the following equations are true:

- `w() == data_w() == W`
- `h() == data_h() == H`

Parameters

in	<i>W,H</i>	Requested width and height of the new image

Note

The returned image can be safely cast to the same image type as that of the source image provided this type is one of [Fl_RGB_Image](#), [Fl_SVG_Image](#), [Fl_Pixmap](#), [Fl_Bitmap](#), [Fl_Tiled_Image](#), [Fl_Anim_GIF_Image](#) and [Fl_Shared_Image](#). Returned objects copied from images of other, derived, image classes belong to the parent class appearing in this list. For example, the copy of an [Fl_GIF_Image](#) is an object of class [Fl_Pixmap](#).

Since FLTK 1.4.0 this method is 'const'. If you derive your own class from [Fl_Image](#) or any subclass your overridden methods of '[Fl_Image::copy\(\) const](#)' and '[Fl_Image::copy\(int, int\) const](#)' **must** also be 'const' for inheritance to work properly. This is different than in FLTK 1.3.x and earlier where these methods have not been 'const'.

Reimplemented in [Fl_Anim_GIF_Image](#), [Fl_Bitmap](#), [Fl_Pixmap](#), [Fl_RGB_Image](#), [Fl_Shared_Image](#), [Fl_SVG_Image](#), and [Fl_Tiled_Image](#).

11.71.3.5 count()

```
int Fl_Image::count () const [inline]
```

Returns the number of data values associated with the image.

The value will be 0 for images with no associated data, 1 for bitmap and color images, and greater than 2 for pixmap images.

See also

[data\(\)](#)

11.71.3.6 d()

```
int Fl_Image::d () const [inline]
```

Returns the image depth.

The return value will be 0 for bitmaps, 1 for pixmaps, and 1 to 4 for color images.

11.71.3.7 data() [1/2]

```
const char *const * Fl_Image::data () const [inline]
```

Returns a pointer to the current image data array.

There can be 0, 1, or more pointers to actual image data in an image.

Use the [count\(\)](#) method to find the size of the data array. You must not dereference the [data\(\)](#) pointer if [count\(\)](#) equals zero.

Note

[data\(\)](#) **may** return NULL.

Example:

[Fl_RGB_Image](#) has exactly one pointer which points at the R, G, B [, A] data array of the image. The total size of this array depends on several attributes like [data_w\(\)](#), [data_h\(\)](#), [d\(\)](#) and [ld\(\)](#) and is basically [data_w\(\)](#) * [data_h\(\)](#) * [d\(\)](#) but there are exceptions if [ld\(\)](#) is non-zero: see description of [ld\(\)](#). Since FLTK 1.4.0 [w\(\)](#) and [h\(\)](#) are no longer significant for the image data size if [scale\(\)](#) has been called on the image to set a different display size.

Other image types have different numbers and types of data pointers which are implementation details and not documented here.

See also

[count\(\)](#), [w\(\)](#), [h\(\)](#), [data_w\(\)](#), [data_h\(\)](#), [d\(\)](#), [ld\(\)](#)

11.71.3.8 data() [2/2]

```
void Fl_Image::data (
    const char *const * p,
    int c) [inline], [protected]
```

Sets the current data pointer and count of pointers in the array.

There can be 0, 1, or more pointers to actual image data in an image.

See also

`const char* const* data\(\), count\(\), w\(\), h\(\), data\_w\(\), data\_h\(\), d\(\), ld\(\)`

11.71.3.9 [desaturate\(\)](#)

```
void Fl_Image::desaturate () [virtual]
```

The [desaturate\(\)](#) method converts an image to grayscale.

If the image contains an alpha channel (depth = 4), the alpha channel is preserved.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented in [Fl_Anim_GIF_Image](#), [Fl_Pixmap](#), [Fl_RGB_Image](#), [Fl_Shared_Image](#), [Fl_SVG_Image](#), and [Fl_Tiled_Image](#).

11.71.3.10 [draw\(\)](#) [1/2]

```
void Fl_Image::draw (
    int X,
    int Y) [inline]
```

Draws the image to the current drawing surface.

Parameters

X,Y	specify the upper-lefthand corner of the image.

11.71.3.11 [draw\(\)](#) [2/2]

```
void Fl_Image::draw (
    int X,
    int Y,
    int W,
    int H,
    int cx = 0,
    int cy = 0) [virtual]
```

Draws the image to the current drawing surface with a bounding box.

Arguments X, Y, W, H specify a bounding box for the image, with the origin (upper-left corner) of the image offset by the cx and cy arguments.

In other words: `fl_push_clip(X,Y,W,H)` is applied, the image is drawn with its upper-left corner at `X-cx, Y-cy` and its own width and height, `fl_pop_clip()` is applied.

Reimplemented in [Fl_Anim_GIF_Image](#), [Fl_Bitmap](#), [Fl_Pixmap](#), [Fl_RGB_Image](#), [Fl_Shared_Image](#), [Fl_SVG_Image](#), and [Fl_Tiled_Image](#).

11.71.3.12 [draw_empty\(\)](#)

```
void Fl_Image::draw_empty (
    int X,
    int Y) [protected]
```

The protected method [draw_empty\(\)](#) draws a box with an X in it.

It can be used to draw any image that lacks image data.

11.71.3.13 [draw_scaled\(\)](#)

```
int Fl_Image::draw_scaled (
    int X,
    int Y,
    int W,
    int H) [protected]
```

Draw the image to the current drawing surface rescaled to a given width and height.

Intended for internal use by the FLTK library.

Parameters

<i>X,Y</i>	position of the image's top-left
<i>W,H</i>	width and height for the drawn image

Returns

1

Deprecated Only for API compatibility with FLTK 1.3.4.

11.71.3.14 fail()

```
int Fl_Image::fail () const
```

Returns a value that is not 0 if there is currently no image available.

Example use:

```
// [...]
Fl_Box box(X, Y, W, H);
Fl_JPEG_Image jpg("/tmp/foo.jpg");
switch (jpg.fail()) {
case Fl_Image::ERR_NO_IMAGE:
case Fl_Image::ERR_FILE_ACCESS:
    fl_alert("/tmp/foo.jpg: %s", strerror(errno)); // shows actual os error to user
    exit(1);
case Fl_Image::ERR_FORMAT:
    fl_alert("/tmp/foo.jpg: couldn't decode image");
    exit(1);
}
box.image(jpg);
```

Returns

Image load failure if non-zero

Return values

0	the image was loaded successfully
<i>ERR_NO_IMAGE</i>	no image was found
<i>ERR_FILE_ACCESS</i>	there was a file access related error (errno should be set)
<i>ERR_FORMAT</i>	image decoding failed
<i>ERR_MEMORY_ACCESS</i>	image decoder tried to access memory outside of given memory block

11.71.3.15 h() [1/2]

```
int Fl_Image::h () const [inline]
```

Returns the current image drawing height in FLTK units.

The values of [h\(\)](#) and [data_h\(\)](#) are identical unless [scale\(\)](#) has been called after which they may differ.

11.71.3.16 h() [2/2]

```
void Fl_Image::h (
    int H) [inline], [protected]
```

Sets the height of the image data.

This protected function sets both image heights: the height of the image data returned by [data_h\(\)](#) and the image drawing height in FLTK units returned by [h\(\)](#).

11.71.3.17 inactive()

```
void Fl_Image::inactive () [inline]
```

The [inactive\(\)](#) method calls [color_average\(FL_BACKGROUND_COLOR, 0.33f\)](#) to produce an image that appears grayed out.

An internal copy is made of the original image before changes are applied, to avoid modifying the original image.

Note

The RGB color of [FL_BACKGROUND_COLOR](#) may change when the connection to the display is made. See [fl_open_display\(\)](#).

11.71.3.18 label() [1/2]

```
void Fl_Image::label (
    Fl_Menu_Item * m) [virtual]
```

This method is an obsolete way to set the image attribute of a menu item.

Deprecated Please use [Fl_Menu_Item::image\(\)](#) instead.

Reimplemented in [Fl_Bitmap](#), [Fl_Pixmap](#), and [Fl_RGB_Image](#).

11.71.3.19 label() [2/2]

```
void Fl_Image::label (
    Fl_Widget * widget) [virtual]
```

This method is an obsolete way to set the image attribute of a widget or menu item.

Deprecated Please use [Fl_Widget::image\(\)](#) or [Fl_Widget::deimage\(\)](#) instead.

Reimplemented in [Fl_Bitmap](#), [Fl_Pixmap](#), and [Fl_RGB_Image](#).

11.71.3.20 ld() [1/2]

```
int Fl_Image::ld () const [inline]
```

Returns the current line data size in bytes.

See also

[ld\(int\)](#)

11.71.3.21 ld() [2/2]

```
void Fl_Image::ld (
    int LD) [inline], [protected]
```

Sets the current line data size in bytes.

Color images may contain extra data (padding) that is included after every line of color image data and is normally not present.

If LD is zero, then line data size is assumed to be [data_w\(\)](#) * [d\(\)](#) bytes.

If LD is non-zero, then it must be positive and larger than [data_w\(\)](#) * [d\(\)](#) to account for the extra data per line.

11.71.3.22 release()

```
virtual void Fl_Image::release () [inline], [virtual]
```

Releases an [Fl_Image](#) - the same as 'delete this'.

This virtual method is for almost all image classes the same as calling

```
delete image;
```

where image is an [Fl_Image](#) * pointer.

However, for subclass [Fl_Shared_Image](#) and its subclasses this virtual method is reimplemented and maintains shared images.

This virtual method makes it possible to destroy all image types in the same way by calling

```
image->release();
```

Reasoning: If you have an '[Fl_Image](#) *' base class pointer and don't know if the object is one of the class [Fl_Shared_Image](#) or any other subclass of [Fl_Image](#) (for instance [Fl_RGB_Image](#)) then you can't just use operator delete since this is not appropriate for [Fl_Shared_Image](#) objects.

The virtual method [release\(\)](#) handles this properly.

Since

1.4.0 in the base class [Fl_Image](#) and virtual in [Fl_Shared_Image](#)

Reimplemented in [Fl_Shared_Image](#).

11.71.3.23 RGB_scaling()

```
void Fl_Image::RGB_scaling (
    Fl\_RGB\_Scaling method) [static]
```

Sets the RGB image scaling method used for copy(int, int).

Applies to all RGB images, defaults to FL_RGB_SCALING_NEAREST.

11.71.3.24 scale()

```
void Fl_Image::scale (
    int width,
    int height,
    int proportional = 1,
    int can_expand = 0) [virtual]
```

Sets the drawing size of the image.

This function controls the values returned by member functions [w\(\)](#) and [h\(\)](#) which in turn control how the image is drawn: the full image data (whose size is given by [data_w\(\)](#) and [data_h\(\)](#)) are drawn scaled to an area of the drawing surface sized at [w\(\)](#) x [h\(\)](#) FLTK units. This can make a difference if the drawing surface has more than 1 pixel per FLTK unit because the image can be drawn at the full resolution of the drawing surface. Examples of such drawing surfaces: HiDPI displays, laser printers, PostScript files, PDF printers.

Parameters

<i>width,height</i>	maximum values, in FLTK units, that w() and h() should return
<i>proportional</i>	if not null, keep the values returned by w() and h() proportional to data_w() and data_h()
<i>can_expand</i>	if null, the values returned by w() and h() will not be larger than data_w() and data_h() , respectively

Note

This function generally changes the values returned by the [w\(\)](#) and [h\(\)](#) member functions. In contrast, the values returned by [data_w\(\)](#) and [data_h\(\)](#) remain unchanged.

Version

1.4 (1.3.4 and FL_ABI_VERSION for [Fl_Shared_Image](#) only)

Example code: scale an image to fit in a box

```
Fl\_Box *b = ... // a box
Fl\_Image *img = new Fl\_PNG\_Image("/path/to/picture.png"); // read a picture file
// set the drawing size of the image to the size of the box keeping its aspect ratio
img->scale(b->w(), b->h());
b->image(img); // use the image as the box image
```

11.71.3.25 scaling_algorithm()

```
static void Fl_Image::scaling_algorithm (
    Fl\_RGB\_Scaling algorithm) [inline], [static]
```

Sets what algorithm is used when resizing a source image to draw it.

The default algorithm is FL_RGB_SCALING_BILINEAR. Drawing an [Fl_Image](#) is sometimes performed by first resizing the source image and then drawing the resized copy. This occurs, e.g., when drawing to screen under X11 without Xrender support after having called [scale\(\)](#). This function controls what method is used when the image to be resized is an [Fl_RGB_Image](#).

Version

1.4

11.71.3.26 uncache()

```
void Fl_Image::uncache () [virtual]
```

If the image has been cached for display, delete the cache data.

This allows you to change the data used for the image and then redraw it without recreating an image object.

Reimplemented in [Fl_Anim_GIF_Image](#), [Fl_Bitmap](#), [Fl_Pixmap](#), [Fl_RGB_Image](#), and [Fl_Shared_Image](#).

11.71.3.27 w() [1/2]

```
int Fl_Image::w () const [inline]
```

Returns the current image drawing width in FLTK units.

The values of [w\(\)](#) and [data_w\(\)](#) are identical unless [scale\(\)](#) has been called after which they may differ.

11.71.3.28 w() [2/2]

```
void Fl_Image::w (
    int W) [inline], [protected]
```

Sets the width of the image data.

This protected function sets both image widths: the width of the image data returned by [data_w\(\)](#) and the image drawing width in FLTK units returned by [w\(\)](#).

The documentation for this class was generated from the following files:

- [Fl_Image.H](#)
- [Fl_Image.cxx](#)

11.72 Fl_Image_Reader Class Reference**Public Member Functions**

- int **error** () const
- const char * **name** () const
- int **open** (const char *filename)
- int **open** (const char *imagename, const unsigned char *data)
- int **open** (const char *imagename, const unsigned char *data, const size_t datasize)
- unsigned char **read_byte** ()
- unsigned int **read_dword** ()
- int **read_long** ()
- unsigned short **read_word** ()
- void **seek** (unsigned int n)
- void **skip** (unsigned int n)
- long **tell** () const

The documentation for this class was generated from the following files:

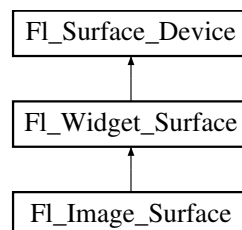
- [Fl_Image_Reader.h](#)
- [Fl_Image_Reader.cxx](#)

11.73 Fl_Image_Surface Class Reference

Directs all graphics requests to an [Fl_Image](#).

```
#include <Fl_Image_Surface.H>
```

Inheritance diagram for [Fl_Image_Surface](#):



Public Member Functions

- [FI_Image_Surface](#) (int w, int h, int high_res=0, [FI_Offscreen](#) off=0)
Constructor with optional high resolution.
- [FI_Shared_Image](#) * [highres_image](#) ()
Returns a possibly high resolution image made of all drawings sent to the [FI_Image_Surface](#) object.
- [FI_RGB_Image](#) * [image](#) ()
Returns a depth-3 image made of all drawings sent to the [FI_Image_Surface](#) object.
- bool [is_current](#) () [FL_OVERRIDE](#)
Is this surface the current drawing surface?
- void [mask](#) (const [FI_RGB_Image](#) *)
Defines a mask applied to drawings made after use of this function.
- [FI_Offscreen](#) [offscreen](#) ()
Returns the [FI_Offscreen](#) object associated to the image surface.
- void [origin](#) (int *x, int *y) [FL_OVERRIDE](#)
Computes the coordinates of the current origin of graphics functions.
- void [origin](#) (int x, int y) [FL_OVERRIDE](#)
Sets the position of the origin of graphics in the drawable part of the drawing surface.
- int [printable_rect](#) (int *w, int *h) [FL_OVERRIDE](#)
Computes the width and height of the drawable area of the drawing surface.
- void [rescale](#) ()
Adapts the [FI_Image_Surface](#) object to the new value of the GUI scale factor.
- void [set_current](#) () [FL_OVERRIDE](#)
Make this surface the current drawing surface.
- ~[FI_Image_Surface](#) ()
The destructor.

Public Member Functions inherited from [FI_Widget_Surface](#)

- void [draw](#) ([FI_Widget](#) *widget, int delta_x=0, int delta_y=0)
Draws the widget on the drawing surface.
- void [draw_decorated_window](#) ([FI_Window](#) *win, int x_offset=0, int y_offset=0)
Draws a window with its title bar and frame if any.
- void [print_window_part](#) ([FI_Window](#) *win, int x, int y, int w, int h, int delta_x=0, int delta_y=0)
Draws a rectangular part of an on-screen window.

Public Member Functions inherited from [FI_Surface_Device](#)

- [FI_Graphics_Driver](#) * **driver** ()
Returns the graphics driver of this drawing surface.
- virtual ~[FI_Surface_Device](#) ()
The destructor.

Protected Member Functions

- void [translate](#) (int x, int y) [FL_OVERRIDE](#)
Translates the current graphics origin accounting for the current rotation.
- void [untranslate](#) () [FL_OVERRIDE](#)
Undoes the effect of a previous [translate\(\)](#) call.

Protected Member Functions inherited from [FI_Widget_Surface](#)

- [FI_Widget_Surface](#) ([FI_Graphics_Driver](#) *d)
The constructor.

Protected Member Functions inherited from [Fl_Surface_Device](#)

- void **driver** (Fl_Graphics_Driver *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void **end_current** ()
FLTK calls this each time a surface ceases to be the current drawing surface.
- **Fl_Surface_Device** (Fl_Graphics_Driver *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

Friends

- class **Fl_Graphics_Driver**

Additional Inherited Members

Static Public Member Functions inherited from [Fl_Surface_Device](#)

- static [Fl_Surface_Device](#) * **pop_current** ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void **push_current** ([Fl_Surface_Device](#) *new_current)
Pushes new_current on top of the stack of current drawing surfaces, and makes it current.
- static [Fl_Surface_Device](#) * **surface** ()
The current drawing surface.

Protected Attributes inherited from [Fl_Widget_Surface](#)

- int **x_offset**
horizontal offset to the origin of graphics coordinates
- int **y_offset**
vertical offset to the origin of graphics coordinates

11.73.1 Detailed Description

Directs all graphics requests to an [Fl_Image](#).

After creation of an [Fl_Image_Surface](#) object, make it the current drawing surface calling [Fl_Surface_Device::push_current\(\)](#), and all subsequent graphics requests will be recorded in the image. It's possible to draw widgets (using [Fl_Image_Surface::draw\(\)](#)) or to use any of the [Drawing functions](#) or the [Color & Font functions](#). Finally, call [image\(\)](#) on the object to obtain a newly allocated [Fl_RGB_Image](#) object. [Fl_Gl_Window](#) objects can be drawn in the image as well.

Usage example:

```
// this is the widget that you want to draw into an image
Fl_Widget *g = ...;

// create an Fl_Image_Surface object
Fl_Image_Surface *image_surface = new Fl_Image_Surface(g->w(), g->h());

// direct all further graphics requests to the image
Fl_Surface_Device::push_current(image_surface);

// draw a white background
fl_color(FL_WHITE);
fl_rectf(0, 0, g->w(), g->h());

// draw the g widget in the image
image_surface->draw(g);

// get the resulting image
Fl_RGB_Image* image = image_surface->image();

// direct graphics requests back to their previous destination
Fl_Surface_Device::pop_current();

// delete the image_surface object, but not the image itself
delete image_surface;
```

11.73.2 Constructor & Destructor Documentation

11.73.2.1 Fl_Image_Surface()

```
Fl_Image_Surface::Fl_Image_Surface (
    int w,
    int h,
    int high_res = 0,
    Fl_Offscreen off = 0)
```

Constructor with optional high resolution.

Parameters

<i>w,h</i>	Width and height of the resulting image. The value of the <code>high_res</code> parameter controls whether <code>w</code> and <code>h</code> are interpreted as pixels or FLTK units.
<i>high_res</i>	If zero, the created image surface is sized at <code>w x h</code> pixels. If non-zero, the pixel size of the created image surface depends on the value of the display scale factor (see Fl::screen_scale(int)): the resulting image has the same number of pixels as an area of the display of size <code>w x h</code> expressed in FLTK units.
<i>off</i>	If not null, the image surface is constructed around a pre-existing <code>Fl_Offscreen</code> . The caller is responsible for both construction and destruction of this <code>Fl_Offscreen</code> object. Is mostly intended for internal use by FLTK.

Version

1.3.4 (1.3.3 without the `highres` parameter)

11.73.3 Member Function Documentation

11.73.3.1 highres_image()

```
Fl_Shared_Image * Fl_Image_Surface::highres_image ()
```

Returns a possibly high resolution image made of all drawings sent to the `Fl_Image_Surface` object.

The `Fl_Image_Surface` object should have been constructed with `Fl_Image_Surface(W, H, 1)`. The returned `Fl_Shared_Image` object is scaled to a size of `WxH` FLTK units and may have a pixel size larger than these values. The returned object should be deallocated with `Fl_Shared_Image::release()` after use.

Deprecated Use `image()` instead.

Version

1.4 (1.3.4 for MacOS platform only)

11.73.3.2 image()

```
Fl_RGB_Image * Fl_Image_Surface::image ()
```

Returns a depth-3 image made of all drawings sent to the `Fl_Image_Surface` object.

The returned object contains its own copy of the RGB data; the caller is responsible for deleting it.

See also

```
Fl_Image_Surface::mask(Fl_RGB_Image*)
```

11.73.3.3 is_current()

```
bool Fl_Image_Surface::is_current () [virtual]
```

Is this surface the current drawing surface?

Reimplemented from [Fl_Surface_Device](#).

11.73.3.4 mask()

```
void Fl_Image_Surface::mask (
    const Fl_RGB_Image * mask)
```

Defines a mask applied to drawings made after use of this function.

The mask is an [Fl_RGB_Image](#) made of a white scene drawn on a solid black background; the drawable part of the image surface is reduced to the white areas of the mask after this member function gets called. If necessary, the `mask` image is internally replaced by a copy resized to the surface's pixel size. Overall, the image returned by [Fl_Image_Surface::image\(\)](#) contains all drawings made until the `mask()` method assigned a mask, at which point subsequent drawing operations to the image surface were passed through the white areas of the mask. On some platforms, shades of gray in the mask image control the blending of foreground and background pixels; mask pixels closer in color to white produce image pixels closer to the image surface pixel, those closer to black produce image pixels closer to what the image surface pixel was before the call to `mask()`.

The mask is easily constructed using an [Fl_Image_Surface](#) object, drawing white areas on a black background there, and calling [Fl_Image_Surface::image\(\)](#).

Parameters

<i>mask</i>	A depth-3 image determining the drawable areas of the image surface. The <code>mask</code> object is not used after return from this member function.

Note

- The image surface must not be the current drawing surface when this function gets called.
- The mask can have any size but is best when it has the size of the image surface.
- It's possible to use several masks in succession on the same image surface provided member function [Fl_Image_Surface::image\(\)](#) is called between successive calls to [Fl_Image_Surface::mask\(const Fl_RGB_Image*\)](#).

Example of procedure to construct a masked image:

```
int W = ..., H = ...; // width and height of the image under construction
Fl_Image_Surface *surf = new Fl_Image_Surface(W, H, 1);
// first, construct the mask
Fl_Surface_Device::push_current(surf);
fl_color(FL_BLACK); // draw a black background
fl_rectf(0, 0, W, H);
fl_color(FL_WHITE); // next, draw in white what the mask should not filter out
fl_pie(0, 0, W, H, 0, 360); // here, an ellipse with axes lengths WxH
Fl_RGB_Image *mask = surf->image(); // get the mask
// second, draw the image background
fl_color(FL_YELLOW); // here, draw a yellow background
fl_rectf(0, 0, W, H);
// third, apply the mask
Fl_Surface_Device::pop_current();
surf->mask(mask);
delete mask; // the mask image can be safely deleted at this point
Fl_Surface_Device::push_current(surf);
// fourth, draw the image foreground, part of which will be filtered out by the mask
surf->draw(widget, 0, 0); // here the foreground is a drawn widget
// fifth, get the final result, masked_image, as a depth-3 Fl_RGB_Image
Fl_RGB_Image *masked_image = surf->image();
// Only the part of the foreground, here a drawn widget, that has not been
// filtered out by the mask, here the white ellipse, is in masked_image;
// the background, here solid yellow, shows up in the remaining areas of masked_image.
Fl_Surface_Device::pop_current();
delete surf;
```

Since

1.4.0

11.73.3.5 offscreen()

```
Fl_Offscreen Fl_Image_Surface::offscreen ()
```

Returns the [Fl_Offscreen](#) object associated to the image surface.

The returned [Fl_Offscreen](#) object is deleted when the [Fl_Image_Surface](#) object is deleted, unless the [Fl_Image_Surface](#) was constructed with non-null [Fl_Offscreen](#) argument.

11.73.3.6 origin() [1/2]

```
void Fl_Image_Surface::origin (
    int * x,
    int * y) [virtual]
```

Computes the coordinates of the current origin of graphics functions.

Parameters

out	x,y	If non-null, *x and *y are set to the horizontal and vertical coordinates of the graphics origin.

Reimplemented from [Fl_Widget_Surface](#).

11.73.3.7 origin() [2/2]

```
void Fl_Image_Surface::origin (
    int x,
    int y) [virtual]
```

Sets the position of the origin of graphics in the drawable part of the drawing surface.

Arguments should be expressed relatively to the result of a previous [printable_rect\(\)](#) call. That is, `printable_rect(&w, &h); origin(w/2, 0);` sets the graphics origin at the top center of the drawable area. Successive [origin\(\)](#) calls don't combine their effects. [Origin\(\)](#) calls are not affected by [rotate\(\)](#) calls (for classes derived from [Fl_Paged_Device](#)).

Parameters

in	x,y	Horizontal and vertical positions in the drawing surface of the desired origin of graphics.

Reimplemented from [Fl_Widget_Surface](#).

11.73.3.8 printable_rect()

```
int Fl_Image_Surface::printable_rect (
    int * w,
    int * h) [virtual]
```

Computes the width and height of the drawable area of the drawing surface.

Values are in the same unit as that used by FLTK drawing functions and are unchanged by calls to [origin\(\)](#). If the object is derived from class [Fl_Paged_Device](#), values account for the user-selected paper type and print orientation and are changed by [scale\(\)](#) calls.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Widget_Surface](#).

11.73.3.9 rescale()

```
void Fl_Image_Surface::rescale ()
```

Adapts the [Fl_Image_Surface](#) object to the new value of the GUI scale factor.

The [Fl_Image_Surface](#) object must not be the current drawing surface. This function is useful only for an object constructed with non-zero `high_res` parameter.

Version

1.4

11.73.3.10 set_current()

```
void Fl_Image_Surface::set_current (
    void ) [virtual]
```

Make this surface the current drawing surface.

This surface will receive all future graphics requests.

Since FLTK 1.4.0 the preferred API to change the current drawing surface is [Fl_Surface_Device::push_current\(\)](#) / [Fl_Surface_Device::pop_current\(\)](#).

Note

It is recommended to use this function only as follows :

- The current drawing surface is the display;
- make current another surface, e.g., an [Fl_Printer](#) or an [Fl_Image_Surface](#) object, calling [set_current\(\)](#) on this object;
- draw to that surface;
- make the display current again with [Fl_Display_Device::display_device\(\)->set_current\(\)](#);
don't do any other call to [set_current\(\)](#) before this one.

Other scenarios of drawing surface changes should be performed via [Fl_Surface_Device::push_current\(\)](#) and [Fl_Surface_Device::pop_current\(\)](#).

Reimplemented from [Fl_Surface_Device](#).

11.73.3.11 translate()

```
void Fl_Image_Surface::translate (
    int x,
    int y) [protected], [virtual]
```

Translates the current graphics origin accounting for the current rotation.

Each [translate\(\)](#) call must be matched by an [untranslate\(\)](#) call. Successive [translate\(\)](#) calls add up their effects.

Reimplemented from [Fl_Widget_Surface](#).

11.73.3.12 untranslate()

```
void Fl_Image_Surface::untranslate (
    void ) [protected], [virtual]
```

Undoes the effect of a previous [translate\(\)](#) call.

Reimplemented from [Fl_Widget_Surface](#).

The documentation for this class was generated from the following files:

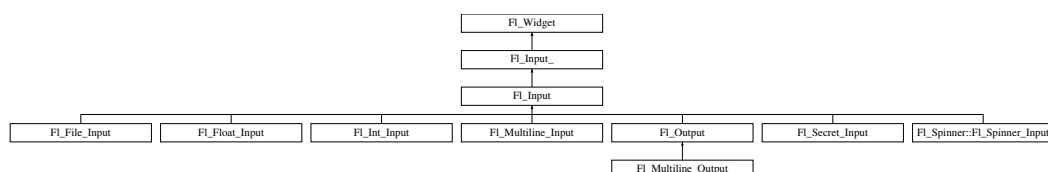
- [Fl_Image_Surface.H](#)
- [Fl_Image_Surface.cxx](#)

11.74 Fl_Input Class Reference

This is the FLTK text input widget.

```
#include <Fl_Input.H>
```

Inheritance diagram for [Fl_Input](#):



Public Member Functions

- [FL_Input](#) (int, int, int, int, const char *=0)
Creates a new [FL_Input](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [FL_Input_](#)

- int [append](#) (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool [can_redo](#) () const
Check if there is a redo action available.
- bool [can_undo](#) () const
Check if the last operation can be undone.
- int [copy](#) (int clipboard)
Put the current selection into the clipboard.
- int [copy_cuts](#) ()
Copies the yank buffer to the clipboard.
- [FL_Color](#) [cursor_color](#) () const
Gets the color of the cursor.
- void [cursor_color](#) ([FL_Color](#) n)
Sets the color of the cursor.
- int [cut](#) ()
Deletes the current selection.
- int [cut](#) (int a, int b)
Deletes all characters between index a and b.
- int [cut](#) (int n)
Deletes the next n bytes rounded to characters before or after the cursor.
- double [dvalue](#) () const
Returns the widget text interpreted as a floating point number.
- [FL_Input_](#) (int, int, int, int, const char *=0)
Creates a new [FL_Input_](#) widget.
- unsigned int [index](#) (int i) const
Returns the character at index i.
- int [input_type](#) () const
Gets the input field type.
- void [input_type](#) (int t)
Sets the input field type.
- int [insert](#) (const char *t, int l=0)
Inserts text at the cursor position.
- int [insert_position](#) () const
Gets the position of the text cursor.
- int [insert_position](#) (int p)
Sets the cursor position and mark.
- int [insert_position](#) (int p, int m)
Sets the index for the cursor and mark.
- int [ivalue](#) () const
Returns the widget text interpreted as a signed integer.
- int [mark](#) () const
Gets the current selection mark.
- int [mark](#) (int m)

- Sets the current selection mark.*
- `int maximum_size () const`
 - Gets the maximum length of the input field in characters.*
- `void maximum_size (int m)`
 - Sets the maximum length of the input field in characters.*
- `int position () const`
- `int position (int p)`
- `int position (int p, int m)`
- `int readonly () const`
 - Gets the read-only state of the input field.*
- `void readonly (int b)`
 - Sets the read-only state of the input field.*
- `int redo ()`
 - Redo previous undo operation.*
- `int replace (int b, int e, const char *text, int len=0)`
 - Deletes text from *b* to *e* and inserts the new string *text*.*
- `void resize (int, int, int, int) FL_OVERRIDE`
 - Changes the size of the widget.*
- `int shortcut () const`
 - Return the shortcut key associated with this widget.*
- `void shortcut (int s)`
 - Sets the shortcut key associated with this widget.*
- `int size () const`
 - Returns the number of bytes in *value()*.*
- `void size (int W, int H)`
 - Sets the width and height of this widget.*
- `int static_value (const char *)`
 - Changes the widget text.*
- `int static_value (const char *, int)`
 - Changes the widget text.*
- `int tab_nav () const`
 - Gets whether the Tab key causes focus navigation in multiline input fields or not.*
- `void tab_nav (int val)`
 - Sets whether the Tab key does focus navigation, or inserts tab characters into *FL_Multiline_Input*.*
- `FL_Color textcolor () const`
 - Gets the color of the text in the input field.*
- `void textcolor (FL_Color n)`
 - Sets the color of the text in the input field.*
- `FL_Font textfont () const`
 - Gets the font of the text in the input field.*
- `void textfont (FL_Font s)`
 - Sets the font of the text in the input field.*
- `FL_Fontsize textsize () const`
 - Gets the size of the text in the input field.*
- `void textsize (FL_Fontsize s)`
 - Sets the size of the text in the input field.*
- `int undo ()`
 - Undoes previous changes to the text buffer.*
- `const char * value () const`
 - Returns the text displayed in the widget.*
- `int value (const char *)`

- Changes the widget text.*

 - int [value](#) (const char *, int)

Changes the widget text.
- int [value](#) (double value)

Changes the widget text to a floating point number ("%g").
- int [value](#) (int value)

Changes the widget text to a signed integer number.
- int [wrap](#) () const

Gets the word wrapping state of the input field.
- void [wrap](#) (int b)

Sets the word wrapping state of the input field.
- [~Fl_Input_](#) ()

Destroys the widget.

Public Member Functions inherited from [Fl_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()

Activates the widget.
- unsigned int [active](#) () const

Returns whether the widget is active.
- int [active_r](#) () const

Returns whether the widget and all of its parents are active.
- [Fl_Align](#) [align](#) () const

Gets the label alignment.
- void [align](#) ([Fl_Align](#) alignment)

Sets the label alignment.
- long [argument](#) () const

Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)

Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()

Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [Fl_Group](#) * [as_group](#) ()

Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).
- virtual [Fl_Group](#) const * [as_group](#) () const
- virtual [Fl_Window](#) * [as_window](#) ()

Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- virtual [Fl_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([Fl_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype](#) [box](#) () const

Gets the box type of the widget.

- void `box` (`FI_Boxtype` new_box)
Sets the box type for the widget.
- `FI_Callback_p` `callback` () const
Gets the current callback function for the widget.
- void `callback` (`FI_Callback` *cb)
Sets the current callback function for the widget.
- void `callback` (`FI_Callback` *cb, `FI_Callback_User_Data` *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void `callback` (`FI_Callback` *cb, void *p)
Sets the current callback function and data for the widget.
- void `callback` (`FI_Callback0` *cb)
Sets the current callback function for the widget.
- void `callback` (`FI_Callback1` *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int `changed` () const
Checks if the widget value changed since the last callback.
- void `clear_active` ()
Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()
Marks the value of the widget as unchanged.
- void `clear_damage` (`uchar` c=0)
Clears or sets the damage flags.
- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FI_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FI_Color` bg)
Sets the background color of the widget.
- void `color` (`FI_Color` bg, `FI_Color` sel)
Sets the background and selection color of the widget.
- `FI_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FI_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

- Internal use only.*

 - void `deactivate` ()
 - Deactivates the widget.*
 - `Fl_Image * deimage` ()
 - Gets the image that is used as part of the widget label when in the inactive state.*
 - const `Fl_Image * deimage` () const
 - Gets the image that is used as part of the widget label when in the inactive state.*
 - void `deimage` (`Fl_Image &img`)
 - Sets the image to use as part of the widget label when in the inactive state.*
 - void `deimage` (`Fl_Image *img`)
 - Sets the image to use as part of the widget label when in the inactive state.*
 - int `deimage_bound` () const
 - Returns whether the inactive image is managed by the widget.*
 - void `do_callback` (`Fl_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
 - Calls the widget callback function with default arguments.*
 - void `do_callback` (`Fl_Widget *widget`, long arg, `Fl_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
 - Calls the widget callback function with arbitrary arguments.*
 - void `do_callback` (`Fl_Widget *widget`, void *arg=0, `Fl_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
 - Calls the widget callback function with arbitrary arguments.*
 - void `draw_label` (int, int, int, int, `Fl_Align`) const
 - Draws the label in an arbitrary bounding box with an arbitrary alignment.*
 - int `h` () const
 - Gets the widget height.*
 - virtual void `hide` ()
 - Makes a widget invisible.*
 - int `horizontal_label_margin` ()
 - Get the spacing between the label and the horizontal edge of the widget.*
 - void `horizontal_label_margin` (int px)
 - Set the spacing between the label and the horizontal edge of the widget.*
 - `Fl_Image * image` ()
 - Gets the image that is used as part of the widget label when in the active state.*
 - const `Fl_Image * image` () const
 - Gets the image that is used as part of the widget label when in the active state.*
 - void `image` (`Fl_Image &img`)
 - Sets the image to use as part of the widget label when in the active state.*
 - void `image` (`Fl_Image *img`)
 - Sets the image to use as part of the widget label when in the active state.*
 - int `image_bound` () const
 - Returns whether the image is managed by the widget.*
 - int `inside` (const `Fl_Widget *wgt`) const
 - Checks if this widget is a child of wgt.*
 - int `is_label_copied` () const
 - Returns whether the current label was assigned with `copy_label()`.*
 - const char * `label` () const
 - Gets the current label text.*
 - void `label` (const char *text)
 - Sets the current label pointer.*
 - void `label` (`Fl_Labeltype` a, const char *b)
 - Shortcut to set the label text and type in one call.*
 - int `label_image_spacing` ()
 - Return the gap size between the label and the image.*

- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize labelsiz](#)e () const
Gets the font size in pixels.
- void [labelsiz](#)e ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group * parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [FL_Color selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

Gets the current tooltip text.
- void [tooltip](#) (const char *text)

Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const

Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const

Gets the widget type.
- void [type](#) ([uchar](#) t)

Sets the widget type.
- int [use_accents_menu](#) ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const

Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)

Sets the user data for this widget.
- void [user_data](#) (void *v)

Sets the user data for this widget.
- int [vertical_label_margin](#) ()

Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const

Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const

Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)

Modifies keyboard focus navigation.
- int [visible_r](#) () const

Returns whether a widget and all its parents are visible.
- int [w](#) () const

Gets the widget width.
- [Fl_When](#) [when](#) () const

Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)

Sets the flags used to decide when a callback is called.

- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual [~Fl_Widget](#) ()
Destroys the widget.

Static Public Attributes

- static const char * [copy_menu_text](#) = "Copy"
[this text may be customized at run-time]
- static const char * [cut_menu_text](#) = "Cut"
[this text may be customized at run-time]
- static const char * [paste_menu_text](#) = "Paste"
[this text may be customized at run-time]

Protected Member Functions

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- int [handle_key](#) ()
Handles a keystroke.
- int [handle_rmb](#) ()
Handle right mouse button down events.

Protected Member Functions inherited from [Fl_Input](#)

- int [apply_undo](#) ()
Apply the current undo/redo operation.
- void [drawtext](#) (int, int, int, int)
Draws the text in the passed bounding box.
- void [drawtext](#) (int, int, int, int, bool draw_active)
Draws the text in the passed bounding box.
- void [handle_mouse](#) (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int [handletext](#) (int e, int, int, int, int)
Handles all kinds of text field related events.
- int [line_end](#) (int i) const
Finds the end of a line.
- int [line_start](#) (int i) const
Finds the start of a line.
- int [linesPerPage](#) ()
- void [maybe_do_callback](#) ([Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- int [up_down_position](#) (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int [word_end](#) (int i) const
Finds the end of a word.
- int [word_start](#) (int i) const
Finds the start of a word.
- int [xscroll](#) () const
- int [yscroll](#) () const
- void [yscroll](#) (int yOffset)

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Friends

- class [FI_Cocoa_Screen_Driver](#)
- class [FI_Screen_Driver](#)

Additional Inherited Members

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
 ,
[OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
 ,
[MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
 ,
[GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
 = 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.74.1 Detailed Description

This is the FLTK text input widget.

It displays a single line of text and lets the user edit it. Normally it is drawn with an inset box and a white background. The text may contain any characters, and will correctly display any UTF text, using ^X notation for unprintable control characters. It assumes the font can draw any characters of the used scripts, which is true for standard fonts under Windows and Mac OS X. Characters can be input using the keyboard or the character palette/map. Character composition is done using dead keys and/or a compose key as defined by the operating system.

Table 11.304 Keyboard and mouse bindings.

Mouse button 1	Moves the cursor to this point. Drag selects characters. Double click selects words. Triple click selects all line. Shift+click extends the selection. When you select text it is automatically copied to the selection buffer.
Mouse button 2	Insert the selection buffer at the point clicked. You can also select a region and replace it with the selection buffer by selecting the region with mouse button 2.
Mouse button 3	Currently acts like button 1.
Backspace	Deletes one character to the left, or deletes the selected region.
Delete	Deletes one character to the right, or deletes the selected region. Combine with Shift for equivalent of ^X (copy+cut).
Enter	May cause the callback, see when() .

Table 11.305 Platform specific keyboard bindings.

Windows/Linux	Mac	Function
^A	Command-A	Selects all text in the widget.
^C	Command-C	Copy the current selection to the clipboard.
^I	^I	Insert a tab.
^J	^J	Insert a Line Feed. (Similar to literal 'Enter' character)
^L	^L	Insert a Form Feed.
^M	^M	Insert a Carriage Return.

^V, Shift-Insert	Command-V	Paste the clipboard. (Macs keyboards don't have "Insert" keys, but if they did, Shift-Insert would work)
^X, Shift-Delete	Command-X, Shift-Delete	Cut. Copy the selection to the clipboard and delete it. (If there's no selection, Shift-Delete acts like Delete)
^Z	Command-Z	Undo. This is a single-level undo mechanism, but all adjacent deletions and insertions are concatenated into a single "undo". Often this will undo a lot more than you expected.
Shift-^Z	Shift-Command-Z	Redo. Currently same behavior as ^Z. Reserved for future multilevel undo/redo.
Arrow Keys	Arrow Keys	Standard cursor movement. Can be combined with Shift to extend selection.
Home	Command-Up, Command-Left	Move to start of line. Can be combined with Shift to extend selection.
End	Command-Down, Command-Right	Move to end of line. Can be combined with Shift to extend selection.
Ctrl-Home	Command-Up, Command-PgUp, Ctrl-Left	Move to top of document/field. In single line input, moves to start of line. In multiline input, moves to start of top line. Can be combined with Shift to extend selection.
Ctrl-End	Command-End, Command-PgDn, Ctrl-Right	Move to bottom of document/field. In single line input, moves to end of line. In multiline input, moves to end of last line. Can be combined with Shift to extend selection.
Ctrl-Left	Alt-Left	Word left. Can be combined with Shift to extend selection.
Ctrl-Right	Alt-Right	Word right. Can be combined with Shift to extend selection.
Ctrl-Backspace	Alt-Backspace	Delete word left.
Ctrl-Delete	Alt-Delete	Delete word right.

11.74.2 Constructor & Destructor Documentation

11.74.2.1 Fl_Input()

```
Fl_Input::Fl_Input (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Input](#) widget using the given position, size, and label string. The default boxtype is FL_DOWN_BOX.

11.74.3 Member Function Documentation

11.74.3.1 draw()

```
void Fl_Input::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.74.3.2 handle()

```
int Fl_Input::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

<code>in</code>	<code>event</code>	the kind of event received

Return values

<code>0</code>	if the event was not used or understood
<code>1</code>	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Secret_Input](#), and [Fl_Spinner::Fl_Spinner_Input](#).

11.74.3.3 handle_key()

```
int Fl_Input::handle_key () [protected]
```

Handles a keystroke.

This `protected` method handles a keystroke in an [Fl_Input](#) or derived class. It handles compose key sequences and can also be used e.g. in [Fl_Multiline_Input](#), [Fl_Float_Input](#) and several more derived classes.

The method first checks in [Fl::compose](#) if the keystroke is a text entry or a control key. If it is text, the method inserts the composed characters into the input field, taking into account the input type (e.g., numeric fields).

If the keystroke is a control key as determined by [Fl::compose](#), the method handles key combinations for Insert, Enter, and Tab depending on the widget's [input_type\(\)](#).

The method then checks for Ctrl key combinations, such as Ctrl-A, Ctrl-C, Ctrl-V, Ctrl-X, and Ctrl-Z, which are commonly used for select all, copy, paste, cut, and undo operations.

Finally, the method checks for ASCII control characters, such as Ctrl-H, Ctrl-I, Ctrl-J, Ctrl-L, and Ctrl-M, which can be used to insert literal control characters into the input field.

If none of the above cases match, the method returns 0, indicating that the keystroke was not handled.

Returns

1 if the keystroke is handled by us, 0 if not.

11.74.3.4 handle_rmb()

```
int Fl_Input::handle_rmb () [protected]
```

Handle right mouse button down events.

Returns

1

The documentation for this class was generated from the following files:

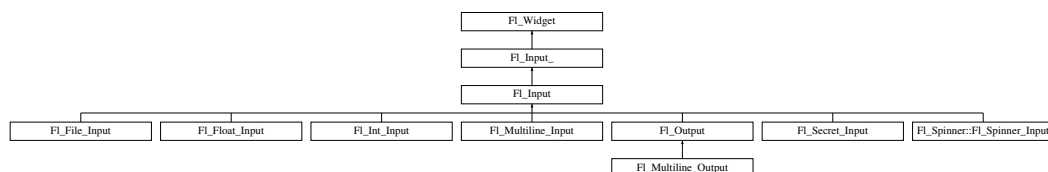
- Fl_Input.H
- Fl_Input.cxx

11.75 Fl_Input_ Class Reference

This class provides a low-overhead text input field.

```
#include <Fl_Input_.H>
```

Inheritance diagram for Fl_Input_:



Public Member Functions

- int [append](#) (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool [can_redo](#) () const
Check if there is a redo action available.
- bool [can_undo](#) () const
Check if the last operation can be undone.
- int [copy](#) (int clipboard)
Put the current selection into the clipboard.
- int [copy_cuts](#) ()
Copies the yank buffer to the clipboard.
- [Fl_Color](#) [cursor_color](#) () const
Gets the color of the cursor.
- void [cursor_color](#) ([Fl_Color](#) n)
Sets the color of the cursor.
- int [cut](#) ()
Deletes the current selection.
- int [cut](#) (int a, int b)
Deletes all characters between index a and b.
- int [cut](#) (int n)

- Deletes the next n bytes rounded to characters before or after the cursor.*

 - double `dvalue` () const

Returns the widget text interpreted as a floating point number.
- `FI_Input_` (int, int, int, int, const char *=0)
- Creates a new `FI_Input_` widget.*
- unsigned int `index` (int i) const
- Returns the character at index i .*
- int `input_type` () const
- Gets the input field type.*
- void `input_type` (int t)
- Sets the input field type.*
- int `insert` (const char *t, int l=0)
- Inserts text at the cursor position.*
- int `insert_position` () const
- Gets the position of the text cursor.*
- int `insert_position` (int p)
- Sets the cursor position and mark.*
- int `insert_position` (int p, int m)
- Sets the index for the cursor and mark.*
- int `ivalue` () const
- Returns the widget text interpreted as a signed integer.*
- int `mark` () const
- Gets the current selection mark.*
- int `mark` (int m)
- Sets the current selection mark.*
- int `maximum_size` () const
- Gets the maximum length of the input field in characters.*
- void `maximum_size` (int m)
- Sets the maximum length of the input field in characters.*
- int `position` () const
- int `position` (int p)
- int `position` (int p, int m)
- int `readonly` () const
- Gets the read-only state of the input field.*
- void `readonly` (int b)
- Sets the read-only state of the input field.*
- int `redo` ()
- Redo previous undo operation.*
- int `replace` (int b, int e, const char *text, int llen=0)
- Deletes text from b to e and inserts the new string `text`.*
- void `resize` (int, int, int, int) `FL_OVERRIDE`
- Changes the size of the widget.*
- int `shortcut` () const
- Return the shortcut key associated with this widget.*
- void `shortcut` (int s)
- Sets the shortcut key associated with this widget.*
- int `size` () const
- Returns the number of bytes in `value()`.*
- void `size` (int W, int H)
- Sets the width and height of this widget.*
- int `static_value` (const char *)

- Changes the widget text.*

 - int [static_value](#) (const char *, int)

Changes the widget text.
- int [tab_nav](#) () const

Gets whether the Tab key causes focus navigation in multiline input fields or not.
- void [tab_nav](#) (int val)

Sets whether the Tab key does focus navigation, or inserts tab characters into [FI_Multiline_Input](#).
- [FI_Color](#) [textcolor](#) () const

Gets the color of the text in the input field.
- void [textcolor](#) ([FI_Color](#) n)

Sets the color of the text in the input field.
- [FI_Font](#) [textfont](#) () const

Gets the font of the text in the input field.
- void [textfont](#) ([FI_Font](#) s)

Sets the font of the text in the input field.
- [FI_Fontsize](#) [textsize](#) () const

Gets the size of the text in the input field.
- void [textsize](#) ([FI_Fontsize](#) s)

Sets the size of the text in the input field.
- int [undo](#) ()

Undoes previous changes to the text buffer.
- const char * [value](#) () const

Returns the text displayed in the widget.
- int [value](#) (const char *)

Changes the widget text.
- int [value](#) (const char *, int)

Changes the widget text.
- int [value](#) (double value)

Changes the widget text to a floating point number ("%g").
- int [value](#) (int value)

Changes the widget text to a signed integer number.
- int [wrap](#) () const

Gets the word wrapping state of the input field.
- void [wrap](#) (int b)

Sets the word wrapping state of the input field.
- [~FI_Input_](#) ()

Destroys the widget.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
- Activates the widget.*
- unsigned int [active](#) () const

Returns whether the widget is active.
- int [active_r](#) () const

Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const

Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)

- Sets the label alignment.*
- long [argument](#) () const
 - Gets the current user data (long) argument that is passed to the callback function.*
- void [argument](#) (long v)
 - Sets the current user data (long) argument that is passed to the callback function.*
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
 - Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).*
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
 - Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).*
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
 - Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
 - Sets the image to use as part of the widget label when in the inactive state.*
- void [bind_deimage](#) (int f)
 - Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- void [bind_image](#) ([FI_Image](#) *img)
 - Sets the image to use as part of the widget label when in the active state.*
- void [bind_image](#) (int f)
 - Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- [FI_Boxtype](#) box () const
 - Gets the box type of the widget.*
- void [box](#) ([FI_Boxtype](#) new_box)
 - Sets the box type for the widget.*
- [FI_Callback_p](#) callback () const
 - Gets the current callback function for the widget.*
- void [callback](#) ([FI_Callback](#) *cb)
 - Sets the current callback function for the widget.*
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
 - Sets the current callback function and managed user data for the widget.*
- void [callback](#) ([FI_Callback](#) *cb, void *p)
 - Sets the current callback function and data for the widget.*
- void [callback](#) ([FI_Callback0](#) *cb)
 - Sets the current callback function for the widget.*
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
 - Sets the current callback function for the widget.*
- unsigned int [changed](#) () const
 - Checks if the widget value changed since the last callback.*
- void [clear_active](#) ()
 - Marks the widget as inactive without sending events or changing focus.*
- void [clear_changed](#) ()
 - Marks the value of the widget as unchanged.*
- void [clear_damage](#) (uchar c=0)
 - Clears or sets the damage flags.*
- void [clear_output](#) ()
 - Sets a widget to accept input.*
- void [clear_visible](#) ()
 - Hides the widget.*
- void [clear_visible_focus](#) ()

- Disables keyboard focus navigation with this widget.*

 - `FL_Color color () const`

Gets the background color of the widget.
- `void color (FL_Color bg)`

Sets the background color of the widget.
- `void color (FL_Color bg, FL_Color sel)`

Sets the background and selection color of the widget.
- `FL_Color color2 () const`

For back compatibility only.
- `void color2 (unsigned a)`

For back compatibility only.
- `int contains (const FL_Widget *w) const`

Checks if w is a child of this widget.
- `void copy_label (const char *new_label)`

Sets the current label.
- `void copy_tooltip (const char *text)`

Sets the current tooltip text.
- `uchar damage () const`

Returns non-zero if `draw()` needs to be called.
- `void damage (uchar c)`

Sets the damage bits for the widget.
- `void damage (uchar c, int x, int y, int w, int h)`

Sets the damage bits for an area inside the widget.
- `int damage_resize (int, int, int, int)`

Internal use only.
- `void deactivate ()`

Deactivates the widget.
- `FL_Image * deimage ()`

Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`

Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`

Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`

Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`

Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `virtual void draw ()=0`

Draws the widget.
- `void draw_label (int, int, int, int, FL_Align) const`

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`

Gets the widget height.
- `virtual int handle (int event)`

Handles the specified event.

- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

- Sets whether this widget needs a keyboard.*

 - unsigned int `output` () const

Returns if a widget is used for output only.
- `Fl_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`Fl_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- `Fl_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`Fl_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.
- void `set_visible` ()

Makes the widget visible.
- void `set_visible_focus` ()

Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()

Makes a widget visible.
- void `size` (int W, int H)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const

Gets the current tooltip text.
- void `tooltip` (const char *text)

Sets the current tooltip text.
- `Fl_Window` * `top_window` () const

Returns a pointer to the top-level window for the widget.
- `Fl_Window` * `top_window_offset` (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- uchar `type` () const

Gets the widget type.

- void **type** (uchar t)
Sets the widget type.
- int **use_accents_menu** ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const
Gets the user data for this widget.
- void **user_data** (Fl_Callback_User_Data *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **Fl_When** when () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual ~**Fl_Widget** ()
Destroys the widget.

Protected Member Functions

- int **apply_undo** ()
Apply the current undo/redo operation.
- void **drawtext** (int, int, int, int)
Draws the text in the passed bounding box.
- void **drawtext** (int, int, int, int, bool draw_active)
Draws the text in the passed bounding box.
- void **handle_mouse** (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int **handletext** (int e, int, int, int, int)
Handles all kinds of text field related events.
- int **line_end** (int i) const
Finds the end of a line.

- int **line_start** (int i) const
Finds the start of a line.
- int **linesPerPage** ()
- void **maybe_do_callback** (FI_Callback_Reason reason=FL_REASON_UNKNOWN)
- int **up_down_position** (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int **word_end** (int i) const
Finds the end of a word.
- int **word_start** (int i) const
Finds the start of a word.
- int **xscroll** () const
- int **yscroll** () const
- void **yscroll** (int yOffset)

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void [default_callback](#) (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.75.1 Detailed Description

This class provides a low-overhead text input field.

This is a virtual base class below [FI_Input](#). It has all the same interfaces, but lacks the [handle\(\)](#) and [draw\(\)](#) method. You may want to subclass it if you are one of those people who likes to change how the editing keys work. It may also be useful for adding scrollbars to the input field.

This can act like any of the subclasses of [FI_Input](#), by setting [type\(\)](#) to one of the following values:

```
#define FL_NORMAL_INPUT      0
#define FL_FLOAT_INPUT      1
#define FL_INT_INPUT        2
#define FL_MULTILINE_INPUT  4
#define FL_SECRET_INPUT     5
#define FL_INPUT_TYPE       7
#define FL_INPUT_READONLY   8
#define FL_NORMAL_OUTPUT    (FL_NORMAL_INPUT | FL_INPUT_READONLY)
#define FL_MULTILINE_OUTPUT (FL_MULTILINE_INPUT | FL_INPUT_READONLY)
#define FL_INPUT_WRAP       16
#define FL_MULTILINE_INPUT_WRAP (FL_MULTILINE_INPUT | FL_INPUT_WRAP)
#define FL_MULTILINE_OUTPUT_WRAP (FL_MULTILINE_INPUT | FL_INPUT_READONLY | FL_INPUT_WRAP)
```

All variables that represent an index into a text buffer are byte-oriented, not character oriented, counting from 0 (at or before the first character) to [size\(\)](#) (at the end of the buffer, after the last byte). Since UTF-8 characters can be up to six bytes long, simply incrementing such an index will not reliably advance to the next character in the text buffer. Indices and pointers into the text buffer should always point at a 7 bit ASCII character or the beginning of a UTF-8 character sequence. Behavior for false UTF-8 sequences and pointers into the middle of a sequence are undefined.

See also

[FI_Text_Display](#), [FI_Text_Editor](#) for more powerful text handling widgets

[FI_Widget::shortcut_label\(int\)](#)

11.75.2 Constructor & Destructor Documentation

11.75.2.1 FI_Input_()

```
Fl_Input_::Fl_Input_ (
```

```

    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)

```

Creates a new [Fl_Input_](#) widget.

This function creates a new [Fl_Input_](#) widget and adds it to the current [Fl_Group](#). The [value\(\)](#) is set to `NULL`. The default boxtype is `FL_DOWN_BOX`.

Parameters

<i>X,Y,W,H</i>	the dimensions of the new widget
<i>l</i>	an optional label text

11.75.2.2 ~Fl_Input_()

```
Fl_Input_::~~Fl_Input_ ()
```

Destroys the widget.

The destructor clears all allocated buffers and removes the widget from the parent [Fl_Group](#).

11.75.3 Member Function Documentation

11.75.3.1 append()

```

int Fl_Input_::append (
    const char * t,
    int l = 0,
    char keep_selection = 0)

```

Append text at the end.

This function appends the string in *t* to the end of the text. It does not moves the new position or mark.

Parameters

in	<i>t</i>	text that will be appended
in	<i>l</i>	length of text, or 0 if the string is terminated by <code>nul</code> .
in	<i>keep_selection</i>	if this is 1, the current text selection will remain, if 0, the cursor will move to the end of the inserted text.

Returns

0 if no text was appended

11.75.3.2 apply_undo()

```
int Fl_Input_::apply_undo () [protected]
```

Apply the current undo/redo operation.

It's up to [undo\(\)](#) and [redo\(\)](#) to push and pop actions to and from the lists.

Returns

1 if the current action changed any text.

See also

[undo\(\)](#), [redo\(\)](#)

11.75.3.3 can_redo()

```
bool Fl_Input_::can_redo () const
```

Check if there is a redo action available.

Returns

true if the widget can redo the last undo action

11.75.3.4 can_undo()

```
bool Fl_Input_::can_undo () const
```

Check if the last operation can be undone.

Returns

true if the widget can undo the last change

11.75.3.5 copy()

```
int Fl_Input_::copy (
    int clipboard)
```

Put the current selection into the clipboard.

This function copies the current selection between [mark\(\)](#) and [position\(\)](#) into the specified `clipboard`. This does not replace the old clipboard contents if [position\(\)](#) and [mark\(\)](#) are equal. Clipboard 0 maps to the current text selection and clipboard 1 maps to the cut/paste clipboard.

Parameters

<i>clipboard</i>	the clipboard destination 0 or 1

Returns

0 if no text is selected, 1 if the selection was copied

See also

`Fl::copy(const char *, int, int)`

11.75.3.6 copy_cuts()

```
int Fl_Input_::copy_cuts ()
```

Copies the *yank* buffer to the clipboard.

This method copies all the previous contiguous cuts from the undo information to the clipboard. This function implements the [^]K shortcut key.

Returns

0 if the operation did not change the clipboard

See also

[copy\(int\)](#), [cut\(\)](#)

11.75.3.7 cursor_color() [1/2]

```
Fl_Color Fl_Input_::cursor_color () const [inline]
```

Gets the color of the cursor.

Returns

the current cursor color

11.75.3.8 cursor_color() [2/2]

```
void Fl_Input_::cursor_color (
    Fl_Color n) [inline]
```

Sets the color of the cursor.

The default color for the cursor is FL_BLACK.

Parameters

in	n	the new cursor color

11.75.3.9 cut() [1/3]

```
int Fl_Input_::cut () [inline]
```

Deletes the current selection.

This function deletes the currently selected text *without* storing it in the clipboard. To use the clipboard, you may call [copy\(\)](#) first or [copy_cuts\(\)](#) after this call.

Returns

0 if no data was copied

11.75.3.10 cut() [2/3]

```
int Fl_Input_::cut (
    int a,
    int b) [inline]
```

Deletes all characters between index a and b.

This function deletes the currently selected text *without* storing it in the clipboard. To use the clipboard, you may call [copy\(\)](#) first or [copy_cuts\(\)](#) after this call.

Parameters

a,b	range of bytes rounded to full characters and clamped to the buffer

Returns

0 if no data was copied

11.75.3.11 cut() [3/3]

```
int Fl_Input_::cut (
    int n) [inline]
```

Deletes the next n bytes rounded to characters before or after the cursor.

This function deletes the currently selected text *without* storing it in the clipboard. To use the clipboard, you may call [copy\(\)](#) first or [copy_cuts\(\)](#) after this call.

Parameters

n	number of bytes rounded to full characters and clamped to the buffer. A negative number will cut characters to the left of the cursor.

Returns

0 if no data was copied

11.75.3.12 drawtext() [1/2]

```
void Fl_Input_::drawtext (
    int X,
    int Y,
    int W,
    int H) [protected]
```

Draws the text in the passed bounding box.

If [damage\(\)](#) & `FL_DAMAGE_ALL` is true, this assumes the area has already been erased to [color\(\)](#). Otherwise it does minimal update and erases the area itself.

Parameters

<i>X,Y,W,H</i>	area that must be redrawn

11.75.3.13 drawtext() [2/2]

```
void Fl_Input_::drawtext (
    int X,
    int Y,
    int W,
    int H,
    bool draw_active) [protected]
```

Draws the text in the passed bounding box.

This version of `drawtext` allows the user to control whether the widget is drawn as active, i.e. with the text cursor, or inactive. This is useful for compound widgets where the input should be shown as active when actually the container widget is the active one.

A caller should not draw the widget with `active` set if another text widget may indeed be the active widget.

Parameters

<i>X,Y,W,H</i>	area that must be redrawn
<i>draw_active</i>	if set, the cursor will be drawn, even if the widget is not active

See also

[Fl_Input_::drawtext\(int X, int Y, int W, int H\)](#)

11.75.3.14 dvalue()

```
double Fl_Input_::dvalue () const
```

Returns the widget text interpreted as a floating point number.

Returns

double precision floating point value

See also

[Fl_Input_::ivalue\(\)](#)

[Fl_Input_::value\(double\)](#)

11.75.3.15 handle_mouse()

```
void Fl_Input_::handle_mouse (
    int X,
    int Y,
    int ,
    int ,
    int drag = 0) [protected]
```

Handles mouse clicks and mouse moves.

Todo Add comment and parameters

11.75.3.16 handletext()

```
int Fl_Input_::handletext (
    int event,
    int X,
    int Y,
    int W,
    int H) [protected]
```

Handles all kinds of text field related events.

This is called by derived classes.

Todo Add comment and parameters

11.75.3.17 index()

```
unsigned int Fl_Input_::index (
    int i) const
```

Returns the character at index *i*.

This function returns the UTF-8 character at *i* as a ucs4 character code.

Parameters

<i>in</i>	<i>i</i>	index into the value field

Returns

the character at index *i*

11.75.3.18 input_type() [1/2]

```
int Fl_Input_::input_type () const [inline]
```

Gets the input field type.

Returns

the current input type

11.75.3.19 input_type() [2/2]

```
void Fl_Input_::input_type (
    int t) [inline]
```

Sets the input field type.

A [redraw\(\)](#) is required to reformat the input field.

Parameters

<code>in</code>	<code>t</code>	new input type

11.75.3.20 insert()

```
int Fl_Input_::insert (
    const char * t,
    int l = 0) [inline]
```

Inserts text at the cursor position.

This function inserts the string in `t` at the cursor [position\(\)](#) and moves the new position and mark to the end of the inserted text.

Parameters

<code>in</code>	<code>t</code>	text that will be inserted
<code>in</code>	<code>l</code>	length of text, or 0 if the string is terminated by <code>nul</code> .

Returns

0 if no text was inserted

11.75.3.21 insert_position() [1/3]

```
int Fl_Input_::insert_position () const [inline]
```

Gets the position of the text cursor.

Returns

the cursor position as an index in the range 0..[size\(\)](#)

See also

[insert_position\(int, int\)](#)

11.75.3.22 insert_position() [2/3]

```
int Fl_Input_::insert_position (
    int p) [inline]
```

Sets the cursor position and mark.

`position(n)` is the same as `position(n, n)`.

Parameters

<code>p</code>	new index for cursor and mark

Returns

0 if no positions changed

See also

[insert_position\(int, int\)](#), [insert_position\(\)](#), [mark\(int\)](#)

11.75.3.23 insert_position() [3/3]

```
int Fl_Input_::insert_position (
    int p,
    int m)
```

Sets the index for the cursor and mark.

The input widget maintains two pointers into the string. The *position* (*p*) is where the cursor is. The *mark* (*m*) is the other end of the selected text. If they are equal then there is no selection. Changing this does not affect the clipboard (use [copy\(\)](#) to do that).

Changing these values causes a [redraw\(\)](#). The new values are bounds checked.

Parameters

<i>p</i>	index for the cursor position
<i>m</i>	index for the mark

Returns

0 if no positions changed

See also

[position\(int\)](#), [position\(\)](#), [mark\(int\)](#)

11.75.3.24 ivalue()

```
int Fl_Input_::ivalue () const
```

Returns the widget text interpreted as a signed integer.

Returns

signed integer value

See also

[Fl_Input_::dvalue\(\)](#)

[Fl_Input_::value\(int\)](#)

11.75.3.25 line_end()

```
int Fl_Input_::line_end (
    int i) const [protected]
```

Finds the end of a line.

This call calculates the end of a line based on the given index *i*.

Parameters

<i>in</i>	<i>i</i>	starting index for the search

Returns

end of the line

11.75.3.26 line_start()

```
int Fl_Input_::line_start (
    int i) const [protected]
```

Finds the start of a line.

This call calculates the start of a line based on the given index *i*.

Parameters

<code>in</code>	<code>i</code>	starting index for the search

Returns

start of the line

11.75.3.27 mark() [1/2]

```
int Fl_Input_::mark () const [inline]
```

Gets the current selection mark.

Returns

index into the text

11.75.3.28 mark() [2/2]

```
int Fl_Input_::mark (
    int m) [inline]
```

Sets the current selection mark.

mark(n) is the same as `insert_position(insert_position(), n)`.

Parameters

<code>m</code>	new index of the mark

Returns

0 if the mark did not change

See also

[insert_position\(\)](#), [insert_position\(int, int\)](#)

11.75.3.29 maximum_size() [1/2]

```
int Fl_Input_::maximum_size () const [inline]
```

Gets the maximum length of the input field in characters.

See also

[maximum_size\(int\)](#).

11.75.3.30 maximum_size() [2/2]

```
void Fl_Input_::maximum_size (
    int m) [inline]
```

Sets the maximum length of the input field in characters.

This limits the number of **characters** that can be inserted in the widget.

Since FLTK 1.3 this is different than the buffer size, since one character can be more than one byte in UTF-8 encoding. In FLTK 1.1 this was the same (one byte = one character).

11.75.3.31 position() [1/3]

```
int Fl_Input_::position () const [inline]
```

Deprecated "since 1.4.0 - use `insert_position()` instead"

11.75.3.32 position() [2/3]

```
int Fl_Input_::position (
    int p) [inline]
```

Deprecated "since 1.4.0 - use insert_position(p) instead"

11.75.3.33 position() [3/3]

```
int Fl_Input_::position (
    int p,
    int m) [inline]
```

Deprecated "since 1.4.0 - use insert_position(p, m) or Fl_Widget::position(x, y) instead"

11.75.3.34 readonly() [1/2]

```
int Fl_Input_::readonly () const [inline]
```

Gets the read-only state of the input field.

Returns

non-zero if this widget is read-only

11.75.3.35 readonly() [2/2]

```
void Fl_Input_::readonly (
    int b) [inline]
```

Sets the read-only state of the input field.

Parameters

<code>in</code>	<code>b</code>	if <code>b</code> is 0, the text in this widget can be edited by the user

11.75.3.36 redo()

```
int Fl_Input_::redo ()
```

Redo previous undo operation.

This call reapplies previously executed undo operations.

Returns

non-zero if any change was made.

11.75.3.37 replace()

```
int Fl_Input_::replace (
    int b,
    int e,
    const char * text,
    int ilen = 0)
```

Deletes text from `b` to `e` and inserts the new string `text`.

All changes to the text buffer go through this function. It deletes the region between `b` and `e` (either one may be less or equal to the other), and then inserts the string `text` at that point and moves the [mark\(\)](#) and [position\(\)](#) to the end of the insertion. Does the callback if [when\(\)](#) & `FL_WHEN_CHANGED` and there is a change.

Set `b` and `e` equal to not delete anything. Set `text` to `NULL` to not insert anything.

`ilen` can be zero or `strlen(text)`, which saves a tiny bit of time if you happen to already know the length of the insertion, or can be used to insert a portion of a string. If `ilen` is zero, `strlen(text)` is used instead.

`b` and `e` are clamped to the `0..size()` range, so it is safe to pass any values. `b`, `e`, and `ilen` are used as numbers of bytes (not characters), where `b` and `e` count from 0 to `size()` (end of buffer).

If `b` and/or `e` don't point to a valid UTF-8 character boundary, they are adjusted to the previous (`b`) or the next (`e`) valid UTF-8 character boundary, resp..

If the current number of characters in the buffer minus deleted characters plus inserted characters in `text` would overflow the number of allowed characters (`maximum_size()`), then only the first characters of the string are inserted, so that `maximum_size()` is not exceeded.

`cut()` and `insert()` are just inline functions that call `replace()`.

Parameters

in	<i>b</i>	beginning index of text to be deleted
in	<i>e</i>	ending index of text to be deleted and insertion position
in	<i>text</i>	string that will be inserted
in	<i>ilen</i>	length of <code>text</code> or 0 for <code>nul</code> terminated strings

Returns

0 if nothing changed

Note

If `text` does not point to a valid UTF-8 character or includes invalid UTF-8 sequences, the text is inserted nevertheless (counting invalid UTF-8 bytes as one character each).

11.75.3.38 `resize()`

```
void Fl_Input_::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Changes the size of the widget.

This call updates the text layout so that the cursor is visible.

Parameters

in	<i>X,Y,W,H</i>	new size of the widget

See also

[Fl_Widget::resize\(int, int, int, int\)](#)

Reimplemented from [Fl_Widget](#).

11.75.3.39 `shortcut()` [1/2]

```
int Fl_Input_::shortcut () const [inline]
```

Return the shortcut key associated with this widget.

Returns

shortcut keystroke

See also

[Fl_Button::shortcut\(\)](#)

11.75.3.40 shortcut() [2/2]

```
void Fl_Input_::shortcut (
    int s) [inline]
```

Sets the shortcut key associated with this widget.

Pressing the shortcut key gives text editing focus to this widget.

Parameters

in	s	new shortcut keystroke

See also

[Fl_Button::shortcut\(\)](#)

11.75.3.41 size() [1/2]

```
int Fl_Input_::size () const [inline]
```

Returns the number of bytes in [value\(\)](#).

This may be greater than `strlen(value())` if there are `nul` characters in the text.

Returns

number of bytes in the text

11.75.3.42 size() [2/2]

```
void Fl_Input_::size (
    int W,
    int H) [inline]
```

Sets the width and height of this widget.

Parameters

in	W,H	new width and height

See also

[Fl_Widget::size\(int, int\)](#)

11.75.3.43 static_value() [1/2]

```
int Fl_Input_::static_value (
    const char * str)
```

Changes the widget text.

This function changes the text and sets the mark and the point to the end of it. The string is *not* copied. If the user edits the string it is copied to the internal buffer then. This can save a great deal of time and memory if your program is rapidly changing the values of text fields, but this will only work if the passed string remains unchanged until either the [Fl_Input](#) is destroyed or [value\(\)](#) is called again.

Parameters

in	str	the new text

Returns

non-zero if the new value is different than the current one

11.75.3.44 static_value() [2/2]

```
int Fl_Input_::static_value (
    const char * str,
    int len)
```

Changes the widget text.

This function changes the text and sets the mark and the point to the end of it. The string is *not* copied. If the user edits the string it is copied to the internal buffer then. This can save a great deal of time and memory if your program is rapidly changing the values of text fields, but this will only work if the passed string remains unchanged until either the [Fl_Input](#) is destroyed or [value\(\)](#) is called again.

You can use the `len` parameter to directly set the length if you know it already or want to put `nul` characters in the text.

Parameters

in	<i>str</i>	the new text
in	<i>len</i>	the length of the new text

Returns

non-zero if the new value is different than the current one

11.75.3.45 tab_nav() [1/2]

```
int Fl_Input_::tab_nav () const [inline]
```

Gets whether the Tab key causes focus navigation in multiline input fields or not.

If enabled (default), hitting Tab causes focus navigation to the next widget.

If disabled, hitting Tab inserts a tab character into the text field.

Returns

1 if Tab advances focus (default), 0 if Tab inserts tab characters.

See also

[tab_nav\(int\)](#), [Fl::OPTION_ARROW_FOCUS](#).

11.75.3.46 tab_nav() [2/2]

```
void Fl_Input_::tab_nav (
    int val) [inline]
```

Sets whether the Tab key does focus navigation, or inserts tab characters into [Fl_Multiline_Input](#).

By default this flag is enabled to provide the 'normal' behavior most users expect; Tab navigates focus to the next widget. To inserting an actual Tab character, users can use Ctrl-I or copy/paste.

Disabling this flag gives the old FLTK behavior where Tab inserts a tab character into the text field, in which case only the mouse can be used to navigate to the next field.

History: This flag was provided for backwards support of FLTK's old 1.1.x behavior where Tab inserts a tab character instead of navigating focus to the next widget. This behavior was unique to [Fl_Multiline_Input](#). With the advent of [Fl_Text_Editor](#), this old behavior has been deprecated.

Parameters

in	<i>val</i>	If <i>val</i> is 1, Tab advances focus (default). If <i>val</i> is 0, Tab inserts a tab character (old FLTK behavior).

See also

[tab_nav\(\)](#), [Fl::OPTION_ARROW_FOCUS](#).

11.75.3.47 textcolor() [1/2]

```
Fl_Color Fl_Input_::textcolor () const [inline]
```

Gets the color of the text in the input field.

Returns

the text color

See also

[textcolor\(Fl_Color\)](#)

11.75.3.48 textcolor() [2/2]

```
void Fl_Input_::textcolor (
    Fl_Color n) [inline]
```

Sets the color of the text in the input field.

The text color defaults to FL_FOREGROUND_COLOR.

Parameters

in	n	new text color

See also

[textcolor\(\)](#)

11.75.3.49 textfont() [1/2]

```
Fl_Font Fl_Input_::textfont () const [inline]
```

Gets the font of the text in the input field.

Returns

the current Fl_Font index

11.75.3.50 textfont() [2/2]

```
void Fl_Input_::textfont (
    Fl_Font s) [inline]
```

Sets the font of the text in the input field.

The text font defaults to FL_HELVETICA.

Parameters

in	s	the new text font

11.75.3.51 textsize() [1/2]

```
Fl_Fontsize Fl_Input_::textsize () const [inline]
```

Gets the size of the text in the input field.

Returns

the text height in pixels

11.75.3.52 `textsize()` [2/2]

```
void Fl_Input_::textsize (  
    Fl_Fontsize s) [inline]
```

Sets the size of the text in the input field.

The text height defaults to `FL_NORMAL_SIZE`.

Parameters

in	s	the new font height in pixel units

11.75.3.53 undo()

```
int Fl_Input_::undo ()
```

Undoes previous changes to the text buffer.

This call undoes a number of previous calls to [replace\(\)](#).

Returns

non-zero if any change was made.

11.75.3.54 up_down_position()

```
int Fl_Input_::up_down_position (
    int i,
    int keepmark = 0) [protected]
```

Moves the cursor to the column given by `up_down_pos`.

This function is helpful when implementing up and down cursor movement. It moves the cursor from the beginning of a line to the column indicated by the global variable `up_down_pos` in pixel units.

Parameters

in	<i>i</i>	index into the beginning of a line of text
in	<i>keepmark</i>	if set, move only the cursor, but not the mark

Returns

index to new cursor position

11.75.3.55 value() [1/5]

```
const char * Fl_Input_::value () const [inline]
```

Returns the text displayed in the widget.

This function returns the current value, which is a pointer to the internal buffer and is valid only until the next event is handled.

Returns

pointer to an internal buffer - do not free() this

See also

[Fl_Input_::value\(const char*\)](#)

11.75.3.56 value() [2/5]

```
int Fl_Input_::value (
    const char * str)
```

Changes the widget text.

This function changes the text and sets the mark and the point to the end of it. The string is copied to the internal buffer. Passing NULL is the same as "".

Parameters

<code>in</code>	<code>str</code>	the new text

Returns

non-zero if the new value is different than the current one

See also

[Fl_Input_::value\(const char* str, int len\)](#), [Fl_Input_::value\(\)](#)

11.75.3.57 value() [3/5]

```
int Fl_Input_::value (
    const char * str,
    int len)
```

Changes the widget text.

This function changes the text and sets the mark and the point to the end of it. The string is copied to the internal buffer. Passing `NULL` is the same as `""`.

You can use the `length` parameter to directly set the length if you know it already or want to put `nul` characters in the text.

Parameters

<code>in</code>	<code>str</code>	the new text
<code>in</code>	<code>len</code>	the length of the new text

Returns

non-zero if the new value is different than the current one

See also

[Fl_Input_::value\(const char* str\)](#), [Fl_Input_::value\(\)](#)

11.75.3.58 value() [4/5]

```
int Fl_Input_::value (
    double v)
```

Changes the widget text to a floating point number ("%g").

Parameters

<code>in</code>	<code>v</code>	the new value

Returns

non-zero if the new value is different than the current one

See also

[Fl_Input_::value\(const char* str\)](#), [Fl_Input_::ivalue\(\)](#)

11.75.3.59 value() [5/5]

```
int Fl_Input_::value (
    int v)
```

Changes the widget text to a signed integer number.

Parameters

in	v	the new value

Returns

non-zero if the new value is different than the current one

See also

[Fl_Input_::value\(const char* str\)](#), [Fl_Input_::ivalue\(\)](#)

11.75.3.60 word_end()

```
int Fl_Input_::word_end (
    int i) const [protected]
```

Finds the end of a word.

Returns the index after the last byte of a word. If the index is already at the end of a word, it will find the end of the following word, so if you call it repeatedly you will move forwards to the end of the text.

Note that this is inconsistent with [line_end\(\)](#).

Parameters

in	i	starting index for the search

Returns

end of the word

11.75.3.61 word_start()

```
int Fl_Input_::word_start (
    int i) const [protected]
```

Finds the start of a word.

Returns the index of the first byte of a word. If the index is already at the beginning of a word, it will find the beginning of the previous word, so if you call it repeatedly you will move backwards to the beginning of the text.

Note that this is inconsistent with [line_start\(\)](#).

Parameters

in	i	starting index for the search

Returns

start of the word, or previous word

11.75.3.62 wrap() [1/2]

```
int Fl_Input_::wrap () const [inline]
```

Gets the word wrapping state of the input field.

Word wrap is only functional with multi-line input fields.

11.75.3.63 wrap() [2/2]

```
void Fl_Input_::wrap (
    int b) [inline]
```

Sets the word wrapping state of the input field.

Word wrap is only functional with multi-line input fields.

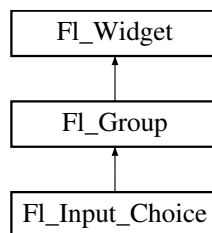
The documentation for this class was generated from the following files:

- Fl_Input_.H
- Fl_Input_.cxx

11.76 Fl_Input_Choice Class Reference

A combination of the input widget and a menu button.

Inheritance diagram for Fl_Input_Choice:

**Public Member Functions**

- void **add** (const char *s)
Adds an item to the menu.
- int **changed** () const
Returns the combined [changed\(\)](#) state of the input and menu button widget.
- void **clear** ()
Removes all items from the menu.
- void **clear_changed** ()
Clears the [changed\(\)](#) state of both input and menu button widgets.
- [Fl_Boxtype](#) **down_box** () const
Gets the box type of the menu button.
- void **down_box** ([Fl_Boxtype](#) b)
Sets the box type of the menu button.
- [Fl_Input_Choice](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Input_Choice](#) widget using the given position, size, and label string.
- [Fl_Input](#) * **input** ()
Returns a pointer to the internal [Fl_Input](#) widget.
- const [Fl_Menu_Item](#) * **menu** ()
Gets the [Fl_Menu_Item](#) array used for the menu.
- void **menu** (const [Fl_Menu_Item](#) *m)
Sets the [Fl_Menu_Item](#) array used for the menu.
- [Fl_Menu_Button](#) * **menubutton** ()
Returns a pointer to the internal [Fl_Menu_Button](#) widget.
- void **resize** (int X, int Y, int W, int H) [FL_OVERRIDE](#)
Resizes the [Fl_Input_Choice](#) widget.
- void **set_changed** ()
Sets the [changed\(\)](#) state of both input and menu button widgets to the specified value.
- [Fl_Color](#) **textcolor** () const

- Gets the [FL_Input](#) text field's text color.*

 - void **textcolor** ([FL_Color](#) c)

Sets the [FL_Input](#) text field's text color to c.
- [FL_Font](#) **textfont** () const

Gets the [FL_Input](#) text field's font style.
- void **textfont** ([FL_Font](#) f)

Sets the [FL_Input](#) text field's font style to f.
- [FL_Fontsize](#) **textsize** () const

Gets the [FL_Input](#) text field's font size.
- void **textsize** ([FL_Fontsize](#) s)

Sets the [FL_Input](#) text field's font size to s.
- int **update_menubutton** ()

Updates the menubutton with the string value in [FL_Input](#).
- const char * **value** () const

Returns the [FL_Input](#) text field's current contents.
- void **value** (const char *val)

Sets the [FL_Input](#) text field's contents to val.
- void **value** (int val)

Chooses item# val in the menu, and sets the [FL_Input](#) text field to that value.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *& **_ddfdesign_kludge** ()

This is for forms compatibility only.
- void **add** ([FL_Widget](#) &)

The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** ([FL_Widget](#) *o)

See void [FL_Group::add\(FL_Widget &w\)](#)
- void **add_resizable** ([FL_Widget](#) &o)

Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * **array** () const

Returns a pointer to the internal array of children.
- [FL_Group](#) const * **as_group** () const [FL_OVERRIDE](#)
- [FL_Group](#) * **as_group** () [FL_OVERRIDE](#)

Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void **begin** ()

Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * **child** (int n) const

Returns the n'th child.
- int **children** () const

Returns how many child widgets the group has.
- void **clear** ()

Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()

Returns the current clipping mode.
- void **clip_children** (int c)

Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)

Removes the widget at index from the group and deletes it.
- void **end** ()

Exactly the same as current(this->[parent\(\)](#)).

- int **find** (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int **find** (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char *s=0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void **focus** ([FL_Widget](#) *W)
- void **forms_end** ()
This is for forms compatibility only.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** ([FL_Widget](#) &o, [FL_Widget](#) *before)
This does insert(w, find(before)).
- void **remove** ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void **remove** ([FL_Widget](#) *o)
Removes the widget o from the group.
- void **remove** (int index)
Removes the widget at index from the group but does not delete it.
- [FL_Widget](#) * **resizable** () const
Returns the group's resizable widget.
- void **resizable** ([FL_Widget](#) &o)
Sets the group's resizable widget.
- void **resizable** ([FL_Widget](#) *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual [~FL_Group](#) ()
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- [FL_Align](#) **align** () const
Gets the label alignment.
- void **align** ([FL_Align](#) alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FL_Gl_Window](#) * **as_gl_window** ()

- Returns an [FI_GI_Window](#) pointer if this widget is an [FI_GI_Window](#).*

 - virtual class [FI_GI_Window](#) const * **as_gi_window** () const
 - virtual [FI_Window](#) * **as_window** ()
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*

 - virtual [FI_Window](#) const * **as_window** () const
 - void **bind_deimage** ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.

 - void **bind_deimage** (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.

 - void **bind_image** ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.

 - void **bind_image** (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.

 - [FI_Boxtype](#) **box** () const

Gets the box type of the widget.

 - void **box** ([FI_Boxtype](#) new_box)

Sets the box type for the widget.

 - [FI_Callback_p](#) **callback** () const

Gets the current callback function for the widget.

 - void **callback** ([FI_Callback](#) *cb)

Sets the current callback function for the widget.

 - void **callback** ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.

 - void **callback** ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.

 - void **callback** ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.

 - void **callback** ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.

 - unsigned int **changed** () const

Checks if the widget value changed since the last callback.

 - void **clear_active** ()

Marks the widget as inactive without sending events or changing focus.

 - void **clear_changed** ()

Marks the value of the widget as unchanged.

 - void **clear_damage** (uchar c=0)

Clears or sets the damage flags.

 - void **clear_output** ()

Sets a widget to accept input.

 - void **clear_visible** ()

Hides the widget.

 - void **clear_visible_focus** ()

Disables keyboard focus navigation with this widget.

 - [FI_Color](#) **color** () const

Gets the background color of the widget.

 - void **color** ([FI_Color](#) bg)

Sets the background color of the widget.

 - void **color** ([FI_Color](#) bg, [FI_Color](#) sel)

Sets the background and selection color of the widget.

 - [FI_Color](#) **color2** () const

For back compatibility only.

- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FL_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)

- Sets the image to use as part of the widget label when in the active state.*

 - int `image_bound` () const

Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const

Checks if this widget is a child of wgt.
- int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const

Gets the current label text.
- void `label` (const char *text)

Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const

Gets the label color.
- void `labelcolor` (`FL_Color` c)

Sets the label color.
- `FL_Font` `labelfont` () const

Gets the font to use.
- void `labelfont` (`FL_Font` f)

Sets the font to use.
- `FL_Fonsize` `labelsize` () const

Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)

Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const

Gets the label type.
- void `labeltype` (`FL_Labeltype` a)

Sets the label type.
- void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FL_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FL_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.

- [FI_Color selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)

- Set the spacing between the label and the vertical edge of the widget.*

 - unsigned int **visible** () const
Returns whether a widget is visible.
 - unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
 - void **visible_focus** (int v)
Modifies keyboard focus navigation.
 - int **visible_r** () const
Returns whether a widget and all its parents are visible.
 - int **w** () const
Gets the widget width.
 - **FL_When** when () const
Returns the conditions under which the callback is called.
 - void **when** (uchar i)
Sets the flags used to decide when a callback is called.
 - **FL_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
 - int **x** () const
Gets the widget position in its window.
 - int **y** () const
Gets the widget position in its window.
 - virtual ~**FL_Widget** ()
Destroys the widget.

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
Draws the widget.
- virtual int **inp_h** () const
See [inp_x\(\)](#) for info.
- virtual int **inp_w** () const
See [inp_x\(\)](#) for info.
- virtual int **inp_x** () const
The methods [inp_x\(\)](#), [inp_y\(\)](#), [inp_w\(\)](#) and [inp_h\(\)](#) return the desired position and size of the internal [FL_Input](#) widget.
- virtual int **inp_y** () const
See [inp_x\(\)](#) for info.
- virtual int **menu_h** () const
See [menu_x\(\)](#) for info.
- virtual int **menu_w** () const
See [menu_x\(\)](#) for info.
- virtual int **menu_x** () const
The methods [menu_x\(\)](#), [menu_y\(\)](#), [menu_w\(\)](#) and [menu_h\(\)](#) return the desired position and size of the internal [FL_Menu_Button](#) widget.
- virtual int **menu_y** () const
See [menu_x\(\)](#) for info.

Protected Member Functions inherited from FL_Group

- `FL_Rect * bounds ()`
Returns the internal array of widget sizes and positions.
- `void draw_child (FL_Widget &widget) const`
Forces a child to redraw.
- `void draw_children ()`
Draws all children of the group.
- `void draw_outside_label (const FL_Widget &widget) const`
Parents normally call this to draw outside labels of child widgets.
- `virtual int on_insert (FL_Widget *, int)`
Allow derived groups to act when a widget is added as a child.
- `virtual int on_move (int, int)`
Allow derived groups to act when a widget is moved within the group.
- `virtual void on_remove (int)`
Allow derived groups to act when a child widget is removed from the group.
- `int * sizes ()`
Returns the internal array of widget sizes and positions.
- `void update_child (FL_Widget &widget) const`
Draws a child only if it needs it.

Protected Member Functions inherited from FL_Widget

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const`
Draws a focus box for the widget at the given position and size.
- `void draw_label () const`
Draws the widget's label at the defined label position.
- `void draw_label (int, int, int, int) const`
Draws the label in an arbitrary bounding box.
- `FL_Widget (int x, int y, int w, int h, const char *label=0L)`
Creates a widget at the given position and size.
- `unsigned int flags () const`
Gets the widget flags mask.
- `void h (int v)`
Internal use only.
- `void set_flag (unsigned int c)`
Sets a flag in the flags mask.
- `void w (int v)`

Internal use only.

- void **x** (int v)

Internal use only.

- void **y** (int v)

Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from **FI_Group**

- static **FI_Group** * **current** ()
Returns the currently active group.
- static void **current** (**FI_Group** *g)
Sets the current group.

Static Public Member Functions inherited from **FI_Widget**

- static void **default_callback** (**FI_Widget** *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from **FI_Widget**

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
 ,
OVERRIDE = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
 ,
MENU_WINDOW = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
 ,
GROUP_RELATIVE = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
 = 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }
flags possible values enumeration.

11.76.1 Detailed Description

A combination of the input widget and a menu button.

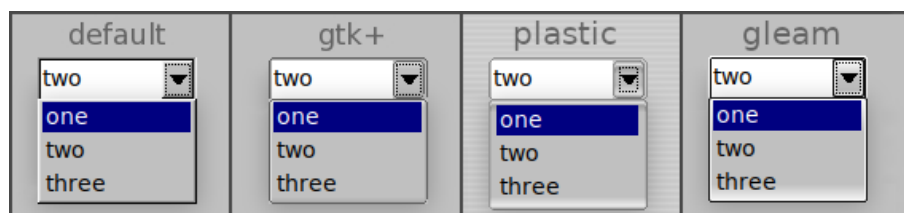


Figure 11.26 FI_Input_Choice widget

The user can either type into the input area, or use the menu button chooser on the right to choose an item which loads the input area with the selected text.

The application can directly access both the internal [Fl_Input](#) and [Fl_Menu_Button](#) widgets respectively using the [input\(\)](#) and [menubutton\(\)](#) accessor methods.

The default behavior is to invoke the [Fl_Input_Choice::callback\(\)](#) if the user changes the input field's contents, either by typing, pasting, or clicking a different item in the choice menu.

The callback can determine if an item was picked vs. typing into the input field by checking the value of [menubutton\(\)->changed\(\)](#), which will be:

- 1: the user picked a different item in the choice menu
- 0: the user typed or pasted directly into the input field

Example Use of Fl_Input_Choice

```
#include <stdio.h>
#include <FL/Fl.H>
#include <FL/Fl_Double_Window.H>
#include <FL/Fl_Input_Choice.H>
// Fl_Input_Choice callback()
void choice_cb(Fl_Widget *w, void *userdata) {
    // Show info about the picked item
    Fl_Input_Choice *choice = (Fl_Input_Choice*)w;
    printf("*** Choice Callback:\n");
    printf("    widget's text value='%s'\n", choice->value()); // normally all you need
    // Access the menu via menubutton()..
    const Fl_Menu_Item *item = choice->menubutton()->mvalue();
    printf("    item label()=' %s'\n", item ? item->label() : "(No item)");
    printf("    item value()=%d\n", choice->menubutton()->value());
    printf("    input value()=' %s'\n", choice->input()->value());
    printf("    The user %s\n", choice->menubutton()->changed()
        ? "picked a menu item"
        : "typed text");
}

int main() {
    Fl_Double_Window win(200,100,"Input Choice");
    win.begin();
    Fl_Input_Choice choice(10,10,100,30);
    choice.callback(choice_cb, 0);
    choice.add("Red");
    choice.add("Orange");
    choice.add("Yellow");
    //choice.value("Red"); // uncomment to make "Red" default
    win.end();
    win.show();
    return Fl::run();
}
```

Subclassing Example

One can subclass [Fl_Input_Choice](#) to override the virtual methods [inp_x/y/w/h\(\)](#) and [menu_x/y/w/h\(\)](#) to take control of the internal [Fl_Input](#) and [Fl_Menu_Button](#) widget positioning. In this example, input and menubutton's positions are swapped:

```
#include <FL/Fl.H>
#include <FL/Fl_Window.H>
#include <FL/Fl_Input_Choice.H>

class MyInputChoice : public Fl_Input_Choice {
protected:
    virtual int inp_x() const { return x() + Fl::box_dx(box()) + menu_w(); } // override to reposition
    virtual int menu_x() const { return x() + Fl::box_dx(box()); } // override to reposition
public:
    MyInputChoice(int X,int Y,int W,int H,const char*L=0) : Fl_Input_Choice(X,Y,W,H,L) {
        resize(X,Y,W,H); // necessary for ctor to trigger our overrides
    }
};

int main(int argc, char **argv) {
    Fl_Window *win = new Fl_Window(400,300);
    MyInputChoice *mychoice = new MyInputChoice(150,40,150,25,"Right Align Input");
    mychoice->add("Aaa");
    mychoice->add("Bbb");
    mychoice->add("Ccc");
    win->end();
    win->resizable(win);
    win->show();
    return Fl::run();
}
```

11.76.2 Constructor & Destructor Documentation

11.76.2.1 Fl_Input_Choice()

```
Fl_Input_Choice::Fl_Input_Choice (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Input_Choice](#) widget using the given position, size, and label string.
Inherited destructor destroys the widget and any values associated with it.

11.76.3 Member Function Documentation

11.76.3.1 add()

```
void Fl_Input_Choice::add (
    const char * s) [inline]
```

Adds an item to the menu.

When any item is selected, the [Fl_Input_Choice callback\(\)](#) is invoked, which can do something with the selected item.

You can access the more complex [Fl_Menu_Button::add\(\)](#) methods (setting item-specific callbacks, userdata, etc), via [menubutton\(\)](#). Example:

```
Fl_Input_Choice *choice = new Fl_Input_Choice(100,10,120,25,"Fonts");
Fl_Menu_Button *mb = choice->menubutton(); // use Fl_Input_Choice's Fl_Menu_Button
mb->add("Helvetica", 0, MyFont_CB, (void*)mydata); // use Fl_Menu_Button's add() methods
mb->add("Courier", 0, MyFont_CB, (void*)mydata);
mb->add("More..", 0, FontDialog_CB, (void*)mydata);
```

11.76.3.2 draw()

```
void Fl_Input_Choice::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw(); // calls Fl_Scrollbar::draw()
```

Reimplemented from [Fl_Group](#).

11.76.3.3 inp_x()

```
virtual int Fl_Input_Choice::inp_x () const [inline], [protected], [virtual]
```

The methods [inp_x\(\)](#), [inp_y\(\)](#), [inp_w\(\)](#) and [inp_h\(\)](#) return the desired position and size of the internal [Fl_Input](#) widget. These can be overridden by a subclass to redefine positioning. See code example in the Description for subclassing details.

11.76.3.4 input()

```
Fl_Input * Fl_Input_Choice::input () [inline]
```

Returns a pointer to the internal [Fl_Input](#) widget.

This can be used to directly access all of the [Fl_Input](#) widget's methods.

11.76.3.5 menu_x()

```
virtual int Fl_Input_Choice::menu_x () const [inline], [protected], [virtual]
```

The methods [menu_x\(\)](#), [menu_y\(\)](#), [menu_w\(\)](#) and [menu_h\(\)](#) return the desired position and size of the internal [Fl_Menu_Button](#) widget.

These can be overridden by a subclass to redefine positioning. See code example in the Description for subclassing details.

11.76.3.6 menubutton()

`Fl_Menu_Button * Fl_Input_Choice::menubutton () [inline]`

Returns a pointer to the internal `Fl_Menu_Button` widget.

This can be used to access any of the methods of the menu button, e.g.

```
Fl_Input_Choice *choice = new Fl_Input_Choice(100,10,120,25,"Choice:");
[...]
```

```
// Print all the items in the choice menu
for ( int t=0; t<choice->menubutton()->size(); t++ ) {
    const Fl_Menu_Item &item = choice->menubutton()->menu()[t];
    printf("item %d -- label=%s\n", t, item.label() ? item.label() : "(Null)");
}
```

11.76.3.7 resize()

```
void Fl_Input_Choice::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Resizes the `Fl_Input_Choice` widget.

Reimplemented from `Fl_Group`.

11.76.3.8 update_menubutton()

`int Fl_Input_Choice::update_menubutton ()`

Updates the menubutton with the string value in `Fl_Input`.

If the string value currently in `Fl_Input` matches one of the menu items in `menubutton()`, that menu item will become the current item selected.

Call this method after setting `value(const char*)` if you need the `menubutton()` to be synchronized with the `Fl_Input` field.

```
// Add items
choice->add(".25");
choice->add(".50");
choice->add("1.0");
choice->add("2.0");
choice->add("4.0");

choice->value("1.0");           // sets Fl_Input to "1.0"
choice->update_menubutton();    // cause menubutton to reflect this value too
                                // (returns 1 if match was found, 0 if not)

// Verify menubutton()'s value.
printf("menu button choice index=%d, value=%s\n",
        choice->menubutton()->value(),    // would be -1 if update not done
        choice->menubutton()->text());    // would be NULL if update not done
```

Returns

1 if a matching menuitem was found and value set, 0 if not.

Version

1.4.0

11.76.3.9 value() [1/2]

```
void Fl_Input_Choice::value (
    const char * val) [inline]
```

Sets the `Fl_Input` text field's contents to `val`.

Note it is possible to set the `value()` to one that is not in the menubutton's list of choices.

Setting the `value()` does NOT affect the menubutton's selection. If that's needed, call `update_menubutton()` after setting `value()`.

See also

`void value(int val), update_menubutton()`

11.76.3.10 value() [2/2]

```
void Fl_Input_Choice::value (
    int val)
```

Chooses item# `val` in the menu, and sets the `Fl_Input` text field to that value. Any previous text is cleared.

See also

```
void value(const char *val)
```

The documentation for this class was generated from the following files:

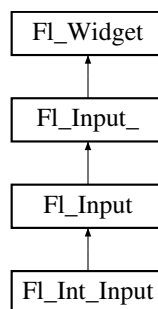
- `Fl_Input_Choice.H`
- `Fl_Input_Choice.cxx`

11.77 Fl_Int_Input Class Reference

The `Fl_Int_Input` class is a subclass of `Fl_Input` that only allows the user to type decimal digits (or hex numbers of the form `0xae`).

```
#include <Fl_Int_Input.H>
```

Inheritance diagram for `Fl_Int_Input`:



Public Member Functions

- `Fl_Int_Input` (int X, int Y, int W, int H, const char *l=0)
Creates a new `Fl_Int_Input` widget using the given position, size, and label string.

Public Member Functions inherited from `Fl_Input`

- `Fl_Input` (int, int, int, int, const char *l=0)
Creates a new `Fl_Input` widget using the given position, size, and label string.
- int `handle` (int) `FL_OVERRIDE`
Handles the specified event.

Public Member Functions inherited from `Fl_Input_`

- int `append` (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool `can_redo` () const
Check if there is a redo action available.
- bool `can_undo` () const
Check if the last operation can be undone.
- int `copy` (int clipboard)
Put the current selection into the clipboard.
- int `copy_cuts` ()

- Copies the yank buffer to the clipboard.*
- `Fl_Color cursor_color () const`
 - Gets the color of the cursor.*
- `void cursor_color (Fl_Color n)`
 - Sets the color of the cursor.*
- `int cut ()`
 - Deletes the current selection.*
- `int cut (int a, int b)`
 - Deletes all characters between index *a* and *b*.*
- `int cut (int n)`
 - Deletes the next *n* bytes rounded to characters before or after the cursor.*
- `double dvalue () const`
 - Returns the widget text interpreted as a floating point number.*
- `Fl_Input_ (int, int, int, int, const char *=0)`
 - Creates a new *Fl_Input_* widget.*
- `unsigned int index (int i) const`
 - Returns the character at index *i*.*
- `int input_type () const`
 - Gets the input field type.*
- `void input_type (int t)`
 - Sets the input field type.*
- `int insert (const char *t, int l=0)`
 - Inserts text at the cursor position.*
- `int insert_position () const`
 - Gets the position of the text cursor.*
- `int insert_position (int p)`
 - Sets the cursor position and mark.*
- `int insert_position (int p, int m)`
 - Sets the index for the cursor and mark.*
- `int ivalue () const`
 - Returns the widget text interpreted as a signed integer.*
- `int mark () const`
 - Gets the current selection mark.*
- `int mark (int m)`
 - Sets the current selection mark.*
- `int maximum_size () const`
 - Gets the maximum length of the input field in characters.*
- `void maximum_size (int m)`
 - Sets the maximum length of the input field in characters.*
- `int position () const`
- `int position (int p)`
- `int position (int p, int m)`
- `int readonly () const`
 - Gets the read-only state of the input field.*
- `void readonly (int b)`
 - Sets the read-only state of the input field.*
- `int redo ()`
 - Redo previous undo operation.*
- `int replace (int b, int e, const char *text, int ilen=0)`
 - Deletes text from *b* to *e* and inserts the new string *text*.*
- `void resize (int, int, int, int) FL_OVERRIDE`

- Changes the size of the widget.*

 - int [shortcut](#) () const

Return the shortcut key associated with this widget.
- void [shortcut](#) (int s)

Sets the shortcut key associated with this widget.
- int [size](#) () const

Returns the number of bytes in [value\(\)](#).
- void [size](#) (int W, int H)

Sets the width and height of this widget.
- int [static_value](#) (const char *)

Changes the widget text.
- int [static_value](#) (const char *, int)

Changes the widget text.
- int [tab_nav](#) () const

Gets whether the Tab key causes focus navigation in multiline input fields or not.
- void [tab_nav](#) (int val)

Sets whether the Tab key does focus navigation, or inserts tab characters into [Fl_Multiline_Input](#).
- [Fl_Color](#) [textcolor](#) () const

Gets the color of the text in the input field.
- void [textcolor](#) ([Fl_Color](#) n)

Sets the color of the text in the input field.
- [Fl_Font](#) [textfont](#) () const

Gets the font of the text in the input field.
- void [textfont](#) ([Fl_Font](#) s)

Sets the font of the text in the input field.
- [Fl_Fontsize](#) [textsize](#) () const

Gets the size of the text in the input field.
- void [textsize](#) ([Fl_Fontsize](#) s)

Sets the size of the text in the input field.
- int [undo](#) ()

Undoes previous changes to the text buffer.
- const char * [value](#) () const

Returns the text displayed in the widget.
- int [value](#) (const char *)

Changes the widget text.
- int [value](#) (const char *, int)

Changes the widget text.
- int [value](#) (double value)

Changes the widget text to a floating point number ("%g").
- int [value](#) (int value)

Changes the widget text to a signed integer number.
- int [wrap](#) () const

Gets the word wrapping state of the input field.
- void [wrap](#) (int b)

Sets the word wrapping state of the input field.
- [~Fl_Input_](#) ()

Destroys the widget.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (**FI_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FI_Gl_Window** * **as_gl_window** ()
Returns an FI_Gl_Window pointer if this widget is an FI_Gl_Window.
- virtual class **FI_Gl_Window** const * **as_gl_window** () const
- virtual **FI_Group** * **as_group** ()
Returns an FI_Group pointer if this widget is an FI_Group.
- virtual **FI_Group** const * **as_group** () const
- virtual **FI_Window** * **as_window** ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual **FI_Window** const * **as_window** () const
- void **bind_deimage** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- **FI_Boxtype box** () const
Gets the box type of the widget.
- void **box** (**FI_Boxtype** new_box)
Sets the box type for the widget.
- **FI_Callback_p callback** () const
Gets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb, **FI_Callback_User_Data** *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void **callback** (**FI_Callback** *cb, void *p)
Sets the current callback function and data for the widget.
- void **callback** (**FI_Callback0** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback1** *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int **changed** () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FL_Color` bg)

Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)

Sets the background and selection color of the widget.
- `FL_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.

- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)

Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [Fl_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([Fl_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- [Fl_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

Gets the current tooltip text.

- void **tooltip** (const char *text)
Sets the current tooltip text.
- **Fl_Window** * **top_window** () const
Returns a pointer to the top-level window for the widget.
- **Fl_Window** * **top_window_offset** (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- **uchar** **type** () const
Gets the widget type.
- void **type** (**uchar** t)
Sets the widget type.
- int **use_accents_menu** ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const
Gets the user data for this widget.
- void **user_data** (**Fl_Callback_User_Data** *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **Fl_When** **when** () const
Returns the conditions under which the callback is called.
- void **when** (**uchar** i)
Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~Fl_Widget** ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Static Public Attributes inherited from [Fl_Input](#)

- static const char * [copy_menu_text](#) = "Copy"
[this text may be customized at run-time]
- static const char * [cut_menu_text](#) = "Cut"
[this text may be customized at run-time]
- static const char * [paste_menu_text](#) = "Paste"
[this text may be customized at run-time]

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from [Fl_Input](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- int [handle_key](#) ()
Handles a keystroke.
- int [handle_rmb](#) ()
Handle right mouse button down events.

Protected Member Functions inherited from [Fl_Input_](#)

- int [apply_undo](#) ()
Apply the current undo/redo operation.
- void [drawtext](#) (int, int, int, int)
Draws the text in the passed bounding box.
- void [drawtext](#) (int, int, int, int, bool draw_active)
Draws the text in the passed bounding box.

- void **handle_mouse** (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int **handletext** (int e, int, int, int, int)
Handles all kinds of text field related events.
- int **line_end** (int i) const
Finds the end of a line.
- int **line_start** (int i) const
Finds the start of a line.
- int **linesPerPage** ()
- void **maybe_do_callback** (FL_Callback_Reason reason=FL_REASON_UNKNOWN)
- int **up_down_position** (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int **word_end** (int i) const
Finds the end of a word.
- int **word_start** (int i) const
Finds the start of a word.
- int **xscroll** () const
- int **yscroll** () const
- void **yscroll** (int yOffset)

Protected Member Functions inherited from FL_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FL_Boxtype t, FL_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FL_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FL_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)

Internal use only.

- void `x` (int v)

Internal use only.

- void `y` (int v)

Internal use only.

11.77.1 Detailed Description

The `Fl_Int_Input` class is a subclass of `Fl_Input` that only allows the user to type decimal digits (or hex numbers of the form 0xae).

11.77.2 Constructor & Destructor Documentation

11.77.2.1 Fl_Int_Input()

```
Fl_Int_Input::Fl_Int_Input (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new `Fl_Int_Input` widget using the given position, size, and label string.

The default boxtype is `FL_DOWN_BOX`.

Inherited destructor destroys the widget and any value associated with it.

The documentation for this class was generated from the following files:

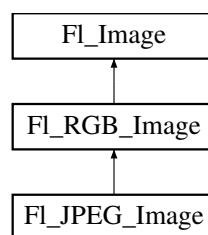
- `Fl_Int_Input.H`
- `Fl_Input.cxx`

11.78 Fl_JPEG_Image Class Reference

The `Fl_JPEG_Image` class supports loading, caching, and drawing of Joint Photographic Experts Group (JPEG) File Interchange Format (JFIF) images.

```
#include <Fl_JPEG_Image.H>
```

Inheritance diagram for `Fl_JPEG_Image`:



Public Member Functions

- `Fl_JPEG_Image` (const char *filename)
The constructor loads the JPEG image from the given jpeg filename.
- `Fl_JPEG_Image` (const char *name, const unsigned char *data, int data_length=-1)
The constructor loads the JPEG image from memory.

Public Member Functions inherited from Fl_RGB_Image

- virtual `Fl_SVG_Image * as_svg_image` ()
Returns whether an image is an Fl_SVG_Image or not.
- int `cache_h` ()

- int **cache_w** ()
- void **color_average** (FI_Color c, float i) **FL_OVERRIDE**
*The **color_average()** method averages the colors in the image with the provided FLTK color value.*
- FI_Image * **copy** () const
- FI_Image * **copy** (int W, int H) const **FL_OVERRIDE**
Creates a resized copy of the image.
- void **desaturate** () **FL_OVERRIDE**
*The **desaturate()** method converts an image to grayscale.*
- void **draw** (int X, int Y)
- void **draw** (int X, int Y, int W, int H, int cx=0, int cy=0) **FL_OVERRIDE**
Draws the image to the current drawing surface with a bounding box.
- FI_RGB_Image (const FI_Pixmap *pxm, FI_Color bg=FL_GRAY)
The constructor creates a new RGBA image from the specified FI_Pixmap.
- FI_RGB_Image (const uchar *bits, int bits_length, int W, int H, int D, int LD)
The constructor creates a new image from the specified data.
- FI_RGB_Image (const uchar *bits, int W, int H, int D=3, int LD=0)
The constructor creates a new image from the specified data.
- void **label** (FI_Menu_Item *m) **FL_OVERRIDE**
This method is an obsolete way to set the image attribute of a menu item.
- void **label** (FI_Widget *w) **FL_OVERRIDE**
This method is an obsolete way to set the image attribute of a widget or menu item.
- virtual void **normalize** ()
Makes sure the object is fully initialized.
- void **uncache** () **FL_OVERRIDE**
If the image has been cached for display, delete the cache data.
- ~FI_RGB_Image () **FL_OVERRIDE**
The destructor frees all memory and server resources that are used by the image.

Public Member Functions inherited from FI_Image

- virtual class FI_Shared_Image * **as_shared_image** ()
Returns whether an image is an FI_Shared_Image or not.
- FI_Image * **copy** () const
Creates a copy of the image in the same size.
- int **count** () const
Returns the number of data values associated with the image.
- int **d** () const
Returns the image depth.
- const char *const * **data** () const
Returns a pointer to the current image data array.
- int **data_h** () const
Returns the height of the image data.
- int **data_w** () const
Returns the width of the image data.
- void **draw** (int X, int Y)
Draws the image to the current drawing surface.
- int **fail** () const
Returns a value that is not 0 if there is currently no image available.
- FI_Image (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int **h** () const

- Returns the current image drawing height in FLTK units.*

 - void **inactive** ()

*The **inactive()** method calls **color_average(FL_BACKGROUND_COLOR, 0.33f)** to produce an image that appears grayed out.*
- int **ld** () const

Returns the current line data size in bytes.
- virtual void **release** ()

*Releases an **FI_Image** - the same as 'delete this'.*
- virtual void **scale** (int width, int height, int proportional=1, int can_expand=0)

Sets the drawing size of the image.
- int **w** () const

Returns the current image drawing width in FLTK units.
- virtual ~**FI_Image** ()

The destructor is a virtual method that frees all memory used by the image.

Protected Member Functions

- void **load_jpg_** (const char *filename, const char *sharename, const unsigned char *data, int data_length=-1)

Protected Member Functions inherited from **FI_Image**

- void **d** (int D)
- Sets the current image depth.*
- void **data** (const char *const *p, int c)
- Sets the current data pointer and count of pointers in the array.*
- void **draw_empty** (int X, int Y)
- The protected method **draw_empty()** draws a box with an X in it.*
- int **draw_scaled** (int X, int Y, int W, int H)
- Draw the image to the current drawing surface rescaled to a given width and height.*
- void **h** (int H)
- Sets the height of the image data.*
- void **ld** (int LD)
- Sets the current line data size in bytes.*
- void **w** (int W)
- Sets the width of the image data.*

Additional Inherited Members

Static Public Member Functions inherited from **FI_RGB_Image**

- static size_t **max_size** ()
- Returns the maximum allowed image size in bytes when creating an **FI_RGB_Image** object.*
- static void **max_size** (size_t size)
- Sets the maximum allowed image size in bytes when creating an **FI_RGB_Image** object.*

Static Public Member Functions inherited from **FI_Image**

- static **FI_Labeltype** **define_FL_IMAGE_LABEL** ()
 - static **FI_RGB_Scaling** **RGB_scaling** ()
- Returns the currently used RGB image scaling method.*
- static void **RGB_scaling** (**FI_RGB_Scaling**)
- Sets the RGB image scaling method used for copy(int, int).*
- static **FI_RGB_Scaling** **scaling_algorithm** ()

Gets what algorithm is used when resizing a source image to draw it.

- static void [scaling_algorithm](#) ([FI_RGB_Scaling](#) algorithm)

Sets what algorithm is used when resizing a source image to draw it.

Public Attributes inherited from [FI_RGB_Image](#)

- int **alloc_array**

If non-zero, the object's data array is delete[]'d when deleting the object.

- const [uchar](#) * **array**

Points to the start of the object's data array.

Static Public Attributes inherited from [FI_Image](#)

- static const int **ERR_FILE_ACCESS** = -2
- static const int **ERR_FORMAT** = -3
- static const int **ERR_MEMORY_ACCESS** = -4
- static const int **ERR_NO_IMAGE** = -1
- static bool **register_images_done** = false

True after [fl_register_images\(\)](#) was called, false before.

Static Protected Member Functions inherited from [FI_Image](#)

- static void **labeltype** (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void **measure** (const [FI_Label](#) *lo, int &lw, int &lh)

11.78.1 Detailed Description

The [FI_JPEG_Image](#) class supports loading, caching, and drawing of Joint Photographic Experts Group (JPEG) File Interchange Format (JFIF) images.

The class supports grayscale and color (RGB) JPEG image files.

11.78.2 Constructor & Destructor Documentation

11.78.2.1 [FI_JPEG_Image\(\)](#) [1/2]

```
FI_JPEG_Image::FI_JPEG_Image (
    const char * filename)
```

The constructor loads the JPEG image from the given jpeg filename.

The inherited destructor frees all memory and server resources that are used by the image.

Use [FI_Image::fail\(\)](#) to check if [FI_JPEG_Image](#) failed to load. [fail\(\)](#) returns `ERR_FILE_ACCESS` if the file could not be opened or read, `ERR_FORMAT` if the JPEG format could not be decoded, and `ERR_NO_IMAGE` if the image could not be loaded for another reason. If the image has loaded correctly, [w\(\)](#), [h\(\)](#), and [d\(\)](#) should return values greater than zero.

Parameters

<code>in</code>	<code>filename</code>	a full path and name pointing to a valid jpeg file.

See also

[FI_JPEG_Image::FI_JPEG_Image\(const char *imagename, const unsigned char *data\)](#)

11.78.2.2 FI_JPEG_Image() [2/2]

```
Fl_JPEG_Image::Fl_JPEG_Image (
    const char * name,
    const unsigned char * data,
    int data_length = -1)
```

The constructor loads the JPEG image from memory.

Construct an image from a block of memory inside the application. Fluid offers "binary Data" chunks as a great way to add image data into the C++ source code. `name_png` can be NULL. If a name is given, the image is added to the list of shared images (see: [Fl_Shared_Image](#)) and will be available by that name.

The inherited destructor frees all memory and server resources that are used by the image.

Use [Fl_Image::fail\(\)](#) to check if [Fl_JPEG_Image](#) failed to load. [fail\(\)](#) returns `ERR_FILE_ACCESS` if the file could not be opened or read, `ERR_FORMAT` if the JPEG format could not be decoded, and `ERR_NO_IMAGE` if the image could not be loaded for another reason. If the image has loaded correctly, `w()`, `h()`, and `d()` should return values greater than zero.

Parameters

<i>name</i>	A unique name or NULL
<i>data</i>	A pointer to the memory location of the JPEG image
<i>data_length</i>	optional length of <code>data</code> . This will protect memory outside of the <code>data</code> array from illegal read operations

See also

[Fl_JPEG_Image::Fl_JPEG_Image\(const char *filename\)](#)
[Fl_Shared_Image](#)

The documentation for this class was generated from the following files:

- [Fl_JPEG_Image.H](#)
- [Fl_JPEG_Image.cxx](#)

11.79 FI_Label Struct Reference

This struct stores all information for a text or mixed graphics label.

```
#include <Fl_Widget.H>
```

Public Member Functions

- void [draw](#) (int, int, int, int, [Fl_Align](#)) const
Draws the label aligned to the given box.
- void [measure](#) (int &w, int &h) const
Measures the size of the label.

Public Attributes

- [Fl_Align](#) **align_**
alignment of label
- [Fl_Color](#) **color**
text color
- [Fl_Image](#) * **deimage**
optional image for a deactivated label
- [Fl_Font](#) **font**
label font used in text
- signed char **h_margin_**

- Spacing between label and the horizontally aligned side of the widget.*
- `Fl_Image * image`
optional image for an active label
- `Fl_Fontsize size`
size of label font
- `uchar spacing`
Spacing between an image and the label text.
- `uchar type`
type of label.
- signed char `v_margin_`
Spacing between label and the vertically aligned side of the widget.
- `const char * value`
label text

11.79.1 Detailed Description

This struct stores all information for a text or mixed graphics label.

Todo There is an aspiration that the `Fl_Label` type will become a widget by itself. That way we will be avoiding a lot of code duplication by handling labels in a similar fashion to widgets containing text. We also provide an easy interface for very complex labels, containing html or vector graphics. However, this re-factoring is not in place in this release.

11.79.2 Member Function Documentation

11.79.2.1 draw()

```
void Fl_Label::draw (
    int X,
    int Y,
    int W,
    int H,
    Fl_Align align) const
```

Draws the label aligned to the given box.

Draws a label with arbitrary alignment in an arbitrary box.

11.79.2.2 measure()

```
void Fl_Label::measure (
    int & W,
    int & H) const
```

Measures the size of the label.

Parameters

<code>in, out</code>	<code>W,H</code>	: this is the requested size for the label text plus image; on return, this will contain the size needed to fit the label

11.79.3 Member Data Documentation

11.79.3.1 type

`uchar Fl_Label::type`
type of label.

See also

[FI_Labeltype](#)

The documentation for this struct was generated from the following files:

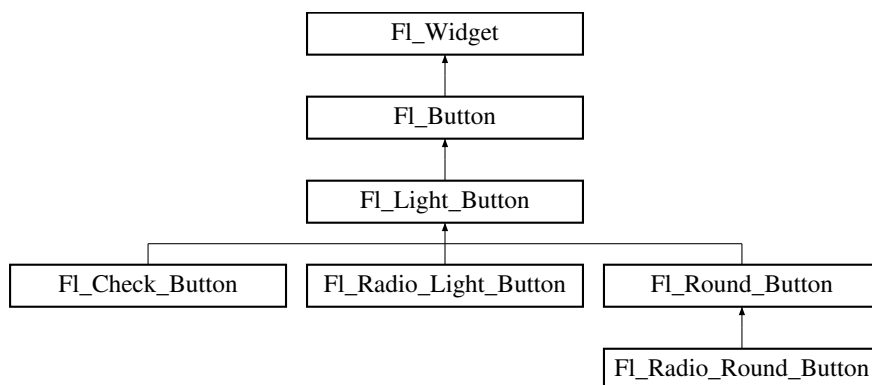
- [FI_Widget.H](#)
- [fl_labeltype.cxx](#)

11.80 FI_Light_Button Class Reference

This subclass displays the "on" state by turning on a light, rather than drawing pushed in.

```
#include <FI_Light_Button.H>
```

Inheritance diagram for FI_Light_Button:



Public Member Functions

- [FI_Light_Button](#) (int *x*, int *y*, int *w*, int *h*, const char **l*=0)
Creates a new [FI_Light_Button](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [FI_Button](#)

- int [clear](#) ()
Same as `value(0)`.
- [uchar compact](#) ()
Return true if buttons are rendered as compact buttons.
- void [compact](#) ([uchar](#) *v*)
Decide if buttons should be rendered in compact mode.
- [FI_Boxtype down_box](#) () const
Returns the current down box type, which is drawn when `value()` is non-zero.
- void [down_box](#) ([FI_Boxtype](#) *b*)
Sets the down box type.
- [FI_Color down_color](#) () const
(for backwards compatibility)
- void [down_color](#) (unsigned *c*)
(for backwards compatibility)
- [FI_Button](#) (int *X*, int *Y*, int *W*, int *H*, const char **L*=0)
The constructor creates the button using the given position, size, and label.
- int [set](#) ()
Same as `value(1)`.

- void **setonly** ()
Turns on this button and turns off all other radio buttons in the group (calling `value (1)` or `set ()` does not do this).
- int **shortcut** () const
Returns the current shortcut key for the button.
- void **shortcut** (const char *s)
(for backwards compatibility)
- void **shortcut** (int s)
Sets the shortcut key to `s`.
- char **value** () const
Returns the current value of the button (0 or 1).
- int **value** (int v)
Sets the current value of the button.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- FI_Align **align** () const
Gets the label alignment.
- void **align** (FI_Align alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class FI_Gl_Window * **as_gl_window** ()
Returns an FI_Gl_Window pointer if this widget is an FI_Gl_Window.
- virtual class FI_Gl_Window const * **as_gl_window** () const
- virtual FI_Group * **as_group** ()
Returns an FI_Group pointer if this widget is an FI_Group.
- virtual FI_Group const * **as_group** () const
- virtual FI_Window * **as_window** ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual FI_Window const * **as_window** () const
- void **bind_deimage** (FI_Image *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (FI_Image *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- FI_Boxtype **box** () const
Gets the box type of the widget.
- void **box** (FI_Boxtype new_box)

- Sets the box type for the widget.*

 - [FI_Callback_p](#) [callback](#) () const
- Gets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, void *p)
- Sets the current callback function and data for the widget.*

 - void [callback](#) ([FI_Callback0](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)
- Sets the current callback function for the widget.*

 - unsigned int [changed](#) () const
- Checks if the widget value changed since the last callback.*

 - void [clear_active](#) ()
- Marks the widget as inactive without sending events or changing focus.*

 - void [clear_changed](#) ()
- Marks the value of the widget as unchanged.*

 - void [clear_damage](#) (uchar c=0)
- Clears or sets the damage flags.*

 - void [clear_output](#) ()
- Sets a widget to accept input.*

 - void [clear_visible](#) ()
- Hides the widget.*

 - void [clear_visible_focus](#) ()
- Disables keyboard focus navigation with this widget.*

 - [FI_Color](#) [color](#) () const
- Gets the background color of the widget.*

 - void [color](#) ([FI_Color](#) bg)
- Sets the background color of the widget.*

 - void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
- Sets the background and selection color of the widget.*

 - [FI_Color](#) [color2](#) () const
- For back compatibility only.*

 - void [color2](#) (unsigned a)
- For back compatibility only.*

 - int [contains](#) (const [FI_Widget](#) *w) const
- Checks if w is a child of this widget.*

 - void [copy_label](#) (const char *new_label)
- Sets the current label.*

 - void [copy_tooltip](#) (const char *text)
- Sets the current tooltip text.*

 - [uchar](#) [damage](#) () const
- Returns non-zero if [draw\(\)](#) needs to be called.*

 - void [damage](#) ([uchar](#) c)
- Sets the damage bits for the widget.*

 - void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int [damage_resize](#) (int, int, int, int)
- Internal use only.*

- void [deactivate](#) ()
Deactivates the widget.
- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)

- Set the gap between the label and the image in pixels.*

 - [FI_Color labelcolor](#) () const
 - Gets the label color.*
 - void [labelcolor](#) ([FI_Color](#) c)
 - Sets the label color.*
 - [FI_Font labelfont](#) () const
 - Gets the font to use.*
 - void [labelfont](#) ([FI_Font](#) f)
 - Sets the font to use.*
 - [FI_Fontsize labelsize](#) () const
 - Gets the font size in pixels.*
 - void [labelsiz](#) ([FI_Fontsize](#) pix)
 - Sets the font size in pixels.*
 - [FI_Labeltype labeltype](#) () const
 - Gets the label type.*
 - void [labeltype](#) ([FI_Labeltype](#) a)
 - Sets the label type.*
 - void [measure_label](#) (int &ww, int &hh) const
 - Sets width ww and height hh accordingly with the label size.*
 - bool [needs_keyboard](#) () const
 - Returns whether this widget needs a keyboard.*
 - void [needs_keyboard](#) (bool needs)
 - Sets whether this widget needs a keyboard.*
 - unsigned int [output](#) () const
 - Returns if a widget is used for output only.*
 - [FI_Group * parent](#) () const
 - Returns a pointer to the parent widget.*
 - void [parent](#) ([FI_Group](#) *p)
 - Internal use only - "for hacks only".*
 - void [position](#) (int X, int Y)
 - Repositions the window or widget.*
 - void [redraw](#) ()
 - Schedules the drawing of the widget.*
 - void [redraw_label](#) ()
 - Schedules the drawing of the label.*
 - virtual void [resize](#) (int x, int y, int w, int h)
 - Changes the size or position of the widget.*
 - [FI_Color selection_color](#) () const
 - Gets the selection color.*
 - void [selection_color](#) ([FI_Color](#) a)
 - Sets the selection color.*
 - void [set_active](#) ()
 - Marks the widget as active without sending events or changing focus.*
 - void [set_changed](#) ()
 - Marks the value of the widget as changed.*
 - void [set_output](#) ()
 - Sets a widget to output only.*
 - void [set_visible](#) ()
 - Makes the widget visible.*
 - void [set_visible_focus](#) ()
 - Enables keyboard focus navigation with this widget.*

- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [FI_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)

- *Sets the flags used to decide when a callback is called.*
- `FL_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~FL_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from `FL_Button`

- `void simulate_key_action ()`

Protected Member Functions inherited from `FL_Widget`

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const`
Draws a focus box for the widget at the given position and size.
- `void draw_label () const`
Draws the widget's label at the defined label position.
- `void draw_label (int, int, int, int) const`
Draws the label in an arbitrary bounding box.
- `FL_Widget (int x, int y, int w, int h, const char *label=0L)`
Creates a widget at the given position and size.
- `unsigned int flags () const`
Gets the widget flags mask.
- `void h (int v)`
Internal use only.
- `void set_flag (unsigned int c)`
Sets a flag in the flags mask.
- `void w (int v)`
Internal use only.
- `void x (int v)`
Internal use only.
- `void y (int v)`
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from Fl_Widget

- static void `default_callback` (`Fl_Widget *widget`, `void *data`)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (`const char *t`)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (`const char *`, `const bool require_alt=false`)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from Fl_Widget

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
 ,
`OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
 ,
`MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
 ,
`GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
 = 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Static Protected Member Functions inherited from Fl_Button

- static void `key_release_timeout` (`void *`)

Static Protected Attributes inherited from Fl_Button

- static `Fl_Widget_Tracker *` `key_release_tracker` = 0

11.80.1 Detailed Description

This subclass displays the "on" state by turning on a light, rather than drawing pushed in.

The shape of the "light" is initially set to `FL_DOWN_BOX`. The color of the light when on is controlled with `selection_color()`, which defaults to `FL_YELLOW`.

Buttons generate callbacks when they are clicked by the user. You control exactly when and how by changing the values for `type()` and `when()`.

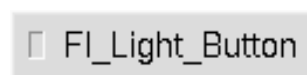


Figure 11.27 Fl_Light_Button

11.80.2 Constructor & Destructor Documentation

11.80.2.1 Fl_Light_Button()

```
Fl_Light_Button::Fl_Light_Button (
    int X,
    int Y,
    int W,
```

```
int H,
const char * l = 0)
```

Creates a new [Fl_Light_Button](#) widget using the given position, size, and label string.

The default box type is `FL_UP_BOX` and the default down box type [down_box\(\)](#) is `FL_NO_BOX` (0).

The [selection_color\(\)](#) sets the color of the "light". Default is `FL_YELLOW`.

The default label alignment is `'FL_ALIGN_LEFT|FL_ALIGN_INSIDE'` so the label is drawn inside the button area right of the "light".

Note

Do not change the default box types of [Fl_Light_Button](#). The box types determine how the button is drawn. If you change the [down_box\(\)](#) type the drawing behavior is undefined.

11.80.3 Member Function Documentation

11.80.3.1 draw()

```
void Fl_Light_Button::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Reimplemented from [Fl_Button](#).

11.80.3.2 handle()

```
int Fl_Light_Button::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

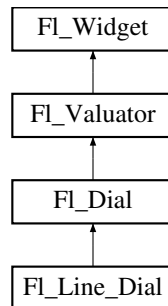
Reimplemented from [Fl_Button](#).

The documentation for this class was generated from the following files:

- Fl_Light_Button.H
- Fl_Light_Button.cxx

11.81 Fl_Line_Dial Class Reference

Inheritance diagram for Fl_Line_Dial:



Public Member Functions

- **Fl_Line_Dial** (int X, int Y, int W, int H, const char *L=0)

Public Member Functions inherited from **Fl_Dial**

- short **angle1** () const
Sets Or gets the angles used for the minimum and maximum values.
- void **angle1** (short a)
*See short **angle1()** const.*
- short **angle2** () const
*See short **angle1()** const.*
- void **angle2** (short a)
*See short **angle1()** const.*
- void **angles** (short a, short b)
*See short **angle1()** const.*
- **Fl_Dial** (int x, int y, int w, int h, const char *l=0)
*Creates a new **Fl_Dial** widget using the given position, size, and label string.*
- int **handle** (int) **FL_OVERRIDE**
Allow subclasses to handle event based on current position and size.

Public Member Functions inherited from **Fl_Valuator**

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)
Sets the maximum value for the valuator.

- double **minimum** () const
Gets the minimum value for the valuator.
- void **minimum** (double a)
Sets the minimum value for the valuator.
- void **precision** (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void **range** (double a, double b)
Sets the minimum and maximum values for the valuator.
- double **round** (double)
Round the passed value to the nearest step increment.
- double **step** () const
Gets or sets the step value.
- void **step** (double a, int b)
See double [FI_Valuator::step\(\)](#) const
- void **step** (double s)
See double [FI_Valuator::step\(\)](#) const.
- void **step** (int a)
See double [FI_Valuator::step\(\)](#) const
- double **value** () const
Gets the floating point(double) value.
- int **value** (double)
Sets the current value.
- **~FI_Valuator** () **FL_OVERRIDE**
Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) **align** () const
Gets the label alignment.
- void **align** ([FI_Align](#) alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * **as_gl_window** ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * **as_gl_window** () const
- virtual [FI_Group](#) * **as_group** ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * **as_group** () const
- virtual [FI_Window](#) * **as_window** ()

- Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).*

 - virtual [FL_Window](#) const * **as_window** () const
- void [bind_deimage](#) ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FL_Boxtype](#) **box** () const

Gets the box type of the widget.
- void [box](#) ([FL_Boxtype](#) new_box)

Sets the box type for the widget.
- [FL_Callback_p](#) **callback** () const

Gets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FL_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.
- void [callback](#) ([FL_Callback0](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()

Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.
- void [clear_output](#) ()

Sets a widget to accept input.
- void [clear_visible](#) ()

Hides the widget.
- void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.
- [FL_Color](#) **color** () const

Gets the background color of the widget.
- void [color](#) ([FL_Color](#) bg)

Sets the background color of the widget.
- void [color](#) ([FL_Color](#) bg, [FL_Color](#) sel)

Sets the background and selection color of the widget.
- [FL_Color](#) **color2** () const

For back compatibility only.
- void [color2](#) (unsigned a)

For back compatibility only.
- int [contains](#) (const [FL_Widget](#) *w) const

- Checks if w is a child of this widget.*

 - void [copy_label](#) (const char *new_label)

Sets the current label.
- void [copy_tooltip](#) (const char *text)

Sets the current tooltip text.
- [uchar damage](#) () const

Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) (uchar c)

Sets the damage bits for the widget.
- void [damage](#) (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)

Internal use only.
- void [deactivate](#) ()

Deactivates the widget.
- [FL_Image * deimage](#) ()

Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image * deimage](#) () const

Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)

Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const

Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))

Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))

Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))

Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const

Gets the widget height.
- virtual void [hide](#) ()

Makes a widget invisible.
- int [horizontal_label_margin](#) ()

Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)

Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image * image](#) ()

Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image * image](#) () const

Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)

Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const

Returns whether the image is managed by the widget.

- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)
Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

- Gets the selection color.*

 - void `selection_color` (`Fl_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.
- void `set_visible` ()

Makes the widget visible.
- void `set_visible_focus` ()

Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()

Makes a widget visible.
- void `size` (int W, int H)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const

Gets the current tooltip text.
- void `tooltip` (const char *text)

Sets the current tooltip text.
- `Fl_Window` * `top_window` () const

Returns a pointer to the top-level window for the widget.
- `Fl_Window` * `top_window_offset` (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- `uchar` `type` () const

Gets the widget type.
- void `type` (`uchar` t)

Sets the widget type.
- int `use_accents_menu` ()

Returns non zero if `MAC_USE_ACCENTS_MENU` flag is set, 0 otherwise.
- void * `user_data` () const

Gets the user data for this widget.
- void `user_data` (`Fl_Callback_User_Data` *v, bool auto_free)

Sets the user data for this widget.
- void `user_data` (void *v)

Sets the user data for this widget.
- int `vertical_label_margin` ()

Get the spacing between the label and the vertical edge of the widget.
- void `vertical_label_margin` (int px)

Set the spacing between the label and the vertical edge of the widget.

- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [FL_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) (uchar i)
Sets the flags used to decide when a callback is called.
- [FL_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual [~FL_Widget](#) ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from [FL_Widget](#)

- static void [default_callback](#) ([FL_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [FL_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from [FI_Dial](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draws dial at current position and size.
- void [draw](#) (int X, int Y, int W, int H)
Draws dial at given position and size.
- int [handle](#) (int event, int X, int Y, int W, int H)
Allows subclasses to handle event based on given position and size.

Protected Member Functions inherited from [FI_Valuator](#)

- [FI_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FI_Valuator](#) widget using the given position, size, and label string.
- void [handle_drag](#) (double newvalue)
Called during a drag operation, after an [FL_WHEN_CHANGED](#) event is received and before the callback.
- void [handle_push](#) ()
Stores the current value in the previous value.
- void [handle_release](#) ()
Called after an [FL_WHEN_RELEASE](#) event is received and before the callback.
- int [horizontal](#) () const
Tells if the valuator is an [FL_HORIZONTAL](#) one.
- double [previous_value](#) () const
Gets the previous floating point value before an event changed it.
- void [set_value](#) (double v)
Sets the current floating point value.
- double [softclamp](#) (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void [value_damage](#) ()
Asks for partial redraw.

Protected Member Functions inherited from [FI_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If [FL_ALIGN_IMAGE_BACKDROP](#) is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.
- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.

- `FI_Widget` (int `x`, int `y`, int `w`, int `h`, const char *`label`=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int `v`)
Internal use only.
- void `set_flag` (unsigned int `c`)
Sets a flag in the flags mask.
- void `w` (int `v`)
Internal use only.
- void `x` (int `v`)
Internal use only.
- void `y` (int `v`)
Internal use only.

The documentation for this class was generated from the following files:

- FI_Line_Dial.H
- FI_Dial.cxx

11.82 FI_Mac_App_Menu Class Reference

Static Public Member Functions

- static void `custom_application_menu_items` (const `FI_Menu_Item` *`m`)
Adds custom menu item(s) to the application menu of the system menu bar.

Static Public Attributes

- static const char * `about`
Localizable text for the "About xxx" application menu item.
- static const char * `hide`
Localizable text for the "Hide xxx" application menu item.
- static const char * `hide_others`
Localizable text for the "Hide Others" application menu item.
- static const char * `print`
Localizable text for the "Print Front Window" application menu item.
- static const char * `print_no_titlebar`
Localizable text for the "Print Front Window" application menu item.
- static const char * `quit`
Localizable text for the "Quit xxx" application menu item.
- static const char * `services`
Localizable text for the "Services" application menu item.
- static const char * `show`
Localizable text for the "Show All" application menu item.
- static const char * `toggle_print_titlebar`
Localizable text for the "Toggle print titlebar" application menu item.

11.82.1 Member Function Documentation

11.82.1.1 custom_application_menu_items()

```
static void Fl_Mac_App_Menu::custom_application_menu_items (  
    const Fl_Menu_Item * m) [static]
```

Adds custom menu item(s) to the application menu of the system menu bar.

They are positioned after the "Print Front Window / Toggle printing of titlebar" items, or at their place if they were removed with `Fl_Mac_App_Menu::print = ""`.

Parameters

<i>m</i>	zero-ending array of Fl_Menu_Item 's.

11.82.2 Member Data Documentation

11.82.2.1 print

```
const char* Fl_Mac_App_Menu::print [static]
```

Localizable text for the "Print Front Window" application menu item.

This menu item and next one won't be displayed if [Fl_Mac_App_Menu::print](#) is set to an empty string.

The documentation for this class was generated from the following file:

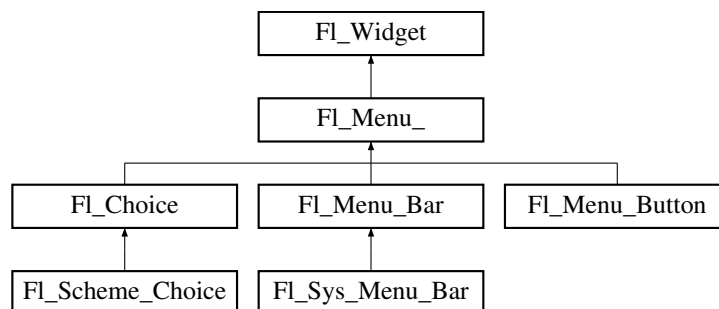
- [mac.H](#)

11.83 Fl_Menu_ Class Reference

Base class of all widgets that have a menu in FLTK.

```
#include <Fl_Menu_.H>
```

Inheritance diagram for Fl_Menu_:



Public Member Functions

- int [add](#) (const char *)
This is a Forms (and SGI GL library) compatible add function, it adds many menu items, with '|' separating the menu items, and tab separating the menu item names from an optional shortcut string.
- int [add](#) (const char *, int [shortcut](#), [Fl_Callback](#) *, void *=0, int=0)
Adds a new menu item.
- int [add](#) (const char *a, const char *b, [Fl_Callback](#) *c, void *d=0, int e=0)
See int [Fl_Menu_::add](#)(const char label, int shortcut, [Fl_Callback](#)*, void *user_data=0, int flags=0)*
- void [clear](#) ()
Same as menu(NULL), set the array pointer to null, indicating a zero-length menu.
- int [clear_submenu](#) (int index)
*Clears the specified submenu pointed to by *index* of all menu items.*
- void [copy](#) (const [Fl_Menu_Item](#) *m, void *user_data=0)
Sets the menu array pointer with a copy of m that will be automatically deleted.
- [Fl_Boxtype](#) [down_box](#) () const
This box type is used to surround the currently-selected items in the menus.
- void [down_box](#) ([Fl_Boxtype](#) b)
Sets the box type used to surround the currently-selected items in the menus.
- [Fl_Color](#) [down_color](#) () const
For back compatibility, same as [selection_color](#)()
- void [down_color](#) (unsigned c)

- For back compatibility, same as [selection_color\(\)](#)*

 - int [find_index](#) (const char *name) const
 - Find the menu item index for a given menu `pathname`, such as "Edit/Copy".*
 - int [find_index](#) (const [FI_Menu_Item](#) *item) const
 - Find the index into the menu array for a given `item`.*
 - int [find_index](#) ([FI_Callback](#) *cb) const
 - Find the index into the menu array for a given callback `cb`.*
 - const [FI_Menu_Item](#) * [find_item](#) (const char *name)
 - Find the menu item for a given menu `pathname`, such as "Edit/Copy".*
 - const [FI_Menu_Item](#) * [find_item](#) ([FI_Callback](#) *)
 - Find the menu item for the given callback `cb`.*
 - const [FI_Menu_Item](#) * [find_item_with_argument](#) (long)
 - Find the menu item for the given user argument `v`.*
 - const [FI_Menu_Item](#) * [find_item_with_user_data](#) (void *)
 - Find the menu item for the given user data `v`.*
 - [FI_Menu_](#) (int, int, int, int, const char *s=0)
 - Creates a new [FI_Menu_](#) widget using the given position, size, and label string.*
 - void [global](#) ()
 - Make the shortcuts for this menu work no matter what window has the focus when you type it.*
 - int [insert](#) (int index, const char *, int [shortcut](#), [FI_Callback](#) *, void *d=0, int e=0)
 - Inserts a new menu item at the specified `index` position.*
 - int [insert](#) (int index, const char *a, const char *b, [FI_Callback](#) *c, void *d=0, int e=0)
 - See int [FI_Menu_::insert](#)(const char* label, int shortcut, [FI_Callback](#)*, void *user_data=0, int flags=0)*
 - int [item_pathname](#) (char *name, int namelen, const [FI_Menu_Item](#) *finditem=0) const
 - Get the menu 'pathname' for the specified menuitem.*
 - const [FI_Menu_Item](#) * [menu](#) () const
 - Returns a pointer to the array of [FI_Menu_Items](#).*
 - void [menu](#) (const [FI_Menu_Item](#) *m)
 - Sets the menu array pointer directly.*
 - [FI_Boxtype](#) [menu_box](#) () const
 - Get the box type for the menu popup windows.*
 - void [menu_box](#) ([FI_Boxtype](#) b)
 - Set the box type for the menu popup windows.*
 - const [FI_Menu_Item](#) * [menu_end](#) ()
 - Finishes menu modifications and returns [menu\(\)](#).*
 - int [mode](#) (int i) const
 - Get the flags of item `i`.*
 - void [mode](#) (int i, int fl)
 - Set the flags of item `i`.*
 - const [FI_Menu_Item](#) * [mvalue](#) () const
 - Return a pointer to the last menu item that was picked.*
 - const [FI_Menu_Item](#) * [picked](#) (const [FI_Menu_Item](#) *)
 - When user picks a menu item, call this.*
 - const [FI_Menu_Item](#) * [prev_mvalue](#) () const
 - Return a pointer to the menu item that was picked before the current one was picked.*
 - void [remove](#) (int)
 - Deletes item `i` from the menu.*
 - void [replace](#) (int, const char *)
 - Changes the text of item `i`.*
 - void [setonly](#) ([FI_Menu_Item](#) *item)
 - Turns the radio item "on" for the menu item and turns "off" adjacent radio items of the same group.*

- void **shortcut** (int i, int s)
Change the shortcut of item i to s.
- int **size** () const
This returns the number of FI_Menu_Item structures that make up the menu, correctly counting submenus.
- void **size** (int W, int H)
- const FI_Menu_Item * **test_shortcut** ()
Returns the menu item with the entered shortcut (key value).
- const char * **text** () const
Returns the title of the last item chosen.
- const char * **text** (int i) const
Returns the title of item i.
- FI_Color **textcolor** () const
Get the current color of menu item labels.
- void **textcolor** (FI_Color c)
Sets the current color of menu item labels.
- FI_Font **textfont** () const
Gets the current font of menu item labels.
- void **textfont** (FI_Font c)
Sets the current font of menu item labels.
- FI_Fontsize **textsize** () const
Gets the font size of menu item labels.
- void **textsize** (FI_Fontsize c)
Sets the font size of menu item labels.
- int **value** () const
Return the index into the menu() of the last item chosen by the user.
- int **value** (const FI_Menu_Item *)
Set the value of a menu to the menu item m.
- int **value** (int i)
Set the value of the menu to index i.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- FI_Align **align** () const
Gets the label alignment.
- void **align** (FI_Align alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class FI_GL_Window * **as_gl_window** ()
Returns an FI_GL_Window pointer if this widget is an FI_GL_Window.
- virtual class FI_GL_Window const * **as_gl_window** () const

- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color](#) [color2](#) () const

- For back compatibility only.*
- void [color2](#) (unsigned a)
- For back compatibility only.*
- int [contains](#) (const [FI_Widget](#) *w) const
- Checks if w is a child of this widget.*
- void [copy_label](#) (const char *new_label)
- Sets the current label.*
- void [copy_tooltip](#) (const char *text)
- Sets the current tooltip text.*
- [uchar](#) [damage](#) () const
- Returns non-zero if [draw\(\)](#) needs to be called.*
- void [damage](#) ([uchar](#) c)
- Sets the damage bits for the widget.*
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int [damage_resize](#) (int, int, int, int)
- Internal use only.*
- void [deactivate](#) ()
- Deactivates the widget.*
- [FI_Image](#) * [deimage](#) ()
- Gets the image that is used as part of the widget label when in the inactive state.*
- const [FI_Image](#) * [deimage](#) () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void [deimage](#) ([FI_Image](#) &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void [deimage](#) ([FI_Image](#) *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int [deimage_bound](#) () const
- Returns whether the inactive image is managed by the widget.*
- void [do_callback](#) ([FI_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- Calls the widget callback function with default arguments.*
- void [do_callback](#) ([FI_Widget](#) *widget, long arg, [FI_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- Calls the widget callback function with arbitrary arguments.*
- void [do_callback](#) ([FI_Widget](#) *widget, void *arg=0, [FI_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- Calls the widget callback function with arbitrary arguments.*
- virtual void [draw](#) ()=0
- Draws the widget.*
- void [draw_label](#) (int, int, int, int, [FI_Align](#)) const
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int [h](#) () const
- Gets the widget height.*
- virtual int [handle](#) (int event)
- Handles the specified event.*
- virtual void [hide](#) ()
- Makes a widget invisible.*
- int [horizontal_label_margin](#) ()
- Get the spacing between the label and the horizontal edge of the widget.*
- void [horizontal_label_margin](#) (int px)
- Set the spacing between the label and the horizontal edge of the widget.*
- [FI_Image](#) * [image](#) ()
- Gets the image that is used as part of the widget label when in the active state.*

- `const FI_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FI_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FI_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FI_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FI_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FI_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FI_Color c)`
Sets the label color.
- `FI_Font labelfont () const`
Gets the font to use.
- `void labelfont (FI_Font f)`
Sets the font to use.
- `FI_Fonsize labelsiz (int) const`
Gets the font size in pixels.
- `void labelsiz (FI_Fonsize pix)`
Sets the font size in pixels.
- `FI_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (FI_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`
Sets width ww and height hh accordingly with the label size.
- `bool needs_keyboard () const`
Returns whether this widget needs a keyboard.
- `void needs_keyboard (bool needs)`
Sets whether this widget needs a keyboard.
- `unsigned int output () const`
Returns if a widget is used for output only.
- `FI_Group * parent () const`
Returns a pointer to the parent widget.
- `void parent (FI_Group *p)`
Internal use only - "for hacks only".
- `void position (int X, int Y)`

- *Repositions the window or widget.*
- void `redraw` ()
- *Schedules the drawing of the widget.*
- void `redraw_label` ()
- *Schedules the drawing of the label.*
- virtual void `resize` (int `x`, int `y`, int `w`, int `h`)
- *Changes the size or position of the widget.*
- `FI_Color` `selection_color` () const
- *Gets the selection color.*
- void `selection_color` (`FI_Color` `a`)
- *Sets the selection color.*
- void `set_active` ()
- *Marks the widget as active without sending events or changing focus.*
- void `set_changed` ()
- *Marks the value of the widget as changed.*
- void `set_output` ()
- *Sets a widget to output only.*
- void `set_visible` ()
- *Makes the widget visible.*
- void `set_visible_focus` ()
- *Enables keyboard focus navigation with this widget.*
- int `shortcut_label` () const
- *Returns whether the widget's label uses '&' to indicate shortcuts.*
- void `shortcut_label` (int `value`)
- *Sets whether the widget's label uses '&' to indicate shortcuts.*
- virtual void `show` ()
- *Makes a widget visible.*
- void `size` (int `W`, int `H`)
- *Changes the size of the widget.*
- int `take_focus` ()
- *Gives the widget the keyboard focus.*
- unsigned int `takeevents` () const
- *Returns if the widget is able to take events.*
- int `test_shortcut` ()
- *Returns true if the widget's label contains the entered '&x' shortcut.*
- const char * `tooltip` () const
- *Gets the current tooltip text.*
- void `tooltip` (const char *`text`)
- *Sets the current tooltip text.*
- `FI_Window` * `top_window` () const
- *Returns a pointer to the top-level window for the widget.*
- `FI_Window` * `top_window_offset` (int &`xoff`, int &`yoff`) const
- *Finds the x/y offset of the current widget relative to the top-level window.*
- `uchar` `type` () const
- *Gets the widget type.*
- void `type` (`uchar` `t`)
- *Sets the widget type.*
- int `use_accents_menu` ()
- *Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*
- void * `user_data` () const
- *Gets the user data for this widget.*

- void **user_data** ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) (uchar i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual [~Fl_Widget](#) ()
Destroys the widget.

Protected Member Functions

- int **item_pathname_** (char *name, int namelen, const [Fl_Menu_Item](#) *finditem, const [Fl_Menu_Item](#) *menu=0) const

Protected Member Functions inherited from [Fl_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([Fl_Boxtype](#) t, [Fl_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([Fl_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.

- void `draw_focus` (`FI_Boxtype` t, int x, int y, int w, int h, `FI_Color` bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `FI_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Protected Attributes

- `uchar` `alloc`
- `uchar` `down_box_`
- `FI_Boxtype` `menu_box_`
- `FI_Color` `textcolor_`
- `FI_Font` `textfont_`
- `FI_Fonsize` `textsize_`

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,

`MAXIMIZED = 1<<24` , `POPUP = 1<<25` , `USERFLAG3 = 1<<29` , `USERFLAG2 = 1<<30` ,
`USERFLAG1 = 1<<31` }

flags possible values enumeration.

11.83.1 Detailed Description

Base class of all widgets that have a menu in FLTK.

Currently FLTK provides you with [Fl_Menu_Button](#), [Fl_Menu_Bar](#), and [Fl_Choice](#).

The class contains a pointer to an array of structures of type [Fl_Menu_Item](#). The array may either be supplied directly by the user program, or it may be "private": a dynamically allocated array managed by the [Fl_Menu_](#).

When the user clicks a menu item, [value\(\)](#) is set to that item and then:

- If the [Fl_Menu_Item](#) has a callback set, that callback is invoked with any userdata configured for it. (The [Fl_Menu_](#) widget's callback is NOT invoked.)
- For any [Fl_Menu_Items](#) that **don't** have a callback set, the [Fl_Menu_](#) widget's callback is invoked with any userdata configured for it. The callback can determine which item was picked using [value\(\)](#), [mvalue\(\)](#), [item_pathname\(\)](#), etc.

The line spacing between menu items can be controlled with the global setting [Fl::menu_linespacing\(\)](#).

See also

[Fl_Widget::shortcut_label\(int\)](#)

11.83.2 Constructor & Destructor Documentation

11.83.2.1 Fl_Menu_()

```
Fl_Menu_::Fl_Menu_ (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Menu_](#) widget using the given position, size, and label string.

[menu\(\)](#) is initialized to null.

11.83.3 Member Function Documentation

11.83.3.1 add() [1/2]

```
int Fl_Menu_::add (
    const char * str)
```

This is a Forms (and SGI GL library) compatible add function, it adds many menu items, with '|' separating the menu items, and tab separating the menu item names from an optional shortcut string.

The passed string is split at any '|' characters and then `add(s,0,0,0,0)` is done with each section. This is often useful if you are just using the value, and is compatible with Forms and other GL programs. The section strings use the same special characters as described for the long version of [add\(\)](#).

No items must be added to a menu during a callback to the same menu.

Parameters

<i>str</i>	string containing multiple menu labels as described above

Returns

the index into the [menu\(\)](#) array, where the entry was added

11.83.3.2 add() [2/2]

```
int Fl_Menu_::add (
    const char * label,
    int shortcut,
    Fl_Callback * callback,
    void * userdata = 0,
    int flags = 0)
```

Adds a new menu item.

Parameters

in	<i>label</i>	The text label for the menu item.
in	<i>shortcut</i>	Optional keyboard shortcut that can be an int or string: (FL_CTRL+'a') or "^a". Default 0 if none.
in	<i>callback</i>	Optional callback invoked when user clicks the item. Default 0 if none.
in	<i>userdata</i>	Optional user data passed as an argument to the callback. Default 0 if none.
in	<i>flags</i>	Optional flags that control the type of menu item; see below. Default is 0 for none.

Returns

The index into the [menu\(\)](#) array, where the entry was added.

Description

If the menu array was directly set with [menu\(x\)](#), then [copy\(\)](#) is done to make a private array.

Since this method can change the internal menu array, any menu item pointers or indices the application may have cached can become stale, and should be recalculated/refreshed.

A menu item's callback must not [add\(\)](#) items to its parent menu during the callback.

Due to backwards compatibility and historical restrictions we recommend to use either

- static menu arrays that are not extended during runtime or
- dynamic, extendable menu item arrays that are entirely created by using [add\(\)](#) or [insert\(\)](#).

This ensures that all menu arrays and strings are copied to internal storage and released when required.

Note

If you create menus from static [Fl_Menu_Item](#) arrays and [add\(\)](#) or [insert\(\)](#) more menu items later, then the menu array is copied to local storage but some local (static) strings may appear to "leak memory". This is a known issue and discouraged usage (see description above) but the impact on memory usage should typically be small.

Detailed Description of Parameters

label

The menu item's label. This argument is required and must not be NULL.

The characters "&", "/", "\", and "_" are treated as special characters in the label string. The "&" character specifies that the following character is an accelerator and will be underlined. The "\"" character is used to escape the next character in the string. Labels starting with the "_" character cause a divider to be placed after that menu item.

A label of the form "File/Quit" will create the submenu "File" with a menu item called "Quit".

The label string is copied to new memory and can be freed. The other arguments (including the shortcut) are copied into the menu item unchanged.

If an item exists already with that name then it is replaced with this new one. Otherwise this new one is added to the end of the correct menu or submenu. The return value is the offset into the array that the new entry was placed at.

shortcut

The keyboard shortcut for this menu item.

This parameter is optional, and defaults to 0 to indicate no shortcut.

The shortcut can either be a raw integer value (eg. FL_CTRL+'A') or a string (eg. "^c" or "^97").

Raw integer shortcuts can be a combination of keyboard chars (eg. 'A') and optional keyboard modifiers (see [Fl::event_state\(\)](#), e.g. FL_SHIFT, etc). In addition, FL_COMMAND can be used to denote FL_META under Mac OS X and FL_CTRL under other platforms.

String shortcuts can be specified in one of two ways:

```
[#+^]<ascii_value>    e.g. "97", "^97", "+97", "#97"
[#+^]<ascii_char>     e.g. "a", "^a", "+a", "#a"
```

..where <ascii_value> is a decimal value representing an ASCII character (eg. 97 is the ascii code for 'a'), and the optional prefixes enhance the value that follows. Multiple prefixes must appear in the order below.

```
# - Alt
+ - Shift
^ - Control
```

Internally, the text shortcuts are converted to integer values using `fl_old_shortcut(const char*)`.

callback

The callback to invoke when this menu item is selected.

This parameter is optional, and defaults to 0 for no callback.

userdata

The callback's 'user data' that is passed to the callback.

This parameter is optional, and defaults to 0.

flags

These are bit flags to define what kind of menu item this is.

This parameter is optional, and defaults to 0 to define a 'regular' menu item.

These flags can be 'OR'ed together:

```
FL_MENU_INACTIVE    // Deactivate menu item (gray out)
FL_MENU_TOGGLE      // Item is a checkbox toggle (shows checkbox for on/off state)
FL_MENU_VALUE       // The on/off state for checkbox/radio buttons (if set, state is 'on')
FL_MENU_RADIO       // Item is a radio button (one checkbox of many can be on)
FL_MENU_INVISIBLE   // Item will not show up (shortcut will work)
FL_SUBMENU_POINTER   // Indicates user_data() is a pointer to another menu array
FL_SUBMENU          // This item is a submenu to other items
FL_MENU_DIVIDER      // Creates divider line below this item. Also ends a group of radio buttons.
```

All other bits in 'flags' are reserved and must not be used.

If FL_SUBMENU is set in an item's flags, then actually two items are added:

- the first item is the menu item (submenu title), as expected, and
- the second item is the submenu terminating item with the label and all other members set to 0.

If you add submenus with the 'path' technique, then the corresponding submenu terminators (maybe more than one) are added as well.

Todo Raw integer shortcut needs examples. Dependent on responses to <https://www.fltk.org/newsgroups.php?gfltk.coredev+v:10086> and results of STR#2344

11.83.3.3 clear()

```
void Fl_Menu_::clear ()
```

Same as menu(NULL), set the array pointer to null, indicating a zero-length menu.

Menus must not be cleared during a callback to the same menu.

11.83.3.4 clear_submenu()

```
int Fl_Menu_::clear_submenu (
    int index)
```

Clears the specified submenu pointed to by `index` of all menu items.

This method is useful for clearing a submenu so that it can be re-populated with new items. Example: a "File/Recent Files/..." submenu that shows the last few files that have been opened.

The specified `index` must point to a submenu.

The submenu is cleared with `remove()`. If the menu array was directly set with `menu(x)`, then `copy()` is done to make a private array.

Warning

Since this method can change the internal menu array, any menu item pointers or indices the application may have cached can become stale, and should be recalculated/refreshed.

Example:

```
int index = menubar->find_index("File/Recent"); // get index of "File/Recent" submenu
if ( index != -1 ) menubar->clear_submenu(index); // clear the submenu
menubar->add("File/Recent/Aaa");
menubar->add("File/Recent/Bbb");
[...]
```

Parameters

<i>index</i>	The index of the submenu to be cleared

Returns

0 on success, -1 if the index is out of range or not a submenu

See also

[remove\(int\)](#)

11.83.3.5 copy()

```
void Fl_Menu_::copy (
    const Fl_Menu_Item * m,
    void * ud = 0)
```

Sets the menu array pointer with a copy of *m* that will be automatically deleted.

If userdata *ud* is not NULL, then all user data pointers are changed in the menus as well. See [void Fl_Menu_::menu\(const Fl_Menu_Item* m\)](#).

11.83.3.6 down_box()

```
void Fl_Menu_::down_box (
    Fl_Boxtype b) [inline]
```

Sets the box type used to surround the currently-selected items in the menus.

11.83.3.7 find_index() [1/3]

```
int Fl_Menu_::find_index (
    const char * pathname) const
```

Find the menu item index for a given menu *pathname*, such as "Edit/Copy".

This method finds a menu item's index position for the given menu *pathname*, also traversing submenus, but **not** submenu pointers (FL_SUBMENU_POINTER).

To get the menu item pointer for a *pathname*, use [find_item\(\)](#)

Parameters

<i>in</i>	<i>pathname</i>	The path and name of the menu item to find

Returns

The index of the matching item, or -1 if not found.

See also

[item_pathname\(\)](#)

11.83.3.8 find_index() [2/3]

```
int Fl_Menu_::find_index (
    const Fl_Menu_Item * item) const
```

Find the index into the menu array for a given *item*.

A way to convert a menu item pointer into an index.

Does **not** handle items that are in submenu pointers (FL_SUBMENU_POINTER).

-1 is returned if the item is not in this menu or is part of an FL_SUBMENU_POINTER submenu.
Current implementation is fast and not expensive.

```
// Convert an index-to-item
int index = 12;
const Fl_Menu_Item *item = mymenu->menu() + index;

// Convert an item-to-index
int index = mymenu->find_index(item);
if ( index == -1 ) { ..error.. }
```

Parameters

in	<i>item</i>	The item to be found

Returns

The index of the item, or -1 if not found.

See also

[menu\(\)](#)

11.83.3.9 find_index() [3/3]

```
int Fl_Menu_::find_index (
    Fl_Callback * cb) const
```

Find the index into the menu array for a given callback *cb*.

This method finds a menu item's index position, also traversing submenus, but **not** submenu pointers (FL_SUBMENU_POINTER). This is useful if an application uses internationalisation and a menu item can not be found using its label. This search is also much faster.

Parameters

<i>cb</i>	Find the first item with this callback

Returns

The index of the item with the specific callback, or -1 if not found

See also

[find_index\(const char*\)](#)

11.83.3.10 find_item() [1/2]

```
const Fl_Menu_Item * Fl_Menu_::find_item (
    const char * pathname)
```

Find the menu item for a given menu pathname, such as "Edit/Copy".

This method finds a menu item in the menu array, also traversing submenus, but not submenu pointers (FL_SUBMENU_POINTER).

To get the menu item's index, use [find_index\(const char*\)](#)

Example:

```
Fl_Menu_Bar *menubar = new Fl_Menu_Bar(..);
menubar->add("File/&Open");
menubar->add("File/&Save");
menubar->add("Edit/&Copy");
// [...]
Fl_Menu_Item *item;
if ( ( item = (Fl_Menu_Item*)menubar->find_item("File/&Open") ) != NULL ) {
    item->labelcolor(FL_RED);
}
if ( ( item = (Fl_Menu_Item*)menubar->find_item("Edit/&Copy") ) != NULL ) {
    item->labelcolor(FL_GREEN);
}
```

Parameters

<i>pathname</i>	The path and name of the menu item

Returns

The item found, or NULL if not found

See also

[find_index\(const char*\)](#), [find_item\(Fl_Callback*\)](#), [item_pathname\(\)](#)

11.83.3.11 find_item() [2/2]

```
const Fl_Menu_Item * Fl_Menu_::find_item (
    Fl_Callback * cb)
```

Find the menu item for the given callback *cb*.

This method finds a menu item in a menu array, also traversing submenus, but not submenu pointers. This is useful if an application uses internationalisation and a menu item can not be found using its label. This search is also much faster.

Parameters

<i>in</i>	<i>cb</i>	find the first item with this callback

Returns

The item found, or NULL if not found

See also

[find_item\(const char*\)](#)

11.83.3.12 find_item_with_argument()

```
const Fl_Menu_Item * Fl_Menu_::find_item_with_argument (
    long v)
```

Find the menu item for the given user argument *v*.

Parameters

<i>in</i>	<i>v</i>	find the first item with this user argument

Returns

The item found, or NULL if not found

See also

[find_item\(const char*\)](#)

11.83.3.13 find_item_with_user_data()

```
const Fl_Menu_Item * Fl_Menu_::find_item_with_user_data (
    void * v)
```

Find the menu item for the given user data *v*.

Parameters

in	v	find the first item with this user data

Returns

The item found, or NULL if not found

See also

[find_item\(const char*\)](#)

11.83.3.14 global()

```
void Fl_Menu_::global ()
```

Make the shortcuts for this menu work no matter what window has the focus when you type it.

This is done by using [Fl::add_handler\(\)](#). This [Fl_Menu_](#) widget does not have to be visible (ie the window it is in can be hidden, or it does not have to be put in a window at all).

Currently there can be only one [global\(\)](#) menu. Setting a new one will replace the old one. There is no way to remove the [global\(\)](#) setting (so don't destroy the widget!)

11.83.3.15 insert()

```
int Fl_Menu_::insert (
    int index,
    const char * label,
    int shortcut,
    Fl_Callback * callback,
    void * userdata = 0,
    int flags = 0)
```

Inserts a new menu item at the specified `index` position.

If `index` is -1, the menu item is appended; same behavior as [add\(\)](#).

To properly insert a menu item, `label` must be the name of the item (eg. "Quit"), and not a 'menu pathname' (eg. "File/Quit"). If a menu pathname is specified, the value of `index` is *ignored*, the new item's position defined by the pathname.

For more details, see [add\(\)](#). Except for the `index` parameter, [add\(\)](#) has more detailed information on parameters and behavior, and is functionally equivalent.

Parameters

in	<i>index</i>	The menu array's index position where the new item is inserted. If -1, behavior is the same as add() .
in	<i>label</i>	The text label for the menu item. If the label is a menu pathname, <code>index</code> is ignored, and the pathname indicates the position of the new item.
in	<i>shortcut</i>	Optional keyboard shortcut. Can be an int (FL_CTRL+'a') or a string ("^a"). Default is 0.
in	<i>callback</i>	Optional callback invoked when user clicks the item. Default 0 if none.
in	<i>userdata</i>	Optional user data passed as an argument to the callback. Default 0 if none.
in	<i>flags</i>	Optional flags that control the type of menu item; see add() for more info. Default is 0 for none.

Returns

The index into the [menu\(\)](#) array, where the entry was added.

See also

[add\(\)](#)

11.83.3.16 item_pathname()

```
int Fl_Menu_::item_pathname (
    char * name,
    int namelen,
    const Fl_Menu_Item * finditem = 0) const
```

Get the menu 'pathname' for the specified menuitem.

If finditem==NULL, [mvalue\(\)](#) is used (the most recently picked menuitem).

Example:

```
Fl_Menu_Bar *menubar = 0;
void my_menu_callback(Fl_Widget*, void*) {
    char name[80];
    if ( menubar->item_pathname(name, sizeof(name)-1) == 0 ) { // recently picked item
        if ( strcmp(name, "File/&Open") == 0 ) { .. } // open invoked
        if ( strcmp(name, "File/&Save") == 0 ) { .. } // save invoked
        if ( strcmp(name, "Edit/&Copy") == 0 ) { .. } // copy invoked
    }
}
int main() {
    [...]
    menubar = new Fl_Menu_Bar(..);
    menubar->add("File/&Open", 0, my_menu_callback);
    menubar->add("File/&Save", 0, my_menu_callback);
    menubar->add("Edit/&Copy", 0, my_menu_callback);
    [...]
}
```

Returns

- 0 : OK (name has menuitem's pathname)
- -1 : item not found (name=="")
- -2 : 'name' not large enough (name=="")

See also

[find_item\(\)](#)

11.83.3.17 menu() [1/2]

```
const Fl_Menu_Item * Fl_Menu_::menu () const [inline]
```

Returns a pointer to the array of `Fl_Menu_Items`.

This will either be the value passed to `menu(value)` or the private copy or an internal (temporary) location (see note below).

Note

Implementation details - may be changed in the future. All modifications of the menu array are done by copying the entire menu array to an internal storage for optimization of memory allocations, for instance when using `add()` or `insert()`. While this is done, `menu()` returns the pointer to this internal location. The entire menu will be copied back to private storage when needed, i.e. when **another** `Fl_Menu_` is modified. You can force this reallocation after you're done with all menu modifications by calling `Fl_Menu_::menu_end()` to make sure `menu()` returns a permanent pointer to private storage (until the menu is modified again). Note also that some menu methods (e.g. `Fl_Menu_Button::popup()`) call `menu_end()` internally to ensure a consistent menu array while the menu is open.

See also

`size()` – returns the [size](#) of the `Fl_Menu_Item` array.
`menu_end()` – finish menu modifications (optional)

Example: How to walk the array:

```
for ( int t=0; t<menubar->size(); t++ ) { // walk array of items
    const Fl_Menu_Item &item = menubar->menu()[t]; // get each item
    fprintf(stderr, "item #%d -- label=%s, value=%s type=%s\n",
        t,
        item.label() ? item.label() : "(Null)", // menu terminators have NULL labels
        (item.flags & FL_MENU_VALUE) ? "set" : "clear", // value of toggle or radio items
        (item.flags & FL_SUBMENU) ? "Submenu" : "Item"); // see if item is a submenu or actual item
}
```

11.83.3.18 menu() [2/2]

```
void Fl_Menu_::menu (
    const Fl_Menu_Item * m)
```

Sets the menu array pointer directly.

If the old menu is private it is deleted. NULL is allowed and acts the same as a zero-length menu. If you try to modify the array (with [add\(\)](#), [replace\(\)](#), or [remove\(\)](#)) a private copy is automatically done.

11.83.3.19 menu_box() [1/2]

```
Fl_Boxtype Fl_Menu_::menu_box () const [inline]
```

Get the box type for the menu popup windows.

Returns

the box type, or FL_NO_BOX if [Fl_Menu_::box\(\)](#) is to be used instead

11.83.3.20 menu_box() [2/2]

```
void Fl_Menu_::menu_box (
    Fl_Boxtype b) [inline]
```

Set the box type for the menu popup windows.

If menu_box set to FL_NO_BOX, the menu window will use [Fl_Menu_::box\(\)](#) instead.

Parameters

in	b	new box type or FL_NO_BOX

11.83.3.21 menu_end()

```
const Fl_Menu_Item * Fl_Menu_::menu_end ()
```

Finishes menu modifications and returns [menu\(\)](#).

Call [menu_end\(\)](#) after using [add\(\)](#), [insert\(\)](#), [remove\(\)](#), or any other methods that may change the menu array if you want to access the menu array anytime later with [menu\(\)](#). This should be called only once after the **last** menu modification for performance reasons.

Does nothing if the menu array is already in a private location.

Some methods like [Fl_Menu_Button::popup\(\)](#) call this method before their menu is opened.

Note

After menu changes like [add\(\)](#), [insert\(\)](#), etc. [menu\(\)](#) would return a pointer to a temporary internal menu array that may be relocated at unexpected times. This is due to performance considerations and may be changed w/o further notice.

Since

1.4.0

Returns

New [Fl_Menu_Item](#) array pointer.

See also

[Fl_Menu_::menu\(\)](#)

11.83.3.22 mode() [1/2]

```
int Fl_Menu_::mode (
    int i) const [inline]
```

Get the flags of item *i*.

For a list of the flags, see [Fl_Menu_Item](#).

11.83.3.23 mode() [2/2]

```
void Fl_Menu_::mode (
    int i,
    int fl) [inline]
```

Set the flags of item *i*.

For a list of the flags, see [Fl_Menu_Item](#).

11.83.3.24 mvalue()

```
const Fl_Menu_Item * Fl_Menu_::mvalue () const [inline]
```

Return a pointer to the last menu item that was picked.

11.83.3.25 picked()

```
const Fl_Menu_Item * Fl_Menu_::picked (
    const Fl_Menu_Item * v)
```

When user picks a menu item, call this.

It will do the callback.

Unfortunately this also casts away `const` for the checkboxes, but this was necessary so non-checkbox menus can really be declared 'const'.

Parameters

<i>in</i>	<i>v</i>	The menu item that was picked by the user.

Returns

The same `Fl_Menu_Item*` that was set (*v*).

11.83.3.26 prev_mvalue()

```
const Fl_Menu_Item * Fl_Menu_::prev_mvalue () const [inline]
```

Return a pointer to the menu item that was picked before the current one was picked.

This call gives developers additional details how a user changed a choice in the [Fl_Choice](#) widget.

11.83.3.27 remove()

```
void Fl_Menu_::remove (
    int i)
```

Deletes item *i* from the menu.

If the menu array was directly set with `menu(x)` then `copy()` is done to make a private array.

No items must be removed from a menu during a callback to the same menu.

Parameters

<i>i</i>	index into menu array

11.83.3.28 replace()

```
void Fl_Menu_::replace (
    int i,
    const char * str)
```

Changes the text of item *i*.

This is the only way to get slash into an [add\(\)](#)'ed menu item. If the menu array was directly set with [menu\(x\)](#) then [copy\(\)](#) is done to make a private array.

Parameters

<i>i</i>	index into menu array
<i>str</i>	new label for menu item at index <i>i</i>

11.83.3.29 size()

```
int Fl_Menu_::size () const
```

This returns the number of [Fl_Menu_Item](#) structures that make up the menu, correctly counting submenus.

This includes the "terminator" item at the end. To copy a menu array you need to copy [size\(\)](#)*sizeof([Fl_Menu_Item](#)) bytes. If the menu is NULL this returns zero (an empty menu will return 1).

11.83.3.30 test_shortcut()

```
const Fl_Menu_Item * Fl_Menu_::test_shortcut () [inline]
```

Returns the menu item with the entered shortcut (key value).

This searches the complete [menu\(\)](#) for a shortcut that matches the entered key value. It must be called for a [FL_KEYBOARD](#) or [FL_SHORTCUT](#) event.

If a match is found, the menu's callback will be called.

Returns

matched [Fl_Menu_Item](#) or NULL.

11.83.3.31 text() [1/2]

```
const char * Fl_Menu_::text () const [inline]
```

Returns the title of the last item chosen.

11.83.3.32 text() [2/2]

```
const char * Fl_Menu_::text (
    int i) const [inline]
```

Returns the title of item *i*.

11.83.3.33 textcolor()

```
Fl_Color Fl_Menu_::textcolor () const [inline]
```

Get the current color of menu item labels.

11.83.3.34 `textfont()` [1/2]

```
Fl_Font Fl_Menu_::textfont () const [inline]
```

Gets the current font of menu item labels.

11.83.3.35 `textfont()` [2/2]

```
void Fl_Menu_::textfont (
    Fl_Font c) [inline]
```

Sets the current font of menu item labels.

11.83.3.36 `textsize()` [1/2]

```
Fl_Fontsize Fl_Menu_::textsize () const [inline]
```

Gets the font size of menu item labels.

11.83.3.37 `textsize()` [2/2]

```
void Fl_Menu_::textsize (
    Fl_Fontsize c) [inline]
```

Sets the font size of menu item labels.

11.83.3.38 `value()` [1/3]

```
int Fl_Menu_::value () const
```

Return the index into the `menu()` of the last item chosen by the user.

The *value* of the menu is the index into the `menu()` of the last item chosen by the user or -1.

It is -1 initially (if no item has been chosen) or if the chosen menu item is part of a submenu addressed by an `FL_SUBMENU_POINTER`.

Note

All menu items are located in a contiguous array of `Fl_Menu_Item`'s unless an item has the `FL_SUBMENU_POINTER` flag which redirects the submenu to an independent submenu array. This submenu array is not counted in the `size()` of the menu, and menu items in this submenu can't return a valid index into the **main** menu. Therefore menu items that are located in such a submenu return -1 when `value()` is called. This may be changed in a future version.

You can use `mvalue()` instead to retrieve the last picked menu item directly.

Returns

Index of the last chosen menu item or -1 (see description).

See also

```
const Fl_Menu_Item *mvalue()
```

11.83.3.39 `value()` [2/3]

```
int Fl_Menu_::value (
    const Fl_Menu_Item * m)
```

Set the value of a menu to the menu item `m`.

The *value* of the menu is the index into the [menu\(\)](#) of the last item chosen by the user or -1.

It is -1 initially (if no item has been chosen) or if the chosen menu item is part of a submenu addressed by an FL_SUBMENU_POINTER.

Note

All menu items are located in a contiguous array of [Fl_Menu_Item](#)'s unless an item has the FL_SUBMENU_POINTER flag which redirects the submenu to an independent submenu array. This submenu array is not counted in the [size\(\)](#) of the menu, and menu items in this submenu can't return a valid index into the **main** menu. Therefore menu items that are located in such a submenu return -1 when [value\(\)](#) is called. This may be changed in a future version.

The menu item can be any menu item, even one in a detached submenu (see note about FL_SUBMENU_POINTER above).

Parameters

<i>in</i>	<i>m</i>	Pointer to any menu item.

Returns

Whether the new value is different than the old one.

Return values

0	The value didn't change.
1	The value was changed.

See also

[int value\(int\)](#)

[int value\(\)](#)

[const Fl_Menu_Item *mvalue\(\)](#)

11.83.3.40 value() [3/3]

```
int Fl_Menu_::value (
    int i) [inline]
```

Set the value of the menu to index *i*.

The *value* of the menu is the index into the [menu\(\)](#) of the last item chosen by the user or -1.

It is -1 initially (if no item has been chosen) or if the chosen menu item is part of a submenu addressed by an FL_SUBMENU_POINTER.

Note

All menu items are located in a contiguous array of [Fl_Menu_Item](#)'s unless an item has the FL_SUBMENU_POINTER flag which redirects the submenu to an independent submenu array. This submenu array is not counted in the [size\(\)](#) of the menu, and menu items in this submenu can't return a valid index into the **main** menu. Therefore menu items that are located in such a submenu return -1 when [value\(\)](#) is called. This may be changed in a future version.

You can set the value as an integer or with a pointer to a menu item. The integer value is restricted to the main menu array (0..[size\(\)](#)-1) whereas the menu item can be any menu item, even one in a detached submenu (see note about FL_SUBMENU_POINTER above).

Parameters

<i>in</i>	<i>i</i>	Index of the menu item in the main menu array. Values outside 0.. size() -1 are ignored (return 0).

Returns

Whether the new value is different than the old one.

Return values

0	The value didn't change.
1	The value was changed.

See also

int [value\(const FI_Menu_Item*\)](#)

int [value\(\)](#)

const [FI_Menu_Item](#) *mvalue()

The documentation for this class was generated from the following files:

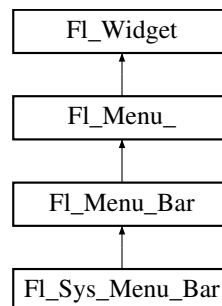
- FI_Menu_.H
- FI_Menu_.cxx
- FI_Menu_add.cxx
- FI_Menu_global.cxx

11.84 FI_Menu_Bar Class Reference

This widget provides a standard menubar interface.

```
#include <FI_Menu_Bar.H>
```

Inheritance diagram for FI_Menu_Bar:

**Public Member Functions**

- [FI_Menu_Bar](#) (int X, int Y, int W, int H, const char *|=0)
Creates a new [FI_Menu_Bar](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- virtual void [play_menu](#) (const [FI_Menu_Item](#) *item)
*Opens the 1st level submenu of the menubar corresponding to *item*.*
- virtual void [update](#) ()
Updates the menu bar after any change to its items.

Public Member Functions inherited from [FI_Menu_](#)

- int [add](#) (const char *)
This is a Forms (and SGI GL library) compatible add function, it adds many menu items, with '|' separating the menu items, and tab separating the menu item names from an optional shortcut string.

- `int add` (`const char *`, `int shortcut`, `Fl_Callback *`, `void *=0`, `int=0`)
Adds a new menu item.
- `int add` (`const char *a`, `const char *b`, `Fl_Callback *c`, `void *d=0`, `int e=0`)
See `int Fl_Menu_::add(const char* label, int shortcut, Fl_Callback*, void *user_data=0, int flags=0)`
- `void clear` ()
Same as `menu(NULL)`, set the array pointer to null, indicating a zero-length menu.
- `int clear_submenu` (`int index`)
Clears the specified submenu pointed to by `index` of all menu items.
- `void copy` (`const Fl_Menu_Item *m`, `void *user_data=0`)
Sets the menu array pointer with a copy of `m` that will be automatically deleted.
- `Fl_Boxtype down_box` () `const`
This box type is used to surround the currently-selected items in the menus.
- `void down_box` (`Fl_Boxtype b`)
Sets the box type used to surround the currently-selected items in the menus.
- `Fl_Color down_color` () `const`
For back compatibility, same as `selection_color()`
- `void down_color` (`unsigned c`)
For back compatibility, same as `selection_color()`
- `int find_index` (`const char *name`) `const`
Find the menu item index for a given menu `pathname`, such as "Edit/Copy".
- `int find_index` (`const Fl_Menu_Item *item`) `const`
Find the index into the menu array for a given `item`.
- `int find_index` (`Fl_Callback *cb`) `const`
Find the index into the menu array for a given callback `cb`.
- `const Fl_Menu_Item * find_item` (`const char *name`)
Find the menu item for a given menu `pathname`, such as "Edit/Copy".
- `const Fl_Menu_Item * find_item` (`Fl_Callback *`)
Find the menu item for the given callback `cb`.
- `const Fl_Menu_Item * find_item_with_argument` (`long`)
Find the menu item for the given user argument `v`.
- `const Fl_Menu_Item * find_item_with_user_data` (`void *`)
Find the menu item for the given user data `v`.
- `Fl_Menu_` (`int, int, int, int, const char *=0`)
Creates a new `Fl_Menu_` widget using the given position, size, and label string.
- `void global` ()
Make the shortcuts for this menu work no matter what window has the focus when you type it.
- `int insert` (`int index`, `const char *`, `int shortcut`, `Fl_Callback *`, `void *=0`, `int=0`)
Inserts a new menu item at the specified `index` position.
- `int insert` (`int index`, `const char *a`, `const char *b`, `Fl_Callback *c`, `void *d=0`, `int e=0`)
See `int Fl_Menu_::insert(const char* label, int shortcut, Fl_Callback*, void *user_data=0, int flags=0)`
- `int item_pathname` (`char *name`, `int namelen`, `const Fl_Menu_Item *finditem=0`) `const`
Get the menu 'pathname' for the specified menuitem.
- `const Fl_Menu_Item * menu` () `const`
Returns a pointer to the array of `Fl_Menu_Items`.
- `void menu` (`const Fl_Menu_Item *m`)
Sets the menu array pointer directly.
- `Fl_Boxtype menu_box` () `const`
Get the box type for the menu popup windows.
- `void menu_box` (`Fl_Boxtype b`)
Set the box type for the menu popup windows.
- `const Fl_Menu_Item * menu_end` ()

- Finishes menu modifications and returns [menu\(\)](#).*

 - int [mode](#) (int i) const

Get the flags of item i.
- void [mode](#) (int i, int fl)

Set the flags of item i.
- const [Fl_Menu_Item](#) * [mvalue](#) () const

Return a pointer to the last menu item that was picked.
- const [Fl_Menu_Item](#) * [picked](#) (const [Fl_Menu_Item](#) *)

When user picks a menu item, call this.
- const [Fl_Menu_Item](#) * [prev_mvalue](#) () const

Return a pointer to the menu item that was picked before the current one was picked.
- void [remove](#) (int)

Deletes item i from the menu.
- void [replace](#) (int, const char *)

Changes the text of item i.
- void [setonly](#) ([Fl_Menu_Item](#) *item)

Turns the radio item "on" for the menu item and turns "off" adjacent radio items of the same group.
- void [shortcut](#) (int i, int s)

Change the shortcut of item i to s.
- int [size](#) () const

This returns the number of [Fl_Menu_Item](#) structures that make up the menu, correctly counting submenus.
- void [size](#) (int W, int H)
- const [Fl_Menu_Item](#) * [test_shortcut](#) ()

Returns the menu item with the entered shortcut (key value).
- const char * [text](#) () const

Returns the title of the last item chosen.
- const char * [text](#) (int i) const

Returns the title of item i.
- [Fl_Color](#) [textcolor](#) () const

Get the current color of menu item labels.
- void [textcolor](#) ([Fl_Color](#) c)

Sets the current color of menu item labels.
- [Fl_Font](#) [textfont](#) () const

Gets the current font of menu item labels.
- void [textfont](#) ([Fl_Font](#) c)

Sets the current font of menu item labels.
- [Fl_Fontsize](#) [textsize](#) () const

Gets the font size of menu item labels.
- void [textsize](#) ([Fl_Fontsize](#) c)

Sets the font size of menu item labels.
- int [value](#) () const

Return the index into the [menu\(\)](#) of the last item chosen by the user.
- int [value](#) (const [Fl_Menu_Item](#) *)

Set the value of a menu to the menu item m.
- int [value](#) (int i)

Set the value of the menu to index i.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (**FI_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FI_Gl_Window** * **as_gl_window** ()
Returns an FI_Gl_Window pointer if this widget is an FI_Gl_Window.
- virtual class **FI_Gl_Window** const * **as_gl_window** () const
- virtual **FI_Group** * **as_group** ()
Returns an FI_Group pointer if this widget is an FI_Group.
- virtual **FI_Group** const * **as_group** () const
- virtual **FI_Window** * **as_window** ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual **FI_Window** const * **as_window** () const
- void **bind_deimage** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- **FI_Boxtype box** () const
Gets the box type of the widget.
- void **box** (**FI_Boxtype** new_box)
Sets the box type for the widget.
- **FI_Callback_p callback** () const
Gets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb, **FI_Callback_User_Data** *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void **callback** (**FI_Callback** *cb, void *p)
Sets the current callback function and data for the widget.
- void **callback** (**FI_Callback0** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback1** *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int **changed** () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FL_Color` bg)

Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)

Sets the background and selection color of the widget.
- `FL_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.

- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [Fl_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([Fl_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [Fl_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from [FI_Menu_](#)

- int **item_pathname_** (char *name, int namelen, const [FI_Menu_Item](#) *finditem, const [FI_Menu_Item](#) *menu=0) const

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Friends

- class [FI_Sys_Menu_Bar_Driver](#)

Additional Inherited Members

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }

flags possible values enumeration.

Protected Attributes inherited from FI_Menu_

- `uchar alloc`
- `uchar down_box_`
- `FI_Boxtype menu_box_`
- `FI_Color textcolor_`
- `FI_Font textfont_`
- `FI_Fonsize textsize_`

11.84.1 Detailed Description

This widget provides a standard menubar interface.

Usually you will put this widget along the top edge of your window. The height of the widget should be 30 for the menu titles to draw correctly with the default font.

The items on the bar and the menus they bring up are defined by a single `FI_Menu_Item` array. Because a `FI_Menu_Item` array defines a hierarchy, the top level menu defines the items in the menubar, while the submenus define the pull-down menus. Sub-sub menus and lower pop up to the right of the submenus.

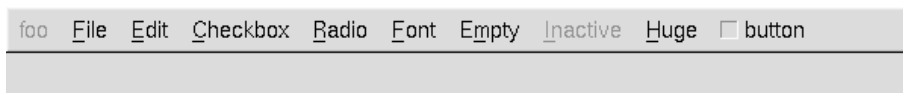


Figure 11.28 menubar

If there is an item in the top menu that is not a title of a submenu, then it acts like a "button" in the menubar. Clicking on it will pick it.

When the user clicks a menu item, `value()` is set to that item and then:

- The item's callback is done if one has been set; the `FI_Menu_Bar` is passed as the `FI_Widget*` argument, along with any userdata configured for the callback.
- If the item does not have a callback, the `FI_Menu_Bar`'s callback is done instead, along with any userdata configured for the callback. The callback can determine which item was picked using `value()`, `mvalue()`, `item_pathname()`, etc.

Submenus will also pop up in response to shortcuts indicated by putting a '&' character in the name field of the menu item. If you put a '&' character in a top-level "button" then the shortcut picks it. The '&' character in submenus is ignored until the menu is popped up.

Typing the `shortcut()` of any of the menu items will cause callbacks exactly the same as when you pick the item with the mouse.

11.84.2 Constructor & Destructor Documentation

11.84.2.1 Fl_Menu_Bar()

```
Fl_Menu_Bar::Fl_Menu_Bar (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Menu_Bar](#) widget using the given position, size, and label string.

The default boxtype is `FL_UP_BOX`.

The constructor sets [menu\(\)](#) to NULL. See [Fl_Menu_](#) for the methods to set or change the menu.

[labelsize\(\)](#), [labelfont\(\)](#), and [labelcolor\(\)](#) are used to control how the menubar items are drawn. They are initialized from the `Fl_Menu` static variables, but you can change them if desired.

[label\(\)](#) is ignored unless you change [align\(\)](#) to put it outside the menubar.

The destructor removes the [Fl_Menu_Bar](#) widget and all of its menu items.

11.84.3 Member Function Documentation

11.84.3.1 draw()

```
void Fl_Menu_Bar::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                  // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

Reimplemented in [Fl_Sys_Menu_Bar](#).

11.84.3.2 handle()

```
int Fl_Menu_Bar::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

11.84.3.3 play_menu()

```
void Fl_Menu_Bar::play_menu (
    const Fl\_Menu\_Item * item) [virtual]
```

Opens the 1st level submenu of the menubar corresponding to *item*.

Since

1.4.0

Reimplemented in [Fl_Sys_Menu_Bar](#).

11.84.3.4 update()

```
virtual void Fl_Menu_Bar::update () [inline], [virtual]
```

Updates the menu bar after any change to its items.

This is useful when the menu bar can be an [Fl_Sys_Menu_Bar](#) object.

Reimplemented in [Fl_Sys_Menu_Bar](#).

The documentation for this class was generated from the following files:

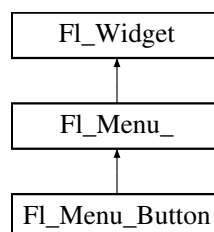
- [Fl_Menu_Bar.H](#)
- [Fl_Menu_Bar.cxx](#)

11.85 Fl_Menu_Button Class Reference

This is a button that when pushed pops up a menu (or hierarchy of menus) defined by an array of [Fl_Menu_Item](#) objects.

```
#include <Fl_Menu_Button.H>
```

Inheritance diagram for [Fl_Menu_Button](#):



Public Types

- enum [popup_buttons](#) {
[POPUP1](#) = 1 , [POPUP2](#) , [POPUP12](#) , [POPUP3](#) ,
[POPUP13](#) , [POPUP23](#) , [POPUP123](#) }

indicate what mouse buttons pop up the menu.

Public Member Functions

- [Fl_Menu_Button](#) (int, int, int, int, const char * = 0)
Creates a new [Fl_Menu_Button](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- const [Fl_Menu_Item](#) * [popup](#) ()
Act exactly as though the user clicked the button or typed the shortcut key.

Public Member Functions inherited from [Fl_Menu_](#)

- int [add](#) (const char *)
This is a Forms (and SGI GL library) compatible add function, it adds many menu items, with '|' separating the menu items, and tab separating the menu item names from an optional shortcut string.
- int [add](#) (const char *, int [shortcut](#), [Fl_Callback](#) *, void *=0, int=0)
Adds a new menu item.
- int [add](#) (const char *a, const char *b, [Fl_Callback](#) *c, void *d=0, int e=0)
See int [Fl_Menu_::add](#)(const char label, int shortcut, [Fl_Callback](#)*, void *user_data=0, int flags=0)*
- void [clear](#) ()
Same as menu(NULL), set the array pointer to null, indicating a zero-length menu.
- int [clear_submenu](#) (int index)
Clears the specified submenu pointed to by index of all menu items.
- void [copy](#) (const [Fl_Menu_Item](#) *m, void *user_data=0)
Sets the menu array pointer with a copy of m that will be automatically deleted.
- [Fl_Boxtype](#) [down_box](#) () const
This box type is used to surround the currently-selected items in the menus.
- void [down_box](#) ([Fl_Boxtype](#) b)
Sets the box type used to surround the currently-selected items in the menus.
- [Fl_Color](#) [down_color](#) () const
For back compatibility, same as [selection_color](#)()
- void [down_color](#) (unsigned c)
For back compatibility, same as [selection_color](#)()
- int [find_index](#) (const char *name) const
Find the menu item index for a given menu pathname, such as "Edit/Copy".
- int [find_index](#) (const [Fl_Menu_Item](#) *item) const
Find the index into the menu array for a given item.
- int [find_index](#) ([Fl_Callback](#) *cb) const
Find the index into the menu array for a given callback cb.
- const [Fl_Menu_Item](#) * [find_item](#) (const char *name)
Find the menu item for a given menu pathname, such as "Edit/Copy".
- const [Fl_Menu_Item](#) * [find_item](#) ([Fl_Callback](#) *)
Find the menu item for the given callback cb.
- const [Fl_Menu_Item](#) * [find_item_with_argument](#) (long)
Find the menu item for the given user argument v.
- const [Fl_Menu_Item](#) * [find_item_with_user_data](#) (void *)
Find the menu item for the given user data v.
- [Fl_Menu_](#) (int, int, int, int, const char *=0)
Creates a new [Fl_Menu_](#) widget using the given position, size, and label string.
- void [global](#) ()
Make the shortcuts for this menu work no matter what window has the focus when you type it.
- int [insert](#) (int index, const char *, int [shortcut](#), [Fl_Callback](#) *, void *=0, int=0)
Inserts a new menu item at the specified index position.
- int [insert](#) (int index, const char *a, const char *b, [Fl_Callback](#) *c, void *d=0, int e=0)
See int [Fl_Menu_::insert](#)(const char label, int shortcut, [Fl_Callback](#)*, void *user_data=0, int flags=0)*
- int [item_pathname](#) (char *name, int namelen, const [Fl_Menu_Item](#) *finditem=0) const
Get the menu 'pathname' for the specified menuitem.
- const [Fl_Menu_Item](#) * [menu](#) () const
Returns a pointer to the array of [Fl_Menu_Items](#).
- void [menu](#) (const [Fl_Menu_Item](#) *m)
Sets the menu array pointer directly.

- [FI_Boxtype menu_box](#) () const
Get the box type for the menu popup windows.
- void [menu_box](#) ([FI_Boxtype](#) b)
Set the box type for the menu popup windows.
- const [FI_Menu_Item](#) * [menu_end](#) ()
Finishes menu modifications and returns [menu\(\)](#).
- int [mode](#) (int i) const
Get the flags of item i.
- void [mode](#) (int i, int fl)
Set the flags of item i.
- const [FI_Menu_Item](#) * [mvalue](#) () const
Return a pointer to the last menu item that was picked.
- const [FI_Menu_Item](#) * [picked](#) (const [FI_Menu_Item](#) *)
When user picks a menu item, call this.
- const [FI_Menu_Item](#) * [prev_mvalue](#) () const
Return a pointer to the menu item that was picked before the current one was picked.
- void [remove](#) (int)
Deletes item i from the menu.
- void [replace](#) (int, const char *)
Changes the text of item i.
- void [setonly](#) ([FI_Menu_Item](#) *item)
Turns the radio item "on" for the menu item and turns "off" adjacent radio items of the same group.
- void [shortcut](#) (int i, int s)
Change the shortcut of item i to s.
- int [size](#) () const
This returns the number of [FI_Menu_Item](#) structures that make up the menu, correctly counting submenus.
- void [size](#) (int W, int H)
- const [FI_Menu_Item](#) * [test_shortcut](#) ()
Returns the menu item with the entered shortcut (key value).
- const char * [text](#) () const
Returns the title of the last item chosen.
- const char * [text](#) (int i) const
Returns the title of item i.
- [FI_Color](#) [textcolor](#) () const
Get the current color of menu item labels.
- void [textcolor](#) ([FI_Color](#) c)
Sets the current color of menu item labels.
- [FI_Font](#) [textfont](#) () const
Gets the current font of menu item labels.
- void [textfont](#) ([FI_Font](#) c)
Sets the current font of menu item labels.
- [FI_Fontsize](#) [textsize](#) () const
Gets the font size of menu item labels.
- void [textsize](#) ([FI_Fontsize](#) c)
Sets the font size of menu item labels.
- int [value](#) () const
Return the index into the [menu\(\)](#) of the last item chosen by the user.
- int [value](#) (const [FI_Menu_Item](#) *)
Set the value of a menu to the menu item m.
- int [value](#) (int i)
Set the value of the menu to index i.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*
- void `clear_damage` (uchar c=0)
- Clears or sets the damage flags.*
- void `clear_output` ()
- Sets a widget to accept input.*
- void `clear_visible` ()
- Hides the widget.*
- void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*
- `FL_Color color` () const
- Gets the background color of the widget.*
- void `color` (FL_Color bg)
- Sets the background color of the widget.*
- void `color` (FL_Color bg, FL_Color sel)
- Sets the background and selection color of the widget.*
- `FL_Color color2` () const
- For back compatibility only.*
- void `color2` (unsigned a)
- For back compatibility only.*
- int `contains` (const FL_Widget *w) const
- Checks if w is a child of this widget.*
- void `copy_label` (const char *new_label)
- Sets the current label.*
- void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*
- `uchar damage` () const
- Returns non-zero if `draw()` needs to be called.*
- void `damage` (uchar c)
- Sets the damage bits for the widget.*
- void `damage` (uchar c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int `damage_resize` (int, int, int, int)
- Internal use only.*
- void `deactivate` ()
- Deactivates the widget.*
- `FL_Image * deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*
- const `FL_Image * deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void `deimage` (FL_Image &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void `deimage` (FL_Image *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)
- Calls the widget callback function with default arguments.*

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([FL_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from FI_Menu_

- int **item_pathname_** (char *name, int namelen, const FI_Menu_Item *finditem, const FI_Menu_Item *menu=0) const

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Static Protected Attributes

- static FI_Menu_Button * **pressed_menu_button_** = NULL

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
 ,
[OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
 ,
[MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
 ,
[GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
 = 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

Protected Attributes inherited from [Fl_Menu_](#)

- [uchar](#) `alloc`
- [uchar](#) `down_box_`
- [Fl_Boxtype](#) `menu_box_`
- [Fl_Color](#) `textcolor_`
- [Fl_Font](#) `textfont_`
- [Fl_Fonsize](#) `textsize_`

11.85.1 Detailed Description

This is a button that when pushed pops up a menu (or hierarchy of menus) defined by an array of [Fl_Menu_Item](#) objects.



Figure 11.29 `menu_button`

Normally any mouse button will pop up a menu and it is lined up below the button as shown in the picture. However an [Fl_Menu_Button](#) may also control a pop-up menu. This is done by setting the `type()`. If `type()` is zero a normal menu button is produced. If it is nonzero then this is a pop-up menu. The bits in `type()` indicate what mouse buttons pop up the menu (see [Fl_Menu_Button::popup_buttons](#)).

The menu will also pop up in response to shortcuts indicated by putting a '&' character in the `label()`.

Typing the `shortcut()` of any of the menu items will cause callbacks exactly the same as when you pick the item with the mouse. The '&' character in menu item names are only looked at when the menu is popped up, however.

When the user clicks a menu item, `value()` is set to that item and then:

- The item's callback is done if one has been set; the `Fl_Menu_Button` is passed as the `Fl_Widget*` argument, along with any userdata configured for the callback.
- If the item does not have a callback, the `Fl_Menu_Button`'s callback is done instead, along with any userdata configured for it. The callback can determine which item was picked using `value()`, `mvalue()`, `item_pathname()`, etc.

11.85.2 Member Enumeration Documentation

11.85.2.1 popup_buttons

enum `Fl_Menu_Button::popup_buttons`

indicate what mouse buttons pop up the menu.

Values for `type()` used to indicate what mouse buttons pop up the menu. `Fl_Menu_Button::POPUP3` is usually what you want.

Enumerator

POPUP1	pops up with the mouse 1st button.
POPUP2	pops up with the mouse 2nd button.
POPUP12	pops up with the mouse 1st or 2nd buttons.
POPUP3	pops up with the mouse 3rd button.
POPUP13	pops up with the mouse 1st or 3rd buttons.
POPUP23	pops up with the mouse 2nd or 3rd buttons.
POPUP123	pops up with any mouse button.

11.85.3 Constructor & Destructor Documentation

11.85.3.1 Fl_Menu_Button()

```
Fl_Menu_Button::Fl_Menu_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new `Fl_Menu_Button` widget using the given position, size, and label string.

The default boxtype is `FL_UP_BOX`.

The constructor sets `menu()` to `NULL`. See `Fl_Menu_` for the methods to set or change the menu.

11.85.4 Member Function Documentation

11.85.4.1 draw()

```
void Fl_Menu_Button::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call `redraw()` instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own `draw()` method*, e.g. for an embedded scrollbar, you can do it (because `draw()` is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements `Fl_Widget`.

11.85.4.2 `handle()`

```
int Fl_Menu_Button::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

11.85.4.3 `popup()`

```
const Fl_Menu_Item * Fl_Menu_Button::popup ()
```

Act exactly as though the user clicked the button or typed the shortcut key.

The menu appears, it waits for the user to pick an item, and if they pick one it sets [value\(\)](#) and does the callback or sets [changed\(\)](#) as described above. The menu item is returned or NULL if the user dismisses the menu.

Note

Since FLTK 1.4.0 [Fl_Menu_::menu_end\(\)](#) is called before the menu pops up to make sure the menu array is located in private storage.

See also

[Fl_Menu_::menu_end\(\)](#)

The documentation for this class was generated from the following files:

- `Fl_Menu_Button.H`
- `Fl_Menu_Button.cxx`

11.86 `Fl_Menu_Item` Struct Reference

The [Fl_Menu_Item](#) structure defines a single menu item that is used by the [Fl_Menu_](#) class.

```
#include <Fl_Menu_Item.H>
```


Public Member Functions

- void **activate** ()
Allows a menu item to be picked.
- int **active** () const
Gets whether or not the item can be picked.
- int **activevisible** () const
Returns non 0 if FL_INACTIVE and FL_INVISIBLE are cleared, 0 otherwise.
- int **add** (const char *, int shortcut, FL_Callback *, void *=0, int=0)
Adds a menu item.
- int **add** (const char *a, const char *b, FL_Callback *c, void *d=0, int e=0)
See int add(const char, int shortcut, FL_Callback*, void*, int)*
- long **argument** () const
Gets the user_data() argument that is sent to the callback function.
- void **argument** (long v)
Sets the user_data() argument that is sent to the callback function.
- FL_Callback_p **callback** () const
Returns the callback function that is set for the menu item.
- void **callback** (FL_Callback *c)
Sets the menu item's callback function.
- void **callback** (FL_Callback *c, void *p)
Sets the menu item's callback function and userdata() argument.
- void **callback** (FL_Callback0 *c)
Sets the menu item's callback function.
- void **callback** (FL_Callback1 *c, long p=0)
Sets the menu item's callback function and userdata() argument.
- void **check** ()
Back compatibility only.
- int **checkbox** () const
Returns true if a checkbox will be drawn next to this item.
- int **checked** () const
Back compatibility only.
- void **clear** ()
Turns the check or radio item "off" for the menu item.
- void **deactivate** ()
Prevents a menu item from being picked.
- void **do_callback** (FL_Widget *o) const
Calls the FL_Menu_Item item's callback, and provides the FL_Widget argument.
- void **do_callback** (FL_Widget *o, long arg) const
Calls the FL_Menu_Item item's callback, and provides the FL_Widget argument.
- void **do_callback** (FL_Widget *o, void *arg) const
Calls the FL_Menu_Item item's callback, and provides the FL_Widget argument.
- void **draw** (int x, int y, int w, int h, const FL_Menu_ *, int t=0) const
Draws the menu item in bounding box x,y,w,h, optionally selects the item.
- const FL_Menu_Item * **find_shortcut** (int *ip=0, const bool require_alt=false) const
Search only the top level menu for a shortcut.
- FL_Menu_Item * **first** ()
Returns the first menu item, same as next(0).
- const FL_Menu_Item * **first** () const
Returns the first menu item, same as next(0).
- void **hide** ()

- Hides an item in the menu.*

 - void **image** (**FI_Image** &image)

Compatibility API for FLUID, same as image.label(this).
- void **image** (**FI_Image** *image)

Compatibility API for FLUID, same as image->label(this).
- void **image_label** (const **FI_Image** *image)

Sets the title (label()) to an icon or image.
- int **insert** (int, const char *, int, **FI_Callback** *, void *=0, int=0)

Inserts an item at position index.
- const char * **label** () const

Returns the title (label) of the menu item.
- void **label** (const char *a)

Sets the title (label) of the menu item.
- void **label** (**FI_Labeltype** a, const char *b)

Sets the title (label) and the label type of the menu item.
- **FI_Color** **labelcolor** () const

Gets the menu item's label color.
- void **labelcolor** (**FI_Color** a)

Sets the menu item's label color.
- **FI_Font** **labelfont** () const

Gets the menu item's label font.
- void **labelfont** (**FI_Font** a)

Sets the menu item's label font.
- **FI_Fonsize** **labelsize** () const

Gets the label font pixel size/height.
- void **labelsize** (**FI_Fonsize** a)

Sets the label font pixel size/height.
- **FI_Labeltype** **labeltype** () const

Returns the menu item's labeltype.
- void **labeltype** (**FI_Labeltype** a)

Sets the menu item's labeltype.
- int **measure** (int *h, const **FI_Menu_** *) const

Measures width of label, including effect of & characters.
- void **multi_label** (const **FI_Multi_Label** *ml)

Sets the title (label()) and labeltype() to an FI_Multi_Label.
- **FI_Menu_Item** * **next** (int i=1)

Advances a pointer by n items through a menu array, skipping the contents of submenus and invisible items.
- const **FI_Menu_Item** * **next** (int=1) const

Advance a pointer by n items through a menu array, skipping the contents of submenus and invisible items.
- const **FI_Menu_Item** * **popup** (int X, int Y, const char *title=0, const **FI_Menu_Item** *picked=0, const **FI_Menu_** *=0) const

This method is called by widgets that want to display menus.
- const **FI_Menu_Item** * **pulldown** (int X, int Y, int W, int H, const **FI_Menu_Item** *picked=0, const **FI_Menu_** *=0, const **FI_Menu_Item** *title=0, int menubar=0) const

Pulldown() is similar to popup(), but a rectangle is provided to position the menu.
- int **radio** () const

Returns true if this item is a radio item.
- void **set** ()

Turns the check or radio item "on" for the menu item.
- void **setonly** (**FI_Menu_Item** const *first=NULL)

Turns the radio item "on" for the menu item and turns "off" adjacent radio items set.

- int **shortcut** () const
Gets what key combination shortcut will trigger the menu item.
- void **shortcut** (int s)
Sets exactly what key combination will trigger the menu item.
- void **show** ()
Makes an item visible in the menu.
- int **size** () const
Size of the menu starting from this menu item.
- int **submenu** () const
Returns true if either FL_SUBMENU or FL_SUBMENU_POINTER is on in the flags.
- const FL_Menu_Item * **test_shortcut** () const
This is designed to be called by a widgets handle() method in response to a FL_SHORTCUT event.
- void **uncheck** ()
Back compatibility only.
- void * **user_data** () const
Gets the [user_data\(\)](#) argument that is sent to the callback function.
- void **user_data** (void *v)
Sets the [user_data\(\)](#) argument that is sent to the callback function.
- int **value** () const
Returns the current value of the check or radio item.
- void **value** (int v)
Sets the current value of the check or radio item.
- int **visible** () const
Gets the visibility of an item.

Public Attributes

- [FL_Callback](#) * **callback_**
menu item callback
- int **flags**
menu item flags like FL_MENU_TOGGLE, FL_MENU_RADIO
- [FL_Color](#) **labelcolor_**
menu item text color
- [FL_Font](#) **labelfont_**
which font for this menu item text
- [FL_Fonsize](#) **labelsize_**
size of menu item text
- [uchar](#) **labeltype_**
how the menu item text looks like
- int **shortcut_**
menu item shortcut
- const char * **text**
menu item text, returned by [label\(\)](#)
- void * **user_data_**
menu item user_data for the menu's callback

11.86.1 Detailed Description

The `Fl_Menu_Item` structure defines a single menu item that is used by the `Fl_Menu_` class.

```
struct Fl_Menu_Item {
    const char*  text;          // label()
    int          shortcut_;
    Fl_Callback* callback_;
    void*        user_data_;
    int          flags;
    uchar        labeltype_;
    uchar        labelfont_;
    uchar        labelsize_;
    uchar        labelcolor_;
};

enum { // values for flags:
    FL_MENU_INACTIVE   = 1,      // Deactivate menu item (gray out)
    FL_MENU_TOGGLE     = 2,      // Item is a checkbox toggle (shows checkbox for on/off state)
    FL_MENU_VALUE      = 4,      // The on/off state for checkbox/radio buttons (if set, state is 'on')
    FL_MENU_RADIO      = 8,      // Item is a radio button (one checkbox of many can be on)
    FL_MENU_INVISIBLE  = 0x10,   // Item will not show up (shortcut will work)
    FL_SUBMENU_POINTER = 0x20,   // Indicates user_data() is a pointer to another menu array
    FL_SUBMENU         = 0x40,   // This item is a submenu to other items
    FL_MENU_DIVIDER    = 0x80,   // Creates divider line below this item. Also ends a group of radio
                                buttons.
    FL_MENU_HORIZONTAL = 0x100   // ??? -- reserved, internal (do not use)
};
```

Note

All other bits in `flags` are reserved for FLTK usage, do not use any bits of the `flags` variable for your own purposes. Even **undocumented** bits can be used for internal purposes in this or any future FLTK version.

Some `flags` bits may be changed during runtime by user code, particularly if you need to change the value of a menu item (ON/OFF) or make it active or inactive. Such changes must be done with caution so they don't affect other (maybe undocumented) bits, i.e. you need to make proper bit operations to set or clear only these particular bits.

Typically menu items are statically defined; for example:

```
Fl_Menu_Item popup[] = {
    {"&alpha",    FL_ALT+'a', the_cb, (void*)1},
    {"&beta",     FL_ALT+'b', the_cb, (void*)2},
    {"&gamma",    FL_ALT+'c', the_cb, (void*)3, FL_MENU_DIVIDER},
    {"&strange",  0,          strange_cb},
    {"&charm",    0,          charm_cb},
    {"&truth",    0,          truth_cb},
    {"&beauty",   0,          beauty_cb},
    {"sub&menu",  0,          0, 0, FL_SUBMENU},
    {"one"},
    {"two"},
    {"three"},
    {0},
    {"inactive",  FL_ALT+'i', 0, 0, FL_MENU_INACTIVE|FL_MENU_DIVIDER},
    {"invisible", FL_ALT+'i', 0, 0, FL_MENU_INVISIBLE},
    {"check",     FL_ALT+'i', 0, 0, FL_MENU_TOGGLE|FL_MENU_VALUE},
    {"box",       FL_ALT+'i', 0, 0, FL_MENU_TOGGLE},
    {0}};
```

produces:

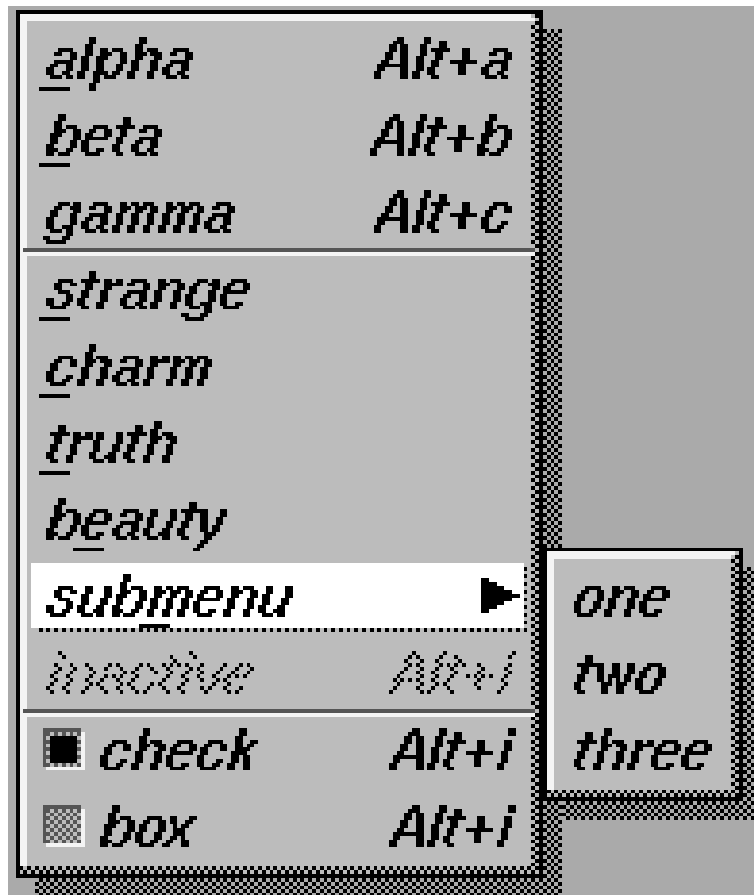


Figure 11.30 menu

A submenu title is identified by the bit `FL_SUBMENU` in the flags field, and ends with a `label()` that is `NULL`. You can nest menus to any depth. A pointer to the first item in the submenu can be treated as an `Fl_Menu` array itself. It is also possible to make separate submenu arrays with `FL_SUBMENU_POINTER` flags. You should use the method functions to access structure members and not access them directly to avoid compatibility problems with future releases of FLTK.

Note

Adding menu items with `insert()`, `add()`, or any of its overloaded variants copies the entire menu to internal storage. Using the memory of a static menu array after that would access unused (but not released) memory and thus have no effect.

11.86.2 Member Function Documentation

11.86.2.1 `add()`

```
int Fl_Menu_Item::add (
    const char * mytext,
    int sc,
    Fl_Callback * cb,
    void * data = 0,
    int myflags = 0)
```

Adds a menu item.

The text is split at `'` characters to automatically produce submenus (actually a totally unnecessary feature as you can now add submenu titles directly by setting `FL_SUBMENU` in the flags).

Returns

the index into the `menu()` array, where the entry was added

See also

[Fl_Menu_Item::insert\(int, const char*, int, Fl_Callback*, void*, int\)](#)

11.86.2.2 argument() [1/2]

```
long Fl_Menu_Item::argument () const [inline]
```

Gets the [user_data\(\)](#) argument that is sent to the callback function.

For convenience you can also define the callback as taking a long argument. This method casts the stored [user_data\(\)](#) argument to long and returns it as a *long* value.

11.86.2.3 argument() [2/2]

```
void Fl_Menu_Item::argument (
    long v) [inline]
```

Sets the [user_data\(\)](#) argument that is sent to the callback function.

For convenience you can also define the callback as taking a long argument. This method casts the given argument *v* to *void** and stores it in the menu item's [userdata\(\)](#) member. This may not be portable to some machines.

11.86.2.4 callback() [1/5]

```
Fl_Callback_p Fl_Menu_Item::callback () const [inline]
```

Returns the callback function that is set for the menu item.

Each item has space for a callback function and an argument for that function. Due to back compatibility, the [Fl_Menu_Item](#) itself is not passed to the callback, instead you have to get it by calling `((Fl_Menu_*)w)->mvalue()` where *w* is the widget argument.

11.86.2.5 callback() [2/5]

```
void Fl_Menu_Item::callback (
    Fl_Callback * c) [inline]
```

Sets the menu item's callback function.

This method does not set the [userdata\(\)](#) argument.

See also

[Fl_Callback_p Fl_Menuitem::callback\(\) const](#)

11.86.2.6 callback() [3/5]

```
void Fl_Menu_Item::callback (
    Fl_Callback * c,
    void * p) [inline]
```

Sets the menu item's callback function and [userdata\(\)](#) argument.

See also

[Fl_Callback_p Fl_Menuitem::callback\(\) const](#)

11.86.2.7 callback() [4/5]

```
void Fl_Menu_Item::callback (
    Fl_Callback0 * c) [inline]
```

Sets the menu item's callback function.

This method does not set the [userdata\(\)](#) argument.

See also

[Fl_Callback_p Fl_Menuitem::callback\(\) const](#)

11.86.2.8 callback() [5/5]

```
void Fl_Menu_Item::callback (
    Fl_Callback1 * c,
    long p = 0) [inline]
```

Sets the menu item's callback function and userdata() argument.

The argument `is` is cast to `void*` and stored as the `userdata()` for the menu item's callback function.

See also

[Fl_Callback_p](#) `Fl_MenuItem::callback()` `const`

11.86.2.9 check()

```
void Fl_Menu_Item::check () [inline]
```

Back compatibility only.

Deprecated Please use [Fl_Menu_Item::set\(\)](#) instead. This method will be removed in FLTK 1.5.0 or later.

See also

[set\(\)](#)

11.86.2.10 checkbox()

```
int Fl_Menu_Item::checkbox () const [inline]
```

Returns true if a checkbox will be drawn next to this item.

This is true if `FL_MENU_TOGGLE` or `FL_MENU_RADIO` is set in the flags.

11.86.2.11 checked()

```
int Fl_Menu_Item::checked () const [inline]
```

Back compatibility only.

Deprecated Please use [Fl_Menu_Item::value\(\)](#) instead. This method will be removed in FLTK 1.5.0 or later.

See also

[value\(\)](#)

11.86.2.12 deactivate()

```
void Fl_Menu_Item::deactivate () [inline]
```

Prevents a menu item from being picked.

Note that this will also cause the menu item to appear grayed-out.

11.86.2.13 do_callback() [1/3]

```
void Fl_Menu_Item::do_callback (
    Fl_Widget * o) const [inline]
```

Calls the [Fl_Menu_Item](#) item's callback, and provides the [Fl_Widget](#) argument.

The callback is called with the stored [user_data\(\)](#) as its second argument. You must first check that [callback\(\)](#) is non-zero before calling this.

11.86.2.14 do_callback() [2/3]

```
void Fl_Menu_Item::do_callback (
    Fl_Widget * o,
    long arg) const [inline]
```

Calls the [Fl_Menu_Item](#) item's callback, and provides the [Fl_Widget](#) argument.

This call overrides the callback's second argument with the given value `arg`. `long arg` is cast to `void*` when calling the callback. You must first check that [callback\(\)](#) is non-zero before calling this.

11.86.2.15 do_callback() [3/3]

```
void Fl_Menu_Item::do_callback (
    Fl_Widget * o,
    void * arg) const [inline]
```

Calls the [Fl_Menu_Item](#) item's callback, and provides the [Fl_Widget](#) argument.

This call overrides the callback's second argument with the given value `arg`. You must first check that [callback\(\)](#) is non-zero before calling this.

11.86.2.16 find_shortcut()

```
const Fl_Menu_Item * Fl_Menu_Item::find_shortcut (
    int * ip = 0,
    const bool require_alt = false) const
```

Search only the top level menu for a shortcut.

Either &x in the label or the shortcut fields are used.

This tests the current event, which must be an `FL_KEYBOARD` or `FL_SHORTCUT`, against a shortcut value.

Parameters

<i>ip</i>	returns the index of the item, if <i>ip</i> is not NULL.
<i>require_alt</i>	if true: match only if Alt key is pressed.

Returns

found [Fl_Menu_Item](#) or NULL

11.86.2.17 image() [1/2]

```
void Fl_Menu_Item::image (
    Fl_Image & image) [inline]
```

Compatibility API for FLUID, same as `image.label(this)`.

Note

This method is intended for internal use by fluid and may not do what you expect.

11.86.2.18 image() [2/2]

```
void Fl_Menu_Item::image (
    Fl_Image * image) [inline]
```

Compatibility API for FLUID, same as `image->label(this)`.

Note

This method is intended for internal use by fluid and may not do what you expect.

11.86.2.19 image_label()

```
void Fl_Menu_Item::image_label (
    const Fl_Image * image) [inline]
```

Sets the title ([label\(\)](#)) to an icon or image.

This sets the [labeltype\(\)](#) to `_FL_IMAGE_LABEL` (note the leading underscore).

See also

`const char* Fl\_Menu\_Item::label\(\) const`

Since

1.4.0

11.86.2.20 insert()

```
int Fl_Menu_Item::insert (
    int index,
    const char * mytext,
    int sc,
    Fl_Callback * cb,
    void * data = 0,
    int myflags = 0)
```

Inserts an item at position `index`.

If `index` is -1, the item is added the same way as [Fl_Menu_Item::add\(\)](#).

If 'mytext' contains any un-escaped front slashes (/), it's assumed a menu pathname is being specified, and the value of `index` will be ignored.

In all other aspects, the behavior of [insert\(\)](#) is the same as [add\(\)](#).

Parameters

in	<i>index</i>	insert new items here
in	<i>mytext</i>	new label string, details see above
in	<i>sc</i>	keyboard shortcut for new item
in	<i>cb</i>	callback function for new item
in	<i>data</i>	user data for new item
in	<i>myflags</i>	menu flags as described in Fl_Menu_Item

Returns

the index into the menu() array, where the entry was added

11.86.2.21 label() [1/3]

```
const char * Fl_Menu_Item::label () const [inline]
```

Returns the title (label) of the menu item.

A NULL here indicates the end of the menu (or of a submenu). A '&' in the item will print an underscore under the next letter, and if the menu is popped up that letter will be a "shortcut" to pick that item. To get a real '&' put two in a row.

The returned value depends on the [labeltype\(\)](#) that has been used to store the label.

Returns

A pointer to the label cast to (const char *)

Return values

(a)	A pointer to text (const char *)
(b)	A pointer to an image (Fl_Image *)
(c)	A pointer to a "multi label" (Fl_Multi_Label *)
NULL	End of menu or submenu

See also

[Fl_Menu_Item::label\(const char *\)](#)

[Fl_Menu_Item::label\(Fl_Labeltype, const char *\)](#)

[Fl_Menu_Item::image_label\(const Fl_Image *\)](#)

[Fl_Menu_Item::multi_label\(const Fl_Multi_Label *\)](#)

[Fl_Multi_Label::label\(Fl_Menu_Item *\)](#)

11.86.2.22 label() [2/3]

```
void Fl_Menu_Item::label (
    const char * a) [inline]
```

Sets the title (label) of the menu item.

Note

This does **not** change the [labeltype\(\)](#) for backwards compatibility. Take care to assign the correct [labeltype\(\)](#) if you assign different label types to the same menu item sequentially.

The default [labeltype\(\)](#) is FL_NORMAL_LABEL.

See also

[label\(Fl_Labeltype, const char*\)](#)
[const char* Fl_Menu_Item::label\(\) const](#)

11.86.2.23 label() [3/3]

```
void Fl_Menu_Item::label (
    Fl_Labeltype a,
    const char * b) [inline]
```

Sets the title (label) and the label type of the menu item.

The default Fl_Labeltype when using [Fl_Menu_Item::label\(const char* a\)](#) is FL_NORMAL_LABEL but you can set any other label type, for instance FL_EMBOSSSED_LABEL.

Special labeltypes are:

- FL_ICON_LABEL: draws the icon ([Fl_Image](#)) associated with the text
- FL_IMAGE_LABEL: draws the image ([Fl_Image](#)) associated with the text
- FL_MULTI_LABEL: draws multiple parts side by side, see [Fl_Multi_Label](#)

See also

[const char* Fl_Menu_Item::label\(\) const](#)

11.86.2.24 labelcolor() [1/2]

```
Fl_Color Fl_Menu_Item::labelcolor () const [inline]
```

Gets the menu item's label color.

This color is passed to the labeltype routine, and is typically the color of the label text. This defaults to FL_BLACK. If this color is not black fltk will **not** use overlay bitplanes to draw the menu - this is so that images put in the menu draw correctly.

11.86.2.25 labelcolor() [2/2]

```
void Fl_Menu_Item::labelcolor (
    Fl_Color a) [inline]
```

Sets the menu item's label color.

See also

[Fl_Color Fl_Menu_Item::labelcolor\(\) const](#)

11.86.2.26 labelfont() [1/2]

```
Fl_Font Fl_Menu_Item::labelfont () const [inline]
```

Gets the menu item's label font.

Fonts are identified by small 8-bit indexes into a table. See the enumeration list for predefined fonts. The default value is a Helvetica font. The function [Fl::set_font\(\)](#) can define new fonts.

11.86.2.27 labelfont() [2/2]

```
void Fl_Menu_Item::labelfont (
    Fl_Font a) [inline]
```

Sets the menu item's label font.

Fonts are identified by small 8-bit indexes into a table. See the enumeration list for predefined fonts. The default value is a Helvetica font. The function [Fl::set_font\(\)](#) can define new fonts.

11.86.2.28 labeltype() [1/2]

```
Fl_Labeltype Fl_Menu_Item::labeltype () const [inline]
```

Returns the menu item's labeltype.

A labeltype identifies a routine that draws the label of the widget. This can be used for special effects such as emboss, or to use the [label\(\)](#) pointer as another form of data such as a bitmap. The value FL_NORMAL_LABEL prints the label as text.

11.86.2.29 labeltype() [2/2]

```
void Fl_Menu_Item::labeltype (
    Fl_Labeltype a) [inline]
```

Sets the menu item's labeltype.

A labeltype identifies a routine that draws the label of the widget. This can be used for special effects such as emboss, or to use the [label\(\)](#) pointer as another form of data such as a bitmap. The value FL_NORMAL_LABEL prints the label as text.

11.86.2.30 measure()

```
int Fl_Menu_Item::measure (
    int * hp,
    const Fl_Menu_ * m) const
```

Measures width of label, including effect of & characters.

Optionally, can get height if hp is not NULL.

11.86.2.31 multi_label()

```
void Fl_Menu_Item::multi_label (
    const Fl_Multi_Label * ml) [inline]
```

Sets the title ([label\(\)](#)) and [labeltype\(\)](#) to an [Fl_Multi_Label](#).

This sets the [labeltype\(\)](#) to _FL_MULTI_LABEL (note the leading underscore).

See also

```
const char* Fl_Menu_Item::label() const
```

Since

1.4.0

11.86.2.32 next() [1/2]

```
Fl_Menu_Item * Fl_Menu_Item::next (
    int i = 1) [inline]
```

Advances a pointer by n items through a menu array, skipping the contents of submenus and invisible items.

There are two calls so that you can advance through const and non-const data.

11.86.2.33 next() [2/2]

```
const Fl_Menu_Item * Fl_Menu_Item::next (
    int n = 1) const
```

Advance a pointer by n items through a menu array, skipping the contents of submenus and invisible items.

There are two calls so that you can advance through const and non-const data.

11.86.2.34 popup()

```
const Fl_Menu_Item * Fl_Menu_Item::popup (
    int X,
    int Y,
    const char * title = 0,
    const Fl_Menu_Item * picked = 0,
    const Fl_Menu_ * menu_button = 0) const
```

This method is called by widgets that want to display menus.

The menu stays up until the user picks an item or dismisses it. The selected item (or NULL if none) is returned. *This does not do the callbacks or change the state of check or radio items.*

The menu is positioned so the cursor is centered over the item picked. This will work even if `picked` is in a submenu. If `picked` is zero or not in the menu item table the menu is positioned with the cursor in the top-left corner.

Parameters

in	<i>X,Y</i>	the position of the mouse cursor, relative to the window that got the most recent event (usually you can pass <code>Fl::event_x()</code> and <code>Fl::event_y()</code> unchanged here).
in	<i>title</i>	a character string title for the menu. If non-zero a small box appears above the menu with the title in it.
in	<i>picked</i>	if this pointer is not NULL, the popup menu will appear so that the picked menu is under the mouse pointer.
in	<i>menu_button</i>	is a pointer to an <code>Fl_Menu_</code> from which the color and boxtypes for the menu are pulled. If NULL then defaults are used.

Returns

a pointer to the menu item selected by the user, or NULL

11.86.2.35 pulldown()

```
const Fl_Menu_Item * Fl_Menu_Item::pulldown (
    int X,
    int Y,
    int W,
    int H,
    const Fl_Menu_Item * initial_item = 0,
    const Fl_Menu_ * pbutton = 0,
    const Fl_Menu_Item * title = 0,
    int menubar = 0) const
```

Pulldown() is similar to `popup()`, but a rectangle is provided to position the menu.

The menu is made at least `W` wide, and the picked item `initial_item` is centered over the rectangle (like `Fl_Choice` uses).

If `initial_item` is NULL or not found, the menu is aligned just below the rectangle (like a pulldown menu).

The `title` and `menubar` arguments are used internally by the `Fl_Menu_Bar` widget.

11.86.2.36 radio()

```
int Fl_Menu_Item::radio () const [inline]
```

Returns true if this item is a radio item.

When a radio button is selected all "adjacent" radio buttons are turned off. A set of radio items is delimited by an item that has `radio()` false, or by an item with `FL_MENU_DIVIDER` turned on.

11.86.2.37 set()

```
void Fl_Menu_Item::set () [inline]
```

Turns the check or radio item "on" for the menu item.

Note that this does not turn off any adjacent radio items like `setonly()` does.

11.86.2.38 setonly()

```
void Fl_Menu_Item::setonly (
    Fl_Menu_Item const * first = NULL)
```

Turns the radio item "on" for the menu item and turns "off" adjacent radio items set.

Note

This method is dangerous if radio items are first in the menu. Make sure that *first* is set correctly or use [Fl_Menu_::setonly\(Fl_Menu_Item*\)](#) instead.

Parameters

<i>in</i>	<i>first</i>	start of menu array or NULL (default) if the radio group is not the first item

11.86.2.39 shortcut()

```
void Fl_Menu_Item::shortcut (
    int s) [inline]
```

Sets exactly what key combination will trigger the menu item.

The value is a logical 'or' of a key and a set of shift flags, for instance FL_ALT+'a' or FL_ALT+FL_F+10 or just 'a'. A value of zero disables the shortcut.

The key can be any value returned by [Fl::event_key\(\)](#), but will usually be an ASCII letter. Use a lower-case letter unless you require the shift key to be held down.

The shift flags can be any set of values accepted by [Fl::event_state\(\)](#). If the bit is on that shift key must be pushed. Meta, Alt, Ctrl, and Shift must be off if they are not in the shift flags (zero for the other bits indicates a "don't care" setting).

11.86.2.40 size()

```
int Fl_Menu_Item::size () const
```

Size of the menu starting from this menu item.

This method counts all menu items starting with *this* menu item, including all menu items in the same (sub)menu level, all nested submenus, **and** the terminating empty (0) menu item.

It does **not** count menu items referred to by FL_SUBMENU_POINTER menu items (except the single menu item with FL_SUBMENU_POINTER).

All menu items counted are consecutive in memory (one array).

Example:

```
schemechoice = new Fl_Choice(X+125,Y,140,25,"FLTK Scheme");
schemechoice->add("none");
schemechoice->add("plastic");
schemechoice->add("gtk+");
schemechoice->add("gleam");
printf("schemechoice->menu()->size() = %d\n", schemechoice->menu()->size());
```

Output:

```
schemechoice->menu()->size() = 5
```

11.86.2.41 submenu()

```
int Fl_Menu_Item::submenu () const [inline]
```

Returns true if either FL_SUBMENU or FL_SUBMENU_POINTER is on in the flags.

FL_SUBMENU indicates an embedded submenu that goes from the next item through the next one with a NULL [label\(\)](#). FL_SUBMENU_POINTER indicates that [user_data\(\)](#) is a pointer to another menu array.

11.86.2.42 test_shortcut()

```
const Fl_Menu_Item * Fl_Menu_Item::test_shortcut () const
```

This is designed to be called by a widgets handle() method in response to a FL_SHORTCUT event.

If the current event matches one of the items shortcut, that item is returned. If the keystroke does not match any shortcuts then NULL is returned. This only matches the [shortcut\(\)](#) fields, not the letters in the title preceeded by ' '

11.86.2.43 `unchecked()`

```
void Fl_Menu_Item::unchecked () [inline]
```

Back compatibility only.

Deprecated Please use `Fl_Menu_Item::clear()` instead. This method will be removed in FLTK 1.5.0 or later.

See also

[clear\(\)](#)

11.86.2.44 `value()`

```
int Fl_Menu_Item::value () const [inline]
```

Returns the current value of the check or radio item.

This is zero (0) if the menu item is not checked and non-zero otherwise.

Since

1.4.0 this method returns 1 if the item is checked but you should not rely on a particular value, only zero or non-zero.

Note

The returned value for a checked menu item was `FL_MENU_VALUE` (4) before FLTK 1.4.0.

The documentation for this struct was generated from the following files:

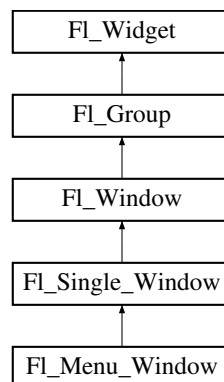
- [Fl_Menu_Item.H](#)
- `Fl_Menu.cxx`
- `Fl_Menu_.cxx`
- `Fl_Menu_add.cxx`

11.87 `Fl_Menu_Window` Class Reference

The `Fl_Menu_Window` widget is a window type used for menus.

```
#include <Fl_Menu_Window.H>
```

Inheritance diagram for `Fl_Menu_Window`:



Public Member Functions

- **`Fl_Menu_Window`** (int W, int H, const char *l=0)
Creates a new `Fl_Menu_Window` widget using the given size, and label string.
- **`Fl_Menu_Window`** (int X, int Y, int W, int H, const char *l=0)
Creates a new `Fl_Menu_Window` widget using the given position, size, and label string.
- **`~Fl_Menu_Window`** ()
Destroys the window and all of its children.

Public Member Functions inherited from [FL_Single_Window](#)

- **FL_Single_Window** (int W, int H, const char *l=0)
Creates a new [FL_Single_Window](#) widget using the given size, and label (title) string.
- **FL_Single_Window** (int X, int Y, int W, int H, const char *l=0)
Creates a new [FL_Single_Window](#) widget using the given position, size, and label (title) string.
- void **flush** () [FL_OVERRIDE](#)
Same as [FL_Window::flush\(\)](#)
- void **make_current** ()
Same as [FL_Window::make_current\(\)](#)
- void **show** () [FL_OVERRIDE](#)
Makes a widget visible.
- void **show** (int argc, char **argv)
*Same as [FL_Window::show\(int argc, char **argv\)](#)*

Public Member Functions inherited from [FL_Window](#)

- void **allow_expand_outside_parent** ()
Allow this subwindow to expand outside the area of its parent window.
- virtual class [FL_Double_Window](#) * **as_double_window** ()
Return non-null if this is an [FL_Double_Window](#) object.
- virtual class [FL_Overlay_Window](#) * **as_overlay_window** ()
Return non-null if this is an [FL_Overlay_Window](#) object.
- [FL_Window](#) const * **as_window** () const [FL_OVERRIDE](#)
- [FL_Window](#) * **as_window** () [FL_OVERRIDE](#)
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).
- unsigned int **border** () const
Returns whether the window possesses a border.
- void **border** (int b)
Sets whether or not the window manager border is around the window.
- void **clear_border** ()
Fast inline function to turn the window manager border off.
- void **clear_modal_states** ()
Clears the "modal" flags and converts a "modal" or "non-modal" window back into a "normal" window.
- void **copy_label** (const char *a)
Sets the window titlebar label to a copy of a character string.
- void **cursor** (const [FL_RGB_Image](#) *, int, int)
Changes the cursor for this window using the provided image as cursor's shape.
- void **cursor** ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=FL_WHITE)
For back compatibility only.
- void **cursor** ([FL_Cursor](#))
Changes the cursor for this window.
- int **decorated_h** () const
Returns the window height including any window title bar and any frame added by the window manager.
- int **decorated_w** () const
Returns the window width including any frame added by the window manager.
- void **default_cursor** ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=FL_WHITE)
For back compatibility only.
- void **default_cursor** ([FL_Cursor](#))
Sets the default window cursor.
- void **draw_backdrop** ()
Draw the background image if one is set and is aligned inside.

- [FL_Window](#) (int *w*, int *h*, const char **title*=0)

*Creates a window from the given width *w*, height *h*, and *title*.*
- [FL_Window](#) (int *x*, int *y*, int *w*, int *h*, const char **title*=0)

*Creates a window from the given position (*x*, *y*), size (*w*, *h*) and *title*.*
- void [free_position](#) ()

Undoes the effect of a previous [resize\(\)](#) or [show\(\)](#) so that the next time [show\(\)](#) is called the window manager is free to position the window.
- void [fullscreen](#) ()

Makes the window completely fill one or more screens, without any window manager border visible.
- unsigned int [fullscreen_active](#) () const

Returns non zero if FULLSCREEN flag is set, 0 otherwise.
- void [fullscreen_off](#) ()

Turns off any side effects of [fullscreen\(\)](#)
- void [fullscreen_off](#) (int *X*, int *Y*, int *W*, int *H*)

Turns off any side effects of [fullscreen\(\)](#) and does [resize\(x,y,w,h\)](#).
- void [fullscreen_screens](#) (int *top*, int *bottom*, int *left*, int *right*)

Sets which screens should be used when this window is in fullscreen mode.
- [uchar](#) [get_size_range](#) (int **minw*, int **minh*, int **maxw*=NULL, int **maxh*=NULL, int **dw*=NULL, int **dh*=NULL, int **aspect*=NULL)

Gets the allowable range to which the user can resize this window.
- int [handle](#) (int) [FL_OVERRIDE](#)

Handles the specified event.
- void [hide](#) () [FL_OVERRIDE](#)

Removes the window from the screen.
- void [hotspot](#) (const [FL_Widget](#) &*p*, int *offscreen*=0)

*See void [FL_Window::hotspot](#)(int *x*, int *y*, int *offscreen* = 0)*
- void [hotspot](#) (const [FL_Widget](#) *, int *offscreen*=0)

*See void [FL_Window::hotspot](#)(int *x*, int *y*, int *offscreen* = 0)*
- void [hotspot](#) (int *x*, int *y*, int *offscreen*=0)

Positions the window so that the mouse is pointing at the given position, or at the center of the given widget, which may be the window itself.
- const void * [icon](#) () const

Gets the current icon window target dependent data.
- void [icon](#) (const [FL_RGB_Image](#) *)

Sets or resets a single window icon.
- void [icon](#) (const void **ic*)

Platform-specific method to set the window icon usable on Windows and X11 only.
- void [iconize](#) ()

Iconifies the window.
- const char * [iconlabel](#) () const

See void [FL_Window::iconlabel](#)(const char)*
- void [iconlabel](#) (const char *)

Sets the icon label.
- void [icons](#) (const [FL_RGB_Image](#) *[], int)

Sets the window icons.
- void [icons](#) (HICON *big_icon*, HICON *small_icon*)

Sets the window icons using HICON handles (Windows platform only).
- const char * [label](#) () const

See void [FL_Window::label](#)(const char)*
- void [label](#) (const char *)

- Sets the window title bar label.*

 - void **label** (const char *label, const char *iconlabel)
- Sets the icon label.*

 - void **make_current** ()
- Sets things up so that the drawing functions in <FL/fl_draw.H> will go into this window.*

 - void **maximize** ()
- Maximizes a top-level window to its current screen.*

 - unsigned int **maximize_active** () const
- Returns whether the window is currently maximized.*

 - unsigned int **menu_window** () const
- Returns true if this window is a menu window.*

 - unsigned int **modal** () const
- Returns true if this window is modal.*

 - unsigned int **non_modal** () const
- Returns true if this window is modal or non-modal.*

 - **fl_uintptr_t os_id** ()
- Returns a platform-specific identification of a shown window, or 0 if not shown.*

 - unsigned int **override** () const
- Returns non zero if OVERRIDE flag is set, 0 otherwise.*

 - void **resize** (int X, int Y, int W, int H) **FL_OVERRIDE**
- Changes the size and position of the window.*

 - int **screen_num** ()
- The number of the screen containing the mapped window.*

 - void **screen_num** (int screen_num)
- Set the number of the screen where to map the window.*

 - void **set_menu_window** ()
- Marks the window as a menu window.*

 - void **set_modal** ()
- A "modal" window, when **shown()**, will prevent any events from being delivered to other windows in the same program, and will also remain on top of the other windows (if the X window manager supports the "transient for" property).*

 - void **set_non_modal** ()
- A "non-modal" window (terminology borrowed from Microsoft Windows) acts like a **modal()** one in that it remains on top, but it has no effect on event delivery.*

 - void **set_override** ()
- Activates the flags NOBORDER|OVERRIDE.*

 - void **set_tooltip_window** ()
- Marks the window as a tooltip window.*

 - const **Fl_Image** * **shape** ()
- Returns the image controlling the window shape or NULL.*

 - void **shape** (const **Fl_Image** &b)
- Set the window's shape with an **Fl_Image**.*

 - void **shape** (const **Fl_Image** *img)
- Assigns a non-rectangular shape to the window.*

 - void **show** () **FL_OVERRIDE**
- Puts the window on the screen.*

 - void **show** (int argc, char **argv)
- Puts the window on the screen with **show()** and parses command-line arguments.*

 - int **shown** ()
- Returns non-zero if **show()** has been called (but not **hide()**).*

 - void **size_range** (int minw, int minh, int maxw=0, int maxh=0, int dw=0, int dh=0, int aspect=0)
- Sets the allowable range to which the user can resize this window.*

- unsigned int **tooltip_window** () const
Returns true if this window is a tooltip window.
- void **un_maximize** ()
Returns a previously maximized top-level window to its previous size.
- void **wait_for_expose** ()
Waits for the window to be displayed after calling [show\(\)](#).
- int **x_root** () const
Gets the x position of the window on the screen.
- const char * **xclass** () const
Returns the xclass for this window, or a default.
- void **xclass** (const char *c)
Sets the xclass for this window.
- int **y_root** () const
Gets the y position of the window on the screen.
- virtual **~FL_Window** ()
The destructor also deletes all the children.

Public Member Functions inherited from **FL_Group**

- **FL_Widget** *& **_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** (**FL_Widget** &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** (**FL_Widget** *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void **add_resizable** (**FL_Widget** &o)
Adds a widget to the group and makes it the resizable widget.
- **FL_Widget** *const * **array** () const
Returns a pointer to the internal array of children.
- **FL_Group** const * **as_group** () const **FL_OVERRIDE**
- **FL_Group** * **as_group** () **FL_OVERRIDE**
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- **FL_Widget** * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
*Removes the widget at *index* from the group and deletes it.*
- void **end** ()
*Exactly the same as *current(this->parent())*.*
- int **find** (const **FL_Widget** &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int **find** (const **FL_Widget** *) const

- Searches the children for the widget and returns the index.*
- `FL_Group` (int, int, int, int, const char * = 0)
 - Creates a new `FL_Group` widget using the given position, size, and label string.*
- void `focus` (`FL_Widget` *W)
- void `forms_end` ()
 - This is for forms compatibility only.*
- void `init_sizes` ()
 - Resets the internal array of widget sizes and positions.*
- void `insert` (`FL_Widget` &, int i)
 - The widget is removed from its current group (if any) and then inserted into this group.*
- void `insert` (`FL_Widget` &o, `FL_Widget` *before)
 - This does `insert(w, find(before))`.*
- void `remove` (`FL_Widget` &)
 - Removes a widget from the group but does not delete it.*
- void `remove` (`FL_Widget` *o)
 - Removes the widget o from the group.*
- void `remove` (int index)
 - Removes the widget at `index` from the group but does not delete it.*
- `FL_Widget` * `resizable` () const
 - Returns the group's resizable widget.*
- void `resizable` (`FL_Widget` &o)
 - Sets the group's resizable widget.*
- void `resizable` (`FL_Widget` *o)
 - The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
- virtual `~FL_Group` ()
 - The destructor also deletes all the children.*

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
 - Activates the widget.*
- unsigned int `active` () const
 - Returns whether the widget is active.*
- int `active_r` () const
 - Returns whether the widget and all of its parents are active.*
- `FL_Align` `align` () const
 - Gets the label alignment.*
- void `align` (`FL_Align` alignment)
 - Sets the label alignment.*
- long `argument` () const
 - Gets the current user data (long) argument that is passed to the callback function.*
- void `argument` (long v)
 - Sets the current user data (long) argument that is passed to the callback function.*
- virtual class `FL_Gl_Window` * `as_gl_window` ()
 - Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.*
- virtual class `FL_Gl_Window` const * `as_gl_window` () const
- void `bind_deimage` (`FL_Image` *img)
 - Sets the image to use as part of the widget label when in the inactive state.*
- void `bind_deimage` (int f)

- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*

 - void `bind_image` (`FI_Image` *img)

Sets the image to use as part of the widget label when in the active state.
- void `bind_image` (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- `FI_Boxtype` `box` () const

Gets the box type of the widget.
- void `box` (`FI_Boxtype` new_box)

Sets the box type for the widget.
- `FI_Callback_p` `callback` () const

Gets the current callback function for the widget.
- void `callback` (`FI_Callback` *cb)

Sets the current callback function for the widget.
- void `callback` (`FI_Callback` *cb, `FI_Callback_User_Data` *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void `callback` (`FI_Callback` *cb, void *p)

Sets the current callback function and data for the widget.
- void `callback` (`FI_Callback0` *cb)

Sets the current callback function for the widget.
- void `callback` (`FI_Callback1` *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int `changed` () const

Checks if the widget value changed since the last callback.
- void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FI_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FI_Color` bg)

Sets the background color of the widget.
- void `color` (`FI_Color` bg, `FI_Color` sel)

Sets the background and selection color of the widget.
- `FI_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FI_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.

- `uchar damage () const`
Returns non-zero if `draw()` needs to be called.
- `void damage (uchar c)`
Sets the damage bits for the widget.
- `void damage (uchar c, int x, int y, int w, int h)`
Sets the damage bits for an area inside the widget.
- `int damage_resize (int, int, int, int)`
Internal use only.
- `void deactivate ()`
Deactivates the widget.
- `FL_Image * deimage ()`
Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`
Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`
Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`
Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`
Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FL_Align) const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`
Gets the widget height.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FL_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FL_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FL_Widget *wgt) const`
Checks if this widget is a child of `wgt`.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`

- Sets the current label pointer.*

 - void `label` (`FI_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const

Gets the label color.
- void `labelcolor` (`FI_Color` c)

Sets the label color.
- `FI_Font` `labelfont` () const

Gets the font to use.
- void `labelfont` (`FI_Font` f)

Sets the font to use.
- `FI_Fonsize` `labelsize` () const

Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)

Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

Gets the label type.
- void `labeltype` (`FI_Labeltype` a)

Sets the label type.
- void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FI_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FI_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- `FI_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`FI_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.

- void `set_visible` ()
Makes the widget visible.
- void `set_visible_focus` ()
Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- void `size` (int W, int H)
Changes the size of the widget.
- int `take_focus` ()
Gives the widget the keyboard focus.
- unsigned int `takeevents` () const
Returns if the widget is able to take events.
- int `test_shortcut` ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const
Gets the current tooltip text.
- void `tooltip` (const char *text)
Sets the current tooltip text.
- `Fl_Window` * `top_window` () const
Returns a pointer to the top-level window for the widget.
- `Fl_Window` * `top_window_offset` (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar` `type` () const
Gets the widget type.
- void `type` (`uchar` t)
Sets the widget type.
- int `use_accents_menu` ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * `user_data` () const
Gets the user data for this widget.
- void `user_data` (`Fl_Callback_User_Data` *v, bool auto_free)
Sets the user data for this widget.
- void `user_data` (void *v)
Sets the user data for this widget.
- int `vertical_label_margin` ()
Get the spacing between the label and the vertical edge of the widget.
- void `vertical_label_margin` (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int `visible` () const
Returns whether a widget is visible.
- unsigned int `visible_focus` () const
Checks whether this widget has a visible focus.
- void `visible_focus` (int v)
Modifies keyboard focus navigation.
- int `visible_r` () const
Returns whether a widget and all its parents are visible.
- int `w` () const
Gets the widget width.
- `Fl_When` `when` () const

- Returns the conditions under which the callback is called.*

 - void `when` (`uchar` i)

Sets the flags used to decide when a callback is called.
- `Fl_Window * window` () const
- Returns a pointer to the nearest parent window up the widget hierarchy.*
- int `x` () const
- Gets the widget position in its window.*
- int `y` () const
- Gets the widget position in its window.*
- virtual `~Fl_Widget` ()
- Destroys the widget.*

Additional Inherited Members

Public Types inherited from `Fl_Window`

- typedef struct HICON__ * `HICON`

Static Public Member Functions inherited from `Fl_Window`

- static `Fl_Window * current` ()
- Returns the last window that was made current.*
- static void `default_callback` (`Fl_Window *`, void *v)
- Back compatibility: Sets the default callback v for win to call on close event.*
- static void `default_icon` (const `Fl_RGB_Image *`)
- Sets a single default window icon.*
- static void `default_icons` (const `Fl_RGB_Image *`[], int)
- Sets the default window icons.*
- static void `default_icons` (HICON big_icon, HICON small_icon)
- Sets the default window icons (Windows platform only).*
- static const char * `default_xclass` ()
- Returns the default xclass.*
- static void `default_xclass` (const char *)
- Sets the default window xclass.*
- static bool `is_a_rescale` ()
- Returns true when a window is being rescaled.*
- static char `show_next_window_iconic` ()
- Returns the static flag whether the next window should be opened iconified.*
- static void `show_next_window_iconic` (char stat)
- Sets a static flag whether the next window should be opened iconified.*

Static Public Member Functions inherited from `Fl_Group`

- static `Fl_Group * current` ()
- Returns the currently active group.*
- static void `current` (`Fl_Group *g`)
- Sets the current group.*

Static Public Member Functions inherited from FI_Widget

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from FI_Window

- void [default_size_range](#) ()
Protected method to calculate the default size range of a window.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- int [force_position](#) () const
Returns the internal state of the window's FORCE_POSITION flag.
- void [force_position](#) (int force)
Sets an internal flag that tells FLTK and the window manager to honor position requests.
- void [free_icons](#) ()
Deletes all icons previously attached to the window.
- int [is_resizable](#) ()
Protected method to determine whether a window is resizable.

Protected Member Functions inherited from FI_Group

- [FI_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw_child](#) ([FI_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [FI_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) ([FI_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.

- virtual int **on_move** (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void **on_remove** (int)
Allow derived groups to act when a child widget is removed from the group.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** (FI_Widget &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Static Protected Attributes inherited from FI_Window

- static FI_Window * **current_**
Stores the last window that was made current.

11.87.1 Detailed Description

The [FI_Menu_Window](#) widget is a window type used for menus.

By default the window is drawn in the hardware overlay planes if they are available so that the menu don't force the rest of the window to redraw.

The documentation for this class was generated from the following files:

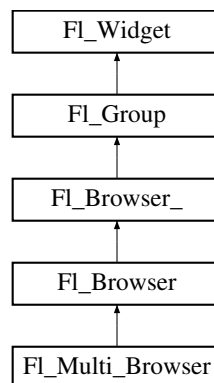
- [FI_Menu_Window.H](#)
- [FI_Menu_Window.cxx](#)

11.88 FI_Multi_Browser Class Reference

The [FI_Multi_Browser](#) class is a subclass of [FI_Browser](#) which lets the user select any set of the lines.

```
#include <FI_Multi_Browser.H>
```

Inheritance diagram for [FI_Multi_Browser](#):



Public Member Functions

- [FI_Multi_Browser](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [FI_Multi_Browser](#) widget using the given position, size, and label string.

Public Member Functions inherited from [FI_Browser](#)

- void [add](#) (const char *newtext, void *d=0)
Adds a new line to the end of the browser.
- void [bottomline](#) (int line)
*Scrolls the browser so the bottom item in the browser is showing the specified *line*.*
- void [clear](#) ()
Removes all the lines in the browser.
- char [column_char](#) () const
Gets the current column separator character.
- void [column_char](#) (char c)
*Sets the column separator to *c*.*
- const int * [column_widths](#) () const
Gets the current column width array.
- void [column_widths](#) (const int *arr)
*Sets the current array to *arr*.*
- void * [data](#) (int line) const
*Returns the user *data()* for specified *line*.*
- void [data](#) (int line, void *d)
*Sets the user data for specified *line* to *d*.*

- void **display** (int line, int val=1)
For back compatibility.
- int **displayed** (int line) const
Returns non-zero if `line` has been scrolled to a position where it is being displayed.
- **FL_Browser** (int X, int Y, int W, int H, const char *L=0)
The constructor makes an empty browser.
- char **format_char** () const
Gets the current format code prefix character, which by default is '@'.
- void **format_char** (char c)
Sets the current format code prefix character to `c`.
- void **hide** () **FL_OVERRIDE**
Hides the entire `FL_Browser` widget – opposite of `show()`.
- void **hide** (int line)
Makes `line` invisible, preventing selection by the user.
- **FL_Image** * **icon** (int line) const
Returns the icon currently defined for `line`.
- void **icon** (int line, **FL_Image** *icon)
Set the image icon for `line` to the value `icon`.
- void **insert** (int line, const char *newtext, void *d=0)
Insert a new entry whose label is `newtext` above given `line`, optional data `d`.
- void **lineposition** (int line, **FL_Line_Position** pos)
Updates the browser so that `line` is shown at position `pos`.
- int **load** (const char *filename)
Clears the browser and reads the file, adding each line from the file to the browser.
- void **make_visible** (int line)
Make the item at the specified `line` `visible()`.
- void **middleline** (int line)
Scrolls the browser so the middle item in the browser is showing the specified `line`.
- void **move** (int to, int from)
Line `from` is removed and reinserted at `to`.
- void **remove** (int line)
Remove entry for given `line` number, making the browser one line shorter.
- void **remove_icon** (int line)
Removes the icon for `line`.
- void **replace** (int a, const char *b)
For back compatibility only.
- int **select** (int line, int val=1)
Sets the selection state of the item at `line` to the value `val`.
- int **selected** (int line) const
Returns 1 if specified `line` is selected, 0 if not.
- void **show** () **FL_OVERRIDE**
Shows the entire `FL_Browser` widget – opposite of `hide()`.
- void **show** (int line)
Makes `line` visible, and available for selection by user.
- int **size** () const
Returns how many lines are in the browser.
- void **size** (int W, int H)
- void **swap** (int a, int b)
Swaps two browser lines `a` and `b`.
- const char * **text** (int line) const
Returns the label text for the specified `line`.

- void **text** (int line, const char *newtext)
Sets the text for the specified line to newtext.
- **FI_Fontsize** **textsize** () const
Gets the default text size (in pixels) for the lines in the browser.
- void **textsize** (**FI_Fontsize** newSize)
Sets the default text size (in pixels) for the lines in the browser to newSize.
- int **topline** () const
Returns the line that is currently visible at the top of the browser.
- void **topline** (int line)
Scrolls the browser so the top item in the browser is showing the specified line.
- int **value** () const
Returns the line number of the currently selected line, or 0 if none selected.
- void **value** (int line)
Sets the browser's value(), which selects the specified line.
- int **visible** (int line) const
Returns non-zero if the specified line is visible, 0 if hidden.
- ~**FI_Browser** ()
The destructor deletes all list items and destroys the browser.

Public Member Functions inherited from **FI_Browser_**

- int **deselect** (int docallbacks=0)
Deselects all items in the list and returns 1 if the state changed or 0 if it did not.
- void **display** (void *item)
Displays the item, scrolling the list as necessary.
- int **handle** (int event) **FL_OVERRIDE**
Handles the event within the normal widget bounding box.
- **uchar** **has_scrollbar** () const
Returns the current scrollbar mode, see FI_Browser_::has_scrollbar(uchar)
- void **has_scrollbar** (**uchar** mode)
Sets whether the widget should have scrollbars or not (default FI_Browser_::BOTH).
- int **hposition** () const
Gets the horizontal scroll position of the list as a pixel position pos.
- void **hposition** (int)
Sets the horizontal scroll position of the list to pixel position pos.
- int **linespacing** () const
Return the height of additional spacing between browser lines.
- void **linespacing** (int pixels)
Add some space between browser lines.
- int **position** () const
- void **position** (int pos)
- void **position** (int x, int y)
- void **resize** (int X, int Y, int W, int H) **FL_OVERRIDE**
Repositions and/or resizes the browser.
- void **scrollbar_left** ()
Moves the vertical scrollbar to the lefthand side of the list.
- void **scrollbar_right** ()
Moves the vertical scrollbar to the righthand side of the list.
- int **scrollbar_size** () const
Gets the current size of the scrollbars' troughs, in pixels.
- void **scrollbar_size** (int newSize)

- Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.*

 - `int scrollbar_width () const`
Returns the global value `Fl::scrollbar_size()`.
 - `void scrollbar_width (int width)`
Sets the global `Fl::scrollbar_size()`, and forces this instance of the widget to use it.
 - `int select (void *item, int val=1, int docallbacks=0)`
Sets the selection state of `item` to `val`, and returns 1 if the state changed or 0 if it did not.
 - `int select_only (void *item, int docallbacks=0)`
Selects `item` and returns 1 if the state changed or 0 if it did not.
 - `void sort (int flags=0)`
Sort the items in the browser based on `flags`.
 - `Fl_Color textcolor () const`
Gets the default text color for the lines in the browser.
 - `void textcolor (Fl_Color col)`
Sets the default text color for the lines in the browser to color `col`.
 - `Fl_Font textfont () const`
Gets the default text font for the lines in the browser.
 - `void textfont (Fl_Font font)`
Sets the default text font for the lines in the browser to `font`.
 - `Fl_Fontsize textsize () const`
Gets the default text size (in pixels) for the lines in the browser.
 - `void textsize (Fl_Fontsize newSize)`
Sets the default text size (in pixels) for the lines in the browser to `size`.
 - `int vposition () const`
Gets the vertical scroll position of the list as a pixel position `pos`.
 - `void vposition (int pos)`
Sets the vertical scroll position of the list to pixel position `pos`.

Public Member Functions inherited from `Fl_Group`

- `Fl_Widget *& _ddfdesign_kludge ()`
This is for forms compatibility only.
- `void add (Fl_Widget &)`
The widget is removed from its current group (if any) and then added to the end of this group.
- `void add (Fl_Widget *o)`
See void `Fl_Group::add(Fl_Widget &w)`
- `void add_resizable (Fl_Widget &o)`
Adds a widget to the group and makes it the resizable widget.
- `Fl_Widget *const * array () const`
Returns a pointer to the internal array of children.
- `Fl_Group const * as_group () const FL_OVERRIDE`
- `Fl_Group * as_group () FL_OVERRIDE`
Returns an `Fl_Group` pointer if this widget is an `Fl_Group`.
- `void begin ()`
Sets the current group so you can build the widget tree by just constructing the widgets.
- `Fl_Widget * child (int n) const`
Returns the `n`'th child.
- `int children () const`
Returns how many child widgets the group has.
- `void clear ()`
Deletes all child widgets from memory recursively.

- unsigned int [clip_children](#) ()
Returns the current clipping mode.
- void [clip_children](#) (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int [delete_child](#) (int n)
Removes the widget at `index` from the group and deletes it.
- void [end](#) ()
Exactly the same as `current(this->parent())`.
- int [find](#) (const [FL_Widget](#) &o) const
See `int FL_Group::find(const FL_Widget *w) const`.
- int [find](#) (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char *=0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void [focus](#) ([FL_Widget](#) *W)
- void [forms_end](#) ()
This is for forms compatibility only.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- void [init_sizes](#) ()
Resets the internal array of widget sizes and positions.
- void [insert](#) ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void [insert](#) ([FL_Widget](#) &o, [FL_Widget](#) *before)
This does `insert(w, find(before))`.
- void [remove](#) ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void [remove](#) ([FL_Widget](#) *o)
Removes the widget `o` from the group.
- void [remove](#) (int index)
Removes the widget at `index` from the group but does not delete it.
- [FL_Widget](#) * [resizable](#) () const
Returns the group's resizable widget.
- void [resizable](#) ([FL_Widget](#) &o)
Sets the group's resizable widget.
- void [resizable](#) ([FL_Widget](#) *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- void [resize](#) (int, int, int, int) [FL_OVERRIDE](#)
Resizes the [FL_Group](#) widget and all of its children.
- virtual [~FL_Group](#) ()
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const

- Returns whether the widget and all of its parents are active.*
- [FL_Align align](#) () const
Gets the label alignment.
- void [align](#) ([FL_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FL_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).
- virtual class [FL_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FL_Window](#) * [as_window](#) ()
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).
- virtual [FL_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FL_Boxtype box](#) () const
Gets the box type of the widget.
- void [box](#) ([FL_Boxtype](#) new_box)
Sets the box type for the widget.
- [FL_Callback_p callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FL_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FL_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.

- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FL_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FL_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FL_Color](#) bg, [FL_Color](#) sel)
Sets the background and selection color of the widget.
- [FL_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FL_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)

- Set the spacing between the label and the horizontal edge of the widget.*

 - `FI_Image * image ()`

Gets the image that is used as part of the widget label when in the active state.

- `const FI_Image * image () const`

Gets the image that is used as part of the widget label when in the active state.

- `void image (FI_Image &img)`

Sets the image to use as part of the widget label when in the active state.

- `void image (FI_Image *img)`

Sets the image to use as part of the widget label when in the active state.

- `int image_bound () const`

Returns whether the image is managed by the widget.

- `int inside (const FI_Widget *wgt) const`

Checks if this widget is a child of wgt.

- `int is_label_copied () const`

Returns whether the current label was assigned with `copy_label()`.

- `const char * label () const`

Gets the current label text.

- `void label (const char *text)`

Sets the current label pointer.

- `void label (FI_Labeltype a, const char *b)`

Shortcut to set the label text and type in one call.

- `int label_image_spacing ()`

Return the gap size between the label and the image.

- `void label_image_spacing (int gap)`

Set the gap between the label and the image in pixels.

- `FI_Color labelcolor () const`

Gets the label color.

- `void labelcolor (FI_Color c)`

Sets the label color.

- `FI_Font labelfont () const`

Gets the font to use.

- `void labelfont (FI_Font f)`

Sets the font to use.

- `FI_Fonsize labelsiz () const`

Gets the font size in pixels.

- `void labelsiz (FI_Fonsize pix)`

Sets the font size in pixels.

- `FI_Labeltype labeltype () const`

Gets the label type.

- `void labeltype (FI_Labeltype a)`

Sets the label type.

- `void measure_label (int &ww, int &hh) const`

Sets width ww and height hh accordingly with the label size.

- `bool needs_keyboard () const`

Returns whether this widget needs a keyboard.

- `void needs_keyboard (bool needs)`

Sets whether this widget needs a keyboard.

- `unsigned int output () const`

Returns if a widget is used for output only.

- `FI_Group * parent () const`

Returns a pointer to the parent widget.

- void [parent](#) ([FI_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [FI_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)

- Sets the user data for this widget.*

 - void **user_data** (void *v)

Sets the user data for this widget.
- int **vertical_label_margin** ()

Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const

Returns whether a widget is visible.
- unsigned int **visible_focus** () const

Checks whether this widget has a visible focus.
- void **visible_focus** (int v)

Modifies keyboard focus navigation.
- int **visible_r** () const

Returns whether a widget and all its parents are visible.
- int **w** () const

Gets the widget width.
- **Fl_When when** () const

Returns the conditions under which the callback is called.
- void **when** (uchar i)

Sets the flags used to decide when a callback is called.
- **Fl_Window * window** () const

Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const

Gets the widget position in its window.
- int **y** () const

Gets the widget position in its window.
- virtual **~Fl_Widget** ()

Destroys the widget.

Additional Inherited Members

Public Types inherited from **Fl_Browser**

- enum **Fl_Line_Position** { **TOP** , **BOTTOM** , **MIDDLE** }
- For internal use only?*

Public Types inherited from **Fl_Browser_**

- enum {
HORIZONTAL = 1 , **VERTICAL** = 2 , **BOTH** = 3 , **ALWAYS_ON** = 4 ,
HORIZONTAL_ALWAYS = 5 , **VERTICAL_ALWAYS** = 6 , **BOTH_ALWAYS** = 7 }
*Values for **has_scrollbar()**.*

Static Public Member Functions inherited from **Fl_Group**

- static **Fl_Group * current** ()
- Returns the currently active group.*
- static void **current** (**Fl_Group** *g)
- Sets the current group.*

Static Public Member Functions inherited from FI_Widget

- static void `default_callback` (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Public Attributes inherited from FI_Browser_

- FI_Scrollbar `hscrollbar`
Horizontal scrollbar.
- FI_Scrollbar `scrollbar`
Vertical scrollbar.

Protected Types inherited from FI_Widget

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from FI_Browser

- FL_BLINE * `_remove` (int line)
Removes the item at the specified line.
- FL_BLINE * `find_line` (int line) const
Returns the item for specified line.
- int `full_height` () const `FL_OVERRIDE`
The height of the entire list of all visible() items in pixels.
- int `incr_height` () const `FL_OVERRIDE`
The default 'average' item height (including inter-item spacing) in pixels.
- void `insert` (int line, FL_BLINE *item)
Insert specified item above line.
- void * `item_at` (int line) const `FL_OVERRIDE`
Return the item at specified line.
- void `item_draw` (void *item, int X, int Y, int W, int H) const `FL_OVERRIDE`
Draws item at the position specified by X Y W H.
- void * `item_first` () const `FL_OVERRIDE`
Returns the very first item in the list.
- int `item_height` (void *item) const `FL_OVERRIDE`
Returns height of item in pixels.

- void * [item_last](#) () const [FL_OVERRIDE](#)
Returns the very last item in the list.
- void * [item_next](#) (void *item) const [FL_OVERRIDE](#)
*Returns the next item after *item*.*
- void * [item_prev](#) (void *item) const [FL_OVERRIDE](#)
*Returns the previous item before *item*.*
- void [item_select](#) (void *item, int val) [FL_OVERRIDE](#)
*Change the selection state of *item* to the value *val*.*
- int [item_selected](#) (void *item) const [FL_OVERRIDE](#)
*See if *item* is selected.*
- void [item_swap](#) (void *a, void *b) [FL_OVERRIDE](#)
*Swap the items *a* and *b*.*
- const char * [item_text](#) (void *item) const [FL_OVERRIDE](#)
*Returns the label text for *item*.*
- int [item_width](#) (void *item) const [FL_OVERRIDE](#)
*Returns width of *item* in pixels.*
- int [lineno](#) (void *item) const
*Returns line number corresponding to *item*, or zero if not found.*
- void [swap](#) (FL_BLINE *a, FL_BLINE *b)
*Swap the two items *a* and *b*.*

Protected Member Functions inherited from [FI_Browser_](#)

- void [bbox](#) (int &X, int &Y, int &W, int &H) const
Returns the bounding box for the interior of the list's display window, inside the scrollbars.
- void [deleting](#) (void *item)
*This method should be used when *item* is being deleted from the list.*
- int [displayed](#) (void *item) const
*Returns non-zero if *item* has been scrolled to a position where it is being displayed.*
- void [draw](#) () [FL_OVERRIDE](#)
Draws the list within the normal widget bounding box.
- void * [find_item](#) (int ypos)
*This method returns the item under mouse *y* position *ypos*.*
- [FI_Browser_](#) (int X, int Y, int W, int H, const char *L=0)
The constructor makes an empty browser.
- virtual int [full_width](#) () const
This method may be provided by the subclass to indicate the full width of the item list, in pixels.
- void [inserting](#) (void *a, void *b)
This method should be used when an item is in the process of being inserted into the list.
- virtual int [item_quick_height](#) (void *item) const
*This method may be provided by the subclass to return the height of the *item*, in pixels.*
- int [leftedge](#) () const
*This method returns the *X* position of the left edge of the list area after adjusting for the scrollbar and border, if any.*
- void [new_list](#) ()
This method should be called when the list data is completely replaced or cleared.
- void [redraw_line](#) (void *item)
*This method should be called when the contents of *item* has changed, but not its height.*
- void [redraw_lines](#) ()
This method will cause the entire list to be redrawn.
- void [replacing](#) (void *a, void *b)
*This method should be used when item *a* is being replaced by item *b*.*

- void * **selection** () const
Returns the item currently selected, or NULL if there is no selection.
- void **swapping** (void *a, void *b)
*This method should be used when two items *a* and *b* are being swapped.*
- void * **top** () const
Returns the item that appears at the top of the list.

Protected Member Functions inherited from **FL_Group**

- **FL_Rect** * **bounds** ()
Returns the internal array of widget sizes and positions.
- void **draw** () **FL_OVERRIDE**
Draws the widget.
- void **draw_child** (**FL_Widget** &widget) const
Forces a child to redraw.
- void **draw_children** ()
Draws all children of the group.
- void **draw_outside_label** (const **FL_Widget** &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int **on_insert** (**FL_Widget** *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int **on_move** (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void **on_remove** (int)
Allow derived groups to act when a child widget is removed from the group.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** (**FL_Widget** &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from **FL_Widget**

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
*If **FL_ALIGN_IMAGE_BACKDROP** is set, the image or deimage will be drawn.*
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (**FL_Boxtype** t, **FL_Color** c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (**FL_Boxtype** t, int x, int y, int w, int h, **FL_Color** c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (**FL_Boxtype** t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (**FL_Boxtype** t, int x, int y, int w, int h, **FL_Color** bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const

- *Draws the label in an arbitrary bounding box.*
- `Fl_Widget` (int *x*, int *y*, int *w*, int *h*, const char **label*=0L)
 - *Creates a widget at the given position and size.*
- unsigned int **flags** () const
 - *Gets the widget flags mask.*
- void *h* (int *v*)
 - *Internal use only.*
- void **set_flag** (unsigned int *c*)
 - *Sets a flag in the flags mask.*
- void *w* (int *v*)
 - *Internal use only.*
- void *x* (int *v*)
 - *Internal use only.*
- void *y* (int *v*)
 - *Internal use only.*

11.88.1 Detailed Description

The `Fl_Multi_Browser` class is a subclass of `Fl_Browser` which lets the user select any set of the lines.



Figure 11.31 `Fl_Multi_Browser`

The user interface is Macintosh style: clicking an item turns off all the others and selects that one, dragging selects all the items the mouse moves over, and ctrl + click (Cmd+click on the Mac OS platform) toggles the items. Shift + click extends the selection until the clicked item. This is different from how forms did it. Normally the callback is done when the user releases the mouse, but you can change this with `when()`. See `Fl_Browser` for methods to add and remove lines from the browser.

11.88.2 Constructor & Destructor Documentation

11.88.2.1 `Fl_Multi_Browser()`

```
Fl_Multi_Browser::Fl_Multi_Browser (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new `Fl_Multi_Browser` widget using the given position, size, and label string.

The default boxtype is `FL_DOWN_BOX`. The constructor specializes `Fl_Browser()` by setting the type to `FL_MULTIBROWSER`. The destructor destroys the widget and frees all memory that has been allocated.

The documentation for this class was generated from the following files:

- `Fl_Multi_Browser.H`
- `Fl_Browser.cxx`

11.89 `Fl_Multi_Label` Struct Reference

Allows a mixed text and/or graphics label to be applied to an `Fl_Menu_Item` or `Fl_Widget`.

```
#include <Fl_Multi_Label.H>
```


Public Member Functions

- void `label (FI_Menu_Item *)`
Associate an `FI_Multi_Label` with an `FI_Menu_Item`.
- void `label (FI_Widget *)`
Associate an `FI_Multi_Label` with an `FI_Widget`.

Public Attributes

- const char * `labela`
Holds the "leftmost" of the two elements in the composite label.
- const char * `labelb`
Holds the "rightmost" of the two elements in the composite label.
- uchar `typea`
Holds the "type" of `labela`.
- uchar `typeb`
Holds the "type" of `labelb`.

11.89.1 Detailed Description

Allows a mixed text and/or graphics label to be applied to an `FI_Menu_Item` or `FI_Widget`.

Most regular FLTK widgets now support the ability to associate both images and text with a label but some special cases, notably the non-widget `FI_Menu_Item` objects, do not. `FI_Multi_Label` may be used to create menu items that have an icon and text, which would not normally be possible for an `FI_Menu_Item`. For example, `FI_Multi_Label` is used in the New->Code submenu in fluid, and others.

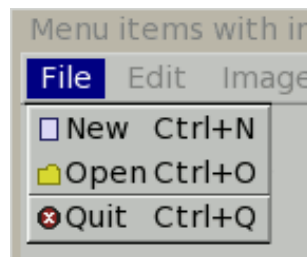


Figure 11.32 Menu items with icons using `FI_Multi_Label`

Each `FI_Multi_Label` holds two elements, `labela` and `labelb`; each may hold either a text label (const char*) or an image (FI_Image*). When displayed, `labela` is drawn first and `labelb` is drawn immediately to its right. More complex labels can be constructed by setting `labelb` as another `FI_Multi_Label` and thus chaining up a series of label elements.

When assigning a label element to one of `labela` or `labelb`, they should be explicitly cast to (const char*) if they are not of that type already.

Note

An `FI_Multi_Label` object and all its components (label text, images, chained `FI_Multi_Label`'s and their linked objects) must exist during the lifetime of the widget or menu item they are assigned to. It is the responsibility of the user's code to release these linked objects if necessary after the widget or menu is deleted.

Example Use: `FI_Menu_Bar`

```
FI_Pixmap *image = new FI_Pixmap(..); // image for menu item; any FI_Image based widget
FI_Menu_Bar *menu = new FI_Menu_Bar(..); // can be any FI_Menu_oriented widget (FI_Choice,
    FI_Menu_Button..)

// Create a menu item
int i = menu->add("File/New", ..);
FI_Menu_Item *item = (FI_Menu_Item*)&(menu->menu()[i]);

// Create a multi label, assign it an image + text
FI_Multi_Label *ml = new FI_Multi_Label;

// Left side of label is an image
```

```

ml->typea = FL_IMAGE_LABEL;
ml->labela = (const char*)image;           // any Fl_Image widget: Fl_Pixmap, Fl_PNG_Image, etc..

// Right side of label is label text
ml->typeb = FL_NORMAL_LABEL;
ml->labelb = item->label();

// Assign the multilabel to the menu item
// ml->label(item); // deprecated since 1.4.0; backwards compatible with 1.3.x
item->label(ml);    // new method since 1.4.0

```

See also

[Fl_Label](#) and [Fl_Labeltype](#) and examples/howto-menu-with-images.cxx

11.89.2 Member Function Documentation

11.89.2.1 label() [1/2]

```

void Fl_Multi_Label::label (
    Fl_Menu_Item * o)

```

Associate an [Fl_Multi_Label](#) with an [Fl_Menu_Item](#).

This uses [Fl_Menu_Item::label\(Fl_Labeltype a, const char *b\)](#) internally to set the *label* and the label type of the menu item, i.e. it stores a *pointer* to the [Fl_Multi_Label](#) object (*this*). An existing label (pointer) will be overwritten.

This sets the type of the menu item's label to `_FL_MULTI_LABEL` - note the leading underscore ('_').

There is no way to use a method like [Fl_Widget::copy_label\(\)](#) that transfers ownership of the [Fl_Multi_Label](#) and its linked objects (images, text, and chained [Fl_Multi_Label](#)'s) to the menu item.

The [Fl_Multi_Label](#) and all linked images, text labels, or chained [Fl_Multi_Label](#) objects must exist during the lifetime of the menu and will not be released when the menu item is destroyed.

Note

The user's code is responsible for releasing the [Fl_Multi_Label](#) and all linked objects (images, text, chained [Fl_Multi_Label](#)'s) after the menu has been deleted. This may cause memory leaks if [Fl_Multi_Label](#) is used and reassigned w/o releasing the objects assigned to it.

Deprecated since 1.4.0: please use [Fl_Menu_Item::label\(Fl_Multi_Label *\)](#)

See also

[Fl_Menu_Item::label\(Fl_Multi_Label *\)](#)

11.89.2.2 label() [2/2]

```

void Fl_Multi_Label::label (
    Fl_Widget * o)

```

Associate an [Fl_Multi_Label](#) with an [Fl_Widget](#).

This method uses [Fl_Widget::label\(Fl_Labeltype, const char *\)](#) internally to set the *label* of the widget, i.e. it stores a *pointer* to the [Fl_Multi_Label](#) object (*this*). An existing label that has been set using [Fl_Widget::copy_label\(\)](#) will be released prior to the assignment of the new label.

This sets the type of the widget's label to `_FL_MULTI_LABEL` - note the leading underscore ('_').

There is no way to use a method like [Fl_Widget::copy_label\(\)](#) that transfers ownership of the [Fl_Multi_Label](#) and its linked objects (images, text, and chained [Fl_Multi_Label](#)'s) to the widget.

The [Fl_Multi_Label](#) and all linked images, text labels, or chained [Fl_Multi_Label](#) objects must exist during the lifetime of the widget and will not be released when the widget is destroyed.

Note

The user's code is responsible for releasing the [Fl_Multi_Label](#) and all linked objects (images, text, chained [Fl_Multi_Label](#)'s) after the widget has been deleted. This may cause memory leaks if [Fl_Multi_Label](#) is used and reassigned w/o releasing the objects assigned to it.

11.89.3 Member Data Documentation

11.89.3.1 labela

```
const char* Fl_Multi_Label::labela
```

Holds the "leftmost" of the two elements in the composite label.

Typically this would be assigned either a text string (const char*), a (Fl_Image*) or a (Fl_Multi_Label*).

11.89.3.2 labelb

```
const char* Fl_Multi_Label::labelb
```

Holds the "rightmost" of the two elements in the composite label.

Typically this would be assigned either a text string (const char*), a (Fl_Image*) or a (Fl_Multi_Label*).

11.89.3.3 typea

```
uchar Fl_Multi_Label::typea
```

Holds the "type" of labela.

Typically this is set to FL_NORMAL_LABEL for a text label, FL_IMAGE_LABEL for an image (based on Fl_image) or FL_MULTI_LABEL if "chaining" multiple [Fl_Multi_Label](#) elements together.

11.89.3.4 typeb

```
uchar Fl_Multi_Label::typeb
```

Holds the "type" of labelb.

Typically this is set to FL_NORMAL_LABEL for a text label, FL_IMAGE_LABEL for an image (based on Fl_image) or FL_MULTI_LABEL if "chaining" multiple [Fl_Multi_Label](#) elements together.

The documentation for this struct was generated from the following files:

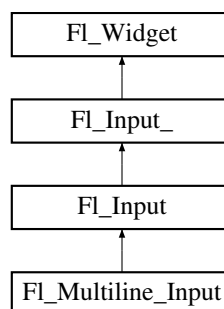
- [Fl_Multi_Label.H](#)
- [Fl_Multi_Label.cxx](#)

11.90 FI_Multiline_Input Class Reference

This input field displays '\n' characters as new lines rather than ^J, and accepts the Return, Tab, and up and down arrow keys.

```
#include <Fl_Multiline_Input.H>
```

Inheritance diagram for Fl_Multiline_Input:



Public Member Functions

- [Fl_Multiline_Input](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [Fl_Multiline_Input](#) widget using the given position, size, and label string.

Public Member Functions inherited from [Fl_Input](#)

- [Fl_Input](#) (int, int, int, int, const char *l=0)

Creates a new [FI_Input](#) widget using the given position, size, and label string.

- int [handle](#) (int) [FL_OVERRIDE](#)

Handles the specified event.

Public Member Functions inherited from [FI_Input_](#)

- int [append](#) (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool [can_redo](#) () const
Check if there is a redo action available.
- bool [can_undo](#) () const
Check if the last operation can be undone.
- int [copy](#) (int clipboard)
Put the current selection into the clipboard.
- int [copy_cuts](#) ()
Copies the yank buffer to the clipboard.
- [FI_Color](#) [cursor_color](#) () const
Gets the color of the cursor.
- void [cursor_color](#) ([FI_Color](#) n)
Sets the color of the cursor.
- int [cut](#) ()
Deletes the current selection.
- int [cut](#) (int a, int b)
*Deletes all characters between index *a* and *b*.*
- int [cut](#) (int n)
*Deletes the next *n* bytes rounded to characters before or after the cursor.*
- double [dvalue](#) () const
Returns the widget text interpreted as a floating point number.
- [FI_Input_](#) (int, int, int, int, const char *s=0)
Creates a new [FI_Input_](#) widget.
- unsigned int [index](#) (int i) const
*Returns the character at index *i*.*
- int [input_type](#) () const
Gets the input field type.
- void [input_type](#) (int t)
Sets the input field type.
- int [insert](#) (const char *t, int l=0)
Inserts text at the cursor position.
- int [insert_position](#) () const
Gets the position of the text cursor.
- int [insert_position](#) (int p)
Sets the cursor position and mark.
- int [insert_position](#) (int p, int m)
Sets the index for the cursor and mark.
- int [ivalue](#) () const
Returns the widget text interpreted as a signed integer.
- int [mark](#) () const
Gets the current selection mark.
- int [mark](#) (int m)
Sets the current selection mark.
- int [maximum_size](#) () const

- Gets the maximum length of the input field in characters.*
- void [maximum_size](#) (int m)
- Sets the maximum length of the input field in characters.*
- int [position](#) () const
- int [position](#) (int p)
- int [position](#) (int p, int m)
- int [readonly](#) () const
- Gets the read-only state of the input field.*
- void [readonly](#) (int b)
- Sets the read-only state of the input field.*
- int [redo](#) ()
- Redo previous undo operation.*
- int [replace](#) (int b, int e, const char *text, int ilen=0)
- Deletes text from b to e and inserts the new string text.*
- void [resize](#) (int, int, int, int) [FL_OVERRIDE](#)
- Changes the size of the widget.*
- int [shortcut](#) () const
- Return the shortcut key associated with this widget.*
- void [shortcut](#) (int s)
- Sets the shortcut key associated with this widget.*
- int [size](#) () const
- Returns the number of bytes in value().*
- void [size](#) (int W, int H)
- Sets the width and height of this widget.*
- int [static_value](#) (const char *)
- Changes the widget text.*
- int [static_value](#) (const char *, int)
- Changes the widget text.*
- int [tab_nav](#) () const
- Gets whether the Tab key causes focus navigation in multiline input fields or not.*
- void [tab_nav](#) (int val)
- Sets whether the Tab key does focus navigation, or inserts tab characters into FL_Multiline_Input.*
- [FL_Color](#) [textcolor](#) () const
- Gets the color of the text in the input field.*
- void [textcolor](#) ([FL_Color](#) n)
- Sets the color of the text in the input field.*
- [FL_Font](#) [textfont](#) () const
- Gets the font of the text in the input field.*
- void [textfont](#) ([FL_Font](#) s)
- Sets the font of the text in the input field.*
- [FL_Fontsize](#) [textsize](#) () const
- Gets the size of the text in the input field.*
- void [textsize](#) ([FL_Fontsize](#) s)
- Sets the size of the text in the input field.*
- int [undo](#) ()
- Undoes previous changes to the text buffer.*
- const char * [value](#) () const
- Returns the text displayed in the widget.*
- int [value](#) (const char *)
- Changes the widget text.*
- int [value](#) (const char *, int)

- *Changes the widget text.*
- int [value](#) (double value)
Changes the widget text to a floating point number ("%g").
- int [value](#) (int value)
Changes the widget text to a signed integer number.
- int [wrap](#) () const
Gets the word wrapping state of the input field.
- void [wrap](#) (int b)
Sets the word wrapping state of the input field.
- [~FI_Input_](#) ()
Destroys the widget.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align alignment](#))
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image *img](#))
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image *img](#))
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype new_box](#))
Sets the box type for the widget.

- [FI_Callback_p callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- uchar [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) (uchar c)
Sets the damage bits for the widget.
- void [damage](#) (uchar c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()

- Deactivates the widget.*

 - `FL_Image * deimage ()`
Gets the image that is used as part of the widget label when in the inactive state.
 - `const FL_Image * deimage () const`
Gets the image that is used as part of the widget label when in the inactive state.
 - `void deimage (FL_Image &img)`
Sets the image to use as part of the widget label when in the inactive state.
 - `void deimage (FL_Image *img)`
Sets the image to use as part of the widget label when in the inactive state.
 - `int deimage_bound () const`
Returns whether the inactive image is managed by the widget.
 - `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with default arguments.
 - `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
 - `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
 - `void draw_label (int, int, int, int, FL_Align) const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
 - `int h () const`
Gets the widget height.
 - `virtual void hide ()`
Makes a widget invisible.
 - `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
 - `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
 - `FL_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
 - `const FL_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
 - `void image (FL_Image &img)`
Sets the image to use as part of the widget label when in the active state.
 - `void image (FL_Image *img)`
Sets the image to use as part of the widget label when in the active state.
 - `int image_bound () const`
Returns whether the image is managed by the widget.
 - `int inside (const FL_Widget *wgt) const`
Checks if this widget is a child of wgt.
 - `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
 - `const char * label () const`
Gets the current label text.
 - `void label (const char *text)`
Sets the current label pointer.
 - `void label (FL_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
 - `int label_image_spacing ()`
Return the gap size between the label and the image.
 - `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.

- [FL_Color labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize labelsiz](#)e () const
Gets the font size in pixels.
- void [labelsiz](#)e ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group * parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [FL_Color selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - virtual void **show** ()
- Makes a widget visible.*

 - void **size** (int W, int H)
- Changes the size of the widget.*

 - int **take_focus** ()
- Gives the widget the keyboard focus.*

 - unsigned int **takeevents** () const
- Returns if the widget is able to take events.*

 - int **test_shortcut** ()
- Returns true if the widget's label contains the entered '&x' shortcut.*

 - const char * **tooltip** () const
- Gets the current tooltip text.*

 - void **tooltip** (const char *text)
- Sets the current tooltip text.*

 - **Fl_Window** * **top_window** () const
- Returns a pointer to the top-level window for the widget.*

 - **Fl_Window** * **top_window_offset** (int &xoff, int &yoff) const
- Finds the x/y offset of the current widget relative to the top-level window.*

 - **uchar** **type** () const
- Gets the widget type.*

 - void **type** (**uchar** t)
- Sets the widget type.*

 - int **use_accents_menu** ()
- Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*

 - void * **user_data** () const
- Gets the user data for this widget.*

 - void **user_data** (**Fl_Callback_User_Data** *v, bool auto_free)
- Sets the user data for this widget.*

 - void **user_data** (void *v)
- Sets the user data for this widget.*

 - int **vertical_label_margin** ()
- Get the spacing between the label and the vertical edge of the widget.*

 - void **vertical_label_margin** (int px)
- Set the spacing between the label and the vertical edge of the widget.*

 - unsigned int **visible** () const
- Returns whether a widget is visible.*

 - unsigned int **visible_focus** () const
- Checks whether this widget has a visible focus.*

 - void **visible_focus** (int v)
- Modifies keyboard focus navigation.*

 - int **visible_r** () const
- Returns whether a widget and all its parents are visible.*

 - int **w** () const
- Gets the widget width.*

 - **Fl_When** **when** () const
- Returns the conditions under which the callback is called.*

 - void **when** (**uchar** i)
- Sets the flags used to decide when a callback is called.*

 - **Fl_Window** * **window** () const
- Returns a pointer to the nearest parent window up the widget hierarchy.*

- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual `~FI_Widget` ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Static Public Attributes inherited from `FI_Input`

- static const char * `copy_menu_text` = "Copy"
[this text may be customized at run-time]
- static const char * `cut_menu_text` = "Cut"
[this text may be customized at run-time]
- static const char * `paste_menu_text` = "Paste"
[this text may be customized at run-time]

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from `FI_Input`

- void `draw` () `FL_OVERRIDE`
Draws the widget.
- int `handle_key` ()
Handles a keystroke.
- int `handle_rmb` ()
Handle right mouse button down events.

Protected Member Functions inherited from [FI_Input_](#)

- int [apply_undo](#) ()
Apply the current undo/redo operation.
- void [drawtext](#) (int, int, int, int)
Draws the text in the passed bounding box.
- void [drawtext](#) (int, int, int, int, bool draw_active)
Draws the text in the passed bounding box.
- void [handle_mouse](#) (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int [handletext](#) (int e, int, int, int, int)
Handles all kinds of text field related events.
- int [line_end](#) (int i) const
Finds the end of a line.
- int [line_start](#) (int i) const
Finds the start of a line.
- int [linesPerPage](#) ()
- void [maybe_do_callback](#) ([FI_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- int [up_down_position](#) (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int [word_end](#) (int i) const
Finds the end of a word.
- int [word_start](#) (int i) const
Finds the start of a word.
- int [xscroll](#) () const
- int [yscroll](#) () const
- void [yscroll](#) (int yOffset)

Protected Member Functions inherited from [FI_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.
- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.

- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.90.1 Detailed Description

This input field displays '\n' characters as new lines rather than ^J, and accepts the Return, Tab, and up and down arrow keys.

This is for editing multiline text.

This is far from the nirvana of text editors, and is probably only good for small bits of text, 10 lines at most. Note that this widget does not support scrollbars or per-character color control.

If you are presenting large amounts of text and need scrollbars or full color control of characters, you probably want [Fl_Text_Editor](#) instead.

In FLTK 1.3.x, the default behavior of the 'Tab' key was changed to support consistent focus navigation. To get the older FLTK 1.1.x behavior, set [Fl_Input::tab_nav\(\)](#) to 0. Newer programs should consider using [Fl_Text_Editor](#).

11.90.2 Constructor & Destructor Documentation

11.90.2.1 `Fl_Multiline_Input()`

```
Fl_Multiline_Input::Fl_Multiline_Input (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Multiline_Input](#) widget using the given position, size, and label string.

The default boxtype is `FL_DOWN_BOX`.

Inherited destructor destroys the widget and any value associated with it.

The documentation for this class was generated from the following files:

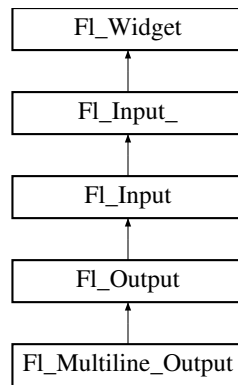
- `Fl_Multiline_Input.H`
- `Fl_Input.cxx`

11.91 `Fl_Multiline_Output` Class Reference

This widget is a subclass of [Fl_Output](#) that displays multiple lines of text.

```
#include <Fl_Multiline_Output.H>
```

Inheritance diagram for `Fl_Multiline_Output`:



Public Member Functions

- [`FI\_Multiline\_Output`](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [`FI\_Multiline\_Output`](#) widget using the given position, size, and label string.

Public Member Functions inherited from [`FI\_Output`](#)

- [`FI\_Output`](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [`FI\_Output`](#) widget using the given position, size, and label string.

Public Member Functions inherited from [`FI\_Input`](#)

- [`FI\_Input`](#) (int, int, int, int, const char *l=0)
Creates a new [`FI\_Input`](#) widget using the given position, size, and label string.
- int [`handle`](#) (int) [`FL\_OVERRIDE`](#)
Handles the specified event.

Public Member Functions inherited from [`FI\_Input\_`](#)

- int [`append`](#) (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool [`can\_redo`](#) () const
Check if there is a redo action available.
- bool [`can\_undo`](#) () const
Check if the last operation can be undone.
- int [`copy`](#) (int clipboard)
Put the current selection into the clipboard.
- int [`copy\_cuts`](#) ()
Copies the yank buffer to the clipboard.
- [`FI\_Color`](#) [`cursor\_color`](#) () const
Gets the color of the cursor.
- void [`cursor\_color`](#) ([`FI\_Color`](#) n)
Sets the color of the cursor.
- int [`cut`](#) ()
Deletes the current selection.
- int [`cut`](#) (int a, int b)
Deletes all characters between index *a* and *b*.
- int [`cut`](#) (int n)
Deletes the next *n* bytes rounded to characters before or after the cursor.
- double [`dvalue`](#) () const

- Returns the widget text interpreted as a floating point number.*

 - `FL_Input_` (int, int, int, int, const char *=0)

Creates a new `FL_Input_` widget.
- unsigned int `index` (int i) const

Returns the character at index `i`.
- int `input_type` () const

Gets the input field type.
- void `input_type` (int t)

Sets the input field type.
- int `insert` (const char *t, int l=0)

Inserts text at the cursor position.
- int `insert_position` () const

Gets the position of the text cursor.
- int `insert_position` (int p)

Sets the cursor position and mark.
- int `insert_position` (int p, int m)

Sets the index for the cursor and mark.
- int `ivalue` () const

Returns the widget text interpreted as a signed integer.
- int `mark` () const

Gets the current selection mark.
- int `mark` (int m)

Sets the current selection mark.
- int `maximum_size` () const

Gets the maximum length of the input field in characters.
- void `maximum_size` (int m)

Sets the maximum length of the input field in characters.
- int `position` () const
- int `position` (int p)
- int `position` (int p, int m)
- int `readonly` () const

Gets the read-only state of the input field.
- void `readonly` (int b)

Sets the read-only state of the input field.
- int `redo` ()

Redo previous undo operation.
- int `replace` (int b, int e, const char *text, int ilen=0)

Deletes text from `b` to `e` and inserts the new string `text`.
- void `resize` (int, int, int, int) `FL_OVERRIDE`

Changes the size of the widget.
- int `shortcut` () const

Return the shortcut key associated with this widget.
- void `shortcut` (int s)

Sets the shortcut key associated with this widget.
- int `size` () const

Returns the number of bytes in `value()`.
- void `size` (int W, int H)

Sets the width and height of this widget.
- int `static_value` (const char *)

Changes the widget text.
- int `static_value` (const char *, int)

- Changes the widget text.*

 - int [tab_nav](#) () const

Gets whether the Tab key causes focus navigation in multiline input fields or not.
- void [tab_nav](#) (int val)

Sets whether the Tab key does focus navigation, or inserts tab characters into [FI_Multiline_Input](#).
- [FI_Color](#) [textcolor](#) () const

Gets the color of the text in the input field.
- void [textcolor](#) ([FI_Color](#) n)

Sets the color of the text in the input field.
- [FI_Font](#) [textfont](#) () const

Gets the font of the text in the input field.
- void [textfont](#) ([FI_Font](#) s)

Sets the font of the text in the input field.
- [FI_Fontsize](#) [textsize](#) () const

Gets the size of the text in the input field.
- void [textsize](#) ([FI_Fontsize](#) s)

Sets the size of the text in the input field.
- int [undo](#) ()

Undoes previous changes to the text buffer.
- const char * [value](#) () const

Returns the text displayed in the widget.
- int [value](#) (const char *)

Changes the widget text.
- int [value](#) (const char *, int)

Changes the widget text.
- int [value](#) (double value)

Changes the widget text to a floating point number ("%g").
- int [value](#) (int value)

Changes the widget text to a signed integer number.
- int [wrap](#) () const

Gets the word wrapping state of the input field.
- void [wrap](#) (int b)

Sets the word wrapping state of the input field.
- ~[FI_Input](#) ()

Destroys the widget.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
 - void [_set_fullscreen](#) ()
 - void [activate](#) ()
- Activates the widget.*
- unsigned int [active](#) () const
- Returns whether the widget is active.*
- int [active_r](#) () const
- Returns whether the widget and all of its parents are active.*
- [FI_Align](#) [align](#) () const
- Gets the label alignment.*
- void [align](#) ([FI_Align](#) alignment)
- Sets the label alignment.*
- long [argument](#) () const

- Gets the current user data (long) argument that is passed to the callback function.*

 - void [argument](#) (long v)
- Sets the current user data (long) argument that is passed to the callback function.*

 - virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()

Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).

 - virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).*

 - virtual [FI_Group](#) * [as_group](#) ()

Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).

 - virtual [FI_Group](#) const * [as_group](#) () const
- Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*

 - virtual [FI_Window](#) * [as_window](#) ()

Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).

 - virtual [FI_Window](#) const * [as_window](#) () const
- Sets the image to use as part of the widget label when in the inactive state.*

 - void [bind_deimage](#) ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.

 - void [bind_deimage](#) (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.

 - void [bind_image](#) ([FI_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.

 - void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- Gets the box type of the widget.*

 - [FI_Boxtype](#) [box](#) () const

Gets the box type of the widget.

 - void [box](#) ([FI_Boxtype](#) new_box)

Sets the box type for the widget.
- Gets the current callback function for the widget.*

 - [FI_Callback_p](#) [callback](#) () const

Gets the current callback function for the widget.

 - void [callback](#) ([FI_Callback](#) *cb)

Sets the current callback function for the widget.

 - void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.

 - void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.

 - void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.

 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.
- Checks if the widget value changed since the last callback.*

 - unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.
- Marks the widget as inactive without sending events or changing focus.*

 - void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.
- Marks the value of the widget as unchanged.*

 - void [clear_changed](#) ()

Marks the value of the widget as unchanged.
- Clears or sets the damage flags.*

 - void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.
- Sets a widget to accept input.*

 - void [clear_output](#) ()

Sets a widget to accept input.
- Hides the widget.*

 - void [clear_visible](#) ()

Hides the widget.
- Disables keyboard focus navigation with this widget.*

 - void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.
- Disables keyboard focus navigation with this widget.*

 - [FI_Color](#) [color](#) () const

- Gets the background color of the widget.*

 - void `color` (`FL_Color` bg)
- Sets the background color of the widget.*

 - void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*

 - `FL_Color` `color2` () const
- For back compatibility only.*

 - void `color2` (unsigned a)
- For back compatibility only.*

 - int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*

 - void `copy_label` (const char *new_label)
- Sets the current label.*

 - void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*

 - `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*

 - void `damage` (`uchar` c)
- Sets the damage bits for the widget.*

 - void `damage` (`uchar` c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int `damage_resize` (int, int, int, int)
- Internal use only.*

 - void `deactivate` ()
- Deactivates the widget.*

 - `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*

 - const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*

 - void `deimage` (`FL_Image` &img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - void `deimage` (`FL_Image` *img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*

 - void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with default arguments.*

 - void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*

 - void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*

 - void `draw_label` (int, int, int, int, `FL_Align`) const
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*

 - int `h` () const
- Gets the widget height.*

 - virtual void `hide` ()
- Makes a widget invisible.*

 - int `horizontal_label_margin` ()
- Get the spacing between the label and the horizontal edge of the widget.*

 - void `horizontal_label_margin` (int px)
- Set the spacing between the label and the horizontal edge of the widget.*

- `FL_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FL_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FL_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FL_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FL_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FL_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FL_Color c)`
Sets the label color.
- `FL_Font labelfont () const`
Gets the font to use.
- `void labelfont (FL_Font f)`
Sets the font to use.
- `FL_Fonsize labelsiz () const`
Gets the font size in pixels.
- `void labelsiz (FL_Fonsize pix)`
Sets the font size in pixels.
- `FL_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (FL_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`
Sets width ww and height hh accordingly with the label size.
- `bool needs_keyboard () const`
Returns whether this widget needs a keyboard.
- `void needs_keyboard (bool needs)`
Sets whether this widget needs a keyboard.
- `unsigned int output () const`
Returns if a widget is used for output only.
- `FL_Group * parent () const`
Returns a pointer to the parent widget.
- `void parent (FL_Group *p)`

- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
Repositions the window or widget.
 - void [redraw](#) ()
Schedules the drawing of the widget.
 - void [redraw_label](#) ()
Schedules the drawing of the label.
 - [Fl_Color](#) [selection_color](#) () const
Gets the selection color.
 - void [selection_color](#) ([Fl_Color](#) a)
Sets the selection color.
 - void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
 - void [set_changed](#) ()
Marks the value of the widget as changed.
 - void [set_output](#) ()
Sets a widget to output only.
 - void [set_visible](#) ()
Makes the widget visible.
 - void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
 - int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
 - void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
 - virtual void [show](#) ()
Makes a widget visible.
 - void [size](#) (int W, int H)
Changes the size of the widget.
 - int [take_focus](#) ()
Gives the widget the keyboard focus.
 - unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
 - int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
 - const char * [tooltip](#) () const
Gets the current tooltip text.
 - void [tooltip](#) (const char *text)
Sets the current tooltip text.
 - [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
 - [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
 - [uchar](#) [type](#) () const
Gets the widget type.
 - void [type](#) ([uchar](#) t)
Sets the widget type.
 - int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
 - void * [user_data](#) () const
Gets the user data for this widget.

- void **user_data** ([FI_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [FI_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) (uchar i)
Sets the flags used to decide when a callback is called.
- [FI_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual [~FI_Widget](#) ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Widget](#)

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Static Public Attributes inherited from [FI_Input](#)

- static const char * [copy_menu_text](#) = "Copy"
[this text may be customized at run-time]
- static const char * [cut_menu_text](#) = "Cut"
[this text may be customized at run-time]
- static const char * [paste_menu_text](#) = "Paste"
[this text may be customized at run-time]

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from [FI_Input](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- int [handle_key](#) ()
Handles a keystroke.
- int [handle_rmb](#) ()
Handle right mouse button down events.

Protected Member Functions inherited from [FI_Input_](#)

- int [apply_undo](#) ()
Apply the current undo/redo operation.
- void [drawtext](#) (int, int, int, int)
Draws the text in the passed bounding box.
- void [drawtext](#) (int, int, int, int, bool draw_active)
Draws the text in the passed bounding box.
- void [handle_mouse](#) (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int [handletext](#) (int e, int, int, int, int)
Handles all kinds of text field related events.
- int [line_end](#) (int i) const
Finds the end of a line.
- int [line_start](#) (int i) const
Finds the start of a line.
- int [linesPerPage](#) ()
- void [maybe_do_callback](#) ([FI_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- int [up_down_position](#) (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int [word_end](#) (int i) const
Finds the end of a word.
- int [word_start](#) (int i) const
Finds the start of a word.
- int [xscroll](#) () const
- int [yscroll](#) () const
- void [yscroll](#) (int yOffset)

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **FI_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.91.1 Detailed Description

This widget is a subclass of [FI_Output](#) that displays multiple lines of text.

It also displays tab characters as whitespace to the next column.

Note that this widget does not support scrollbars, or per-character color control.

If you are presenting large amounts of read-only text and need scrollbars, or full color control of characters, then use [FI_Text_Display](#). If you want to display HTML text, use [FI_Help_View](#).

A caret cursor (^) shows the keyboard navigation mark for keyboard selection of the output text, e.g. Arrow Keys to move the cursor, Shift + Arrow Keys to create a text selection, and '^C' to copy the selected text to the paste buffer. The caret cursor can be disabled by disabling the widget's "visible focus" using [clear_visible_focus\(\)](#), inherited from the [FI_Widget](#) base class. Doing this also disables the widget's keyboard navigation.

11.91.2 Constructor & Destructor Documentation

11.91.2.1 Fl_Multiline_Output()

```
Fl_Multiline_Output::Fl_Multiline_Output (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Multiline_Output](#) widget using the given position, size, and label string.

The default boxtype is `FL_DOWN_BOX`.

Inherited destructor destroys the widget and any value associated with it.

The documentation for this class was generated from the following files:

- `Fl_Multiline_Output.H`
- `Fl_Input.cxx`

11.92 Fl_Native_File_Chooser Class Reference

This class lets an FLTK application easily and consistently access the operating system's native file chooser.

```
#include <Fl_Native_File_Chooser.H>
```

Public Types

- enum [Option](#) {
[NO_OPTIONS](#) = 0x0000 , [SAVEAS_CONFIRM](#) = 0x0001 , [NEW_FOLDER](#) = 0x0002 , [PREVIEW](#) = 0x0004 ,
[USE_FILTER_EXT](#) = 0x0008 }
- enum [Type](#) {
[BROWSE_FILE](#) = 0 , [BROWSE_DIRECTORY](#) , [BROWSE_MULTI_FILE](#) , [BROWSE_MULTI_DIRECTORY](#) ,
[BROWSE_SAVE_FILE](#) , [BROWSE_SAVE_DIRECTORY](#) }

Public Member Functions

- int [count](#) () const
Returns the number of filenames (or directory names) the user selected.
- const char * [directory](#) () const
Returns the current preset [directory\(\)](#) value.
- void [directory](#) (const char *val)
Preset the directory the browser will show when opened.
- const char * [errmsg](#) () const
Returns a system dependent error message for the last method that failed.
- const char * [filename](#) () const
Return the filename the user chose.
- const char * [filename](#) (int i) const
Return one of the filenames the user selected.
- const char * [filter](#) () const
Returns the filter string last set.
- void [filter](#) (const char *f)
Sets the filename filters used for browsing.
- int [filter_value](#) () const
Returns which filter value was last selected by the user.
- void [filter_value](#) (int i)
Sets which filter will be initially selected.
- int [filters](#) () const
Gets how many filters were available, not including "All Files".

- [FL_Native_File_Chooser](#) (int val=[BROWSE_FILE](#))
The constructor.
- int **options** () const
Gets the platform specific [FL_Native_File_Chooser::Option](#) flags.
- void **options** (int o)
Sets the platform specific chooser options to val.
- const char * **preset_file** () const
Get the preset filename.
- void **preset_file** (const char *f)
Sets the default filename for the chooser.
- int **show** ()
Post the chooser's dialog.
- const char * **title** () const
Get the title of the file chooser's dialog window.
- void **title** (const char *t)
Set the title of the file chooser's dialog window.
- int **type** () const
Gets the current [FL_Native_File_Chooser::Type](#) of browser.
- void **type** (int t)
Sets the current [FL_Native_File_Chooser::Type](#) of browser.
- [~FL_Native_File_Chooser](#) ()
Destructor.

Static Public Attributes

- static const char * **file_exists_message** = "File exists. Are you sure you want to overwrite?"
Localizable message.

11.92.1 Detailed Description

This class lets an FLTK application easily and consistently access the operating system's native file chooser. Some operating systems have very complex and specific file choosers that many users want access to specifically, instead of FLTK's default file chooser(s).

In cases where there is no native file browser, FLTK's own file browser is used instead.

To use this widget, use the following include in your code:

```
#include <FL/Fl_Native_File_Chooser.H>
```

The following example shows how to pick a single file:

```
// Create and post the local native file chooser
#include <FL/Fl_Native_File_Chooser.H>
[...]
```

```
Fl_Native_File_Chooser fnfc;
fnfc.title("Pick a file");
fnfc.type(FL_Native_File_Chooser::BROWSE_FILE);
fnfc.filter("Text\t*.txt\n"
           "C Files\t*.{c,cc,h,cpp}");
fnfc.directory("/var/tmp"); // default directory to use
// Show native chooser
switch ( fnfc.show() ) {
    case -1: printf("ERROR: %s\n", fnfc.errmsg()); break; // ERROR
    case 1: printf("CANCEL\n"); break; // CANCEL
    default: printf("PICKED: %s\n", fnfc.filename()); break; // FILE CHOSEN
}
```

The [FL_Native_File_Chooser](#) widget transmits UTF-8 encoded filenames to its user. It is recommended to open files that may have non-ASCII names with the [fl_fopen\(\)](#) or [fl_open\(\)](#) utility functions that handle these names in a cross-platform way (whereas the standard [fopen\(\)](#)/[open\(\)](#) functions fail on the Windows platform to open files with a non-ASCII name).

Platform Specific Caveats

- Under X11/Wayland the dialog is chosen as follows:

1. If command `zenity` is available at run-time and if `Fl::option(OPTION_FNFC_USES_ZENITY)` is turned on, the `zenity`-based dialog opens. This is expected to be more appropriate than other dialog forms for sandboxed apps, but member function `filter_value()` is not effective.
 2. Else if the app runs under the KDE desktop and if `Fl::option(OPTION_FNFC_USES_KDIALOG)` is turned on, and if command `kdiallog` is available at run-time, the `kdiallog`-based dialog opens. Member function `filter_value()` is not effective with this dialog.
 3. Else if the GTK library is available at run-time on the computer and if `Fl::option(OPTION_FNFC_USES_GTK)` is not turned off, the GTK-styled dialog opens. Call `fl_register_images()` to add a "Preview" button to this dialog. Use the static public attributes of class `Fl_File_Chooser` to localize the browser.
 4. Otherwise, FLTK's own dialog produced by the `Fl_File_Chooser` class opens. Call `fl_register_images()` to add a "Preview" button to it. It's best if you also call `Fl_File_Icon::load_system_icons()` at the start of `main()`, to enable the nicer looking file browser widgets. Use the static public attributes of class `Fl_File_Chooser` to localize the browser.
- Some operating systems support certain OS specific options; see `Fl_Native_File_Chooser::options()` for a list.

Figure 11.33 The `Fl_Native_File_Chooser` on different platforms

11.92.2 Member Enumeration Documentation

11.92.2.1 Option

enum `Fl_Native_File_Chooser::Option`

Enumerator

NO_OPTIONS	no options enabled
SAVEAS_CONFIRM	Show native 'Save As' overwrite confirm dialog.

Enumerator

NEW_FOLDER	Show 'New Folder' icon (if supported)
PREVIEW	enable preview mode (if supported)
USE_FILTER_EXT	Chooser filter presets the output file extension (if supported)

11.92.2.2 Type

```
enum Fl_Native_File_Chooser::Type
```

Enumerator

BROWSE_FILE	browse files (lets user choose one file)
BROWSE_DIRECTORY	browse directories (lets user choose one directory)
BROWSE_MULTI_FILE	browse files (lets user choose multiple files)
BROWSE_MULTI_DIRECTORY	browse directories (lets user choose multiple directories)
BROWSE_SAVE_FILE	browse to save a file
BROWSE_SAVE_DIRECTORY	browse to save a directory

11.92.3 Constructor & Destructor Documentation

11.92.3.1 Fl_Native_File_Chooser()

```
Fl_Native_File_Chooser::Fl_Native_File_Chooser (
    int val = BROWSE_FILE)
```

The constructor.

Internally allocates the native widgets. Optional `val` presets the type of browser this will be, which can also be changed with `type()`.

11.92.3.2 ~Fl_Native_File_Chooser()

```
Fl_Native_File_Chooser::~Fl_Native_File_Chooser ()
```

Destructor.

Deallocates any resources allocated to this widget.

11.92.4 Member Function Documentation

11.92.4.1 count()

```
int Fl_Native_File_Chooser::count () const
```

Returns the number of filenames (or directory names) the user selected.

Example:

```
if ( fnfc->show() == 0 ) {
    // Print all filenames user selected
    for (int n=0; n<fnfc->count(); n++) {
        printf("%d) '%s'\n", n, fnfc->filename(n));
    }
}
```

11.92.4.2 directory()

```
void Fl_Native_File_Chooser::directory (
    const char * val)
```

Preset the directory the browser will show when opened.

If `val` is NULL, or no directory is specified, the chooser will attempt to use the last non-cancelled folder.

11.92.4.3 errmsg()

```
const char * Fl_Native_File_Chooser::errmsg () const
```

Returns a system dependent error message for the last method that failed.

This message should at least be flagged to the user in a dialog box, or to some kind of error log. Contents will be valid only for methods that document [errmsg\(\)](#) will have info on failures.

11.92.4.4 filename() [1/2]

```
const char * Fl_Native_File_Chooser::filename () const
```

Return the filename the user chose.

Use this if only expecting a single filename. If more than one filename is expected, use [filename\(int\)](#) instead. Return value may be "" if no filename was chosen (eg. user cancelled).

11.92.4.5 filename() [2/2]

```
const char * Fl_Native_File_Chooser::filename (
    int i) const
```

Return one of the filenames the user selected.

Use [count\(\)](#) to determine how many filenames the user selected.

Example:

```
if ( fnfc->show() == 0 ) {
    // Print all filenames user selected
    for (int n=0; n<fnfc->count(); n++ ) {
        printf("%d) '%s'\n", n, fnfc->filename(n));
    }
}
```

11.92.4.6 filter() [1/2]

```
const char * Fl_Native_File_Chooser::filter () const
```

Returns the filter string last set.

Can be NULL if no filter was set.

11.92.4.7 filter() [2/2]

```
void Fl_Native_File_Chooser::filter (
    const char * f)
```

Sets the filename filters used for browsing.

The default is NULL, which browses all files.

The filter string can be any of:

- A single wildcard (eg. "*.txt")
- Multiple wildcards (eg. ".*{cxx,h,H}")
- A descriptive name followed by a "\t" and a wildcard (eg. "Text Files\t*.txt")
- A list of separate wildcards with a "\n" between each (eg. ".*{cxx,H}\n*.txt")
- A list of descriptive names and wildcards (eg. "C++ Files\t*.{cxx,H}\nTxt Files\t*.txt")

The format of each filter is a wildcard, or an optional user description followed by '\t' and the wildcard.

On most platforms, each filter is available to the user via a pulldown menu in the file chooser. The 'All Files' option is always available to the user.

11.92.4.8 filter_value() [1/2]

```
int Fl_Native_File_Chooser::filter_value () const
```

Returns which filter value was last selected by the user.

This is only valid if the chooser returns success and if the particular file chooser supports it. Otherwise the value is not changed.

Some "native" file choosers don't support returning the filter selection by the user, particularly the system dialog based browsers:

- `kdialog` (KDE system dialog)
- `zenity` (another system dialog based chooser).

Note: this list may not be complete.

These system file chooser dialogs don't return the filter value chosen by the user.

See also

[filter\(const char *f\)](#)

11.92.4.9 filter_value() [2/2]

```
void Fl_Native_File_Chooser::filter_value (
    int i)
```

Sets which filter will be initially selected.

The first filter is indexed as 0. If [filter_value\(\)](#) == [filters\(\)](#), then "All Files" was chosen. If [filter_value\(\)](#) > [filters\(\)](#), then a custom filter was set.

Some "native" file choosers don't support this way to set the initial filter type, particularly the system dialog based browsers:

- `kdialog` (KDE system dialog)
- `zenity` (another system dialog based chooser).

Note: this list may not be complete.

As far as we know these dialogs use the **first** item in the list of filter values as the initial filter presented to the user.

See also

[filter\(const char *f\)](#)

11.92.4.10 options()

```
void Fl_Native_File_Chooser::options (
    int o)
```

Sets the platform specific chooser options to `val`.

`val` is expected to be one or more [Fl_Native_File_Chooser::Option](#) flags ORed together. Some platforms have OS-specific functions that can be enabled/disabled via this method.

Flag	Description	Win	Mac	Other
-----	-----	-----	-----	-----
NEW_FOLDER	Shows the 'New Folder' button.	Ignored	Used	Used
PREVIEW	Enables the 'Preview' mode by default .	Ignored	Ignored	Used
SAVEAS_CONFIRM	Confirm dialog if BROWSE_SAVE_FILE file exists.	Used	Used	Used
USE_FILTER_EXT	Chooser filter presets the output file extension.	Ignored	Used	Used (GTK)

11.92.4.11 preset_file()

```
void Fl_Native_File_Chooser::preset_file (
    const char * f)
```

Sets the default filename for the chooser.

Use [directory\(\)](#) to set the default directory. Mainly used to preset the filename for save dialogs, and on most platforms can be used for opening files as well.

11.92.4.12 show()

```
int Fl_Native_File_Chooser::show ()
```

Post the chooser's dialog.

Blocks until dialog has been completed or cancelled.

Returns

- 0 – user picked a file
- 1 – user cancelled
- -1 – failed; [errmsg\(\)](#) has reason

11.92.4.13 title() [1/2]

```
const char * Fl_Native_File_Chooser::title () const
```

Get the title of the file chooser's dialog window.

Return value may be NULL if no title was set.

11.92.4.14 title() [2/2]

```
void Fl_Native_File_Chooser::title (
    const char * t)
```

Set the title of the file chooser's dialog window.

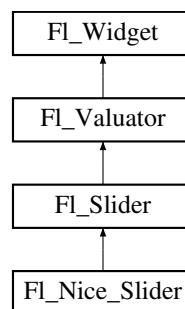
Can be NULL if no title desired. The default title varies according to the platform, so you are advised to set the title explicitly.

The documentation for this class was generated from the following files:

- [Fl_Native_File_Chooser.H](#)
- [Fl_Native_File_Chooser.cxx](#)
- [Fl_Native_File_Chooser_GTK.cxx](#)

11.93 FI_Nice_Slider Class Reference

Inheritance diagram for `Fl_Nice_Slider`:

**Public Member Functions**

- **Fl_Nice_Slider** (int X, int Y, int W, int H, const char *L=0)

Public Member Functions inherited from [Fl_Slider](#)

- void [bounds](#) (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- [Fl_Slider](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Slider](#) widget using the given position, size, and label string.
- **Fl_Slider** (uchar t, int X, int Y, int W, int H, const char *L)
Creates a new [Fl_Slider](#) widget using the given type, position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- int [scrollvalue](#) (int pos, int [size](#), int first, int total)
Sets the size and position of the sliding knob in the box.
- [Fl_Boxtype slider](#) () const
Gets the slider box type.
- void [slider](#) ([Fl_Boxtype](#) c)
Sets the slider box type.
- float [slider_size](#) () const
Get the dimensions of the moving piece of slider.

- void `slider_size` (double v)
Set the dimensions of the moving piece of slider.

Public Member Functions inherited from FI_Valuator

- void `bounds` (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double `clamp` (double)
Clamps the passed value to the valuator range.
- virtual int `format` (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double `increment` (double, int)
Adds n times the step value to the passed value.
- double `maximum` () const
Gets the maximum value for the valuator.
- void `maximum` (double a)
Sets the maximum value for the valuator.
- double `minimum` () const
Gets the minimum value for the valuator.
- void `minimum` (double a)
Sets the minimum value for the valuator.
- void `precision` (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void `range` (double a, double b)
Sets the minimum and maximum values for the valuator.
- double `round` (double)
Round the passed value to the nearest step increment.
- double `step` () const
Gets or sets the step value.
- void `step` (double a, int b)
See double FI_Valuator::step() const
- void `step` (double s)
See double FI_Valuator::step() const.
- void `step` (int a)
See double FI_Valuator::step() const
- double `value` () const
Gets the floating point(double) value.
- int `value` (double)
Sets the current value.
- `~FI_Valuator` () `FL_OVERRIDE`
Destructor is accessible despite protected constructor.

Public Member Functions inherited from FI_Widget

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
Activates the widget.
- unsigned int `active` () const
Returns whether the widget is active.

- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [Fl_Align](#) align () const
Gets the label alignment.
- void [align](#) ([Fl_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()
Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [Fl_Group](#) * [as_group](#) ()
Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).
- virtual [Fl_Group](#) const * [as_group](#) () const
- virtual [Fl_Window](#) * [as_window](#) ()
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- virtual [Fl_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype](#) box () const
Gets the box type of the widget.
- void [box](#) ([Fl_Boxtype](#) new_box)
Sets the box type for the widget.
- [Fl_Callback_p](#) callback () const
Gets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([Fl_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.

- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FL_Color` bg)
Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)
Sets the background and selection color of the widget.
- `FL_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FL_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FL_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const

- Gets the widget height.*

 - virtual void `hide` ()
- Makes a widget invisible.*

 - int `horizontal_label_margin` ()

Get the spacing between the label and the horizontal edge of the widget.

 - void `horizontal_label_margin` (int px)

Set the spacing between the label and the horizontal edge of the widget.

 - `FL_Image` * `image` ()

Gets the image that is used as part of the widget label when in the active state.

 - const `FL_Image` * `image` () const

Gets the image that is used as part of the widget label when in the active state.

 - void `image` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the active state.

 - void `image` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the active state.

 - int `image_bound` () const

Returns whether the image is managed by the widget.

 - int `inside` (const `FL_Widget` *wgt) const

Checks if this widget is a child of wgt.

 - int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.

 - const char * `label` () const

Gets the current label text.

 - void `label` (const char *text)

Sets the current label pointer.

 - void `label` (`FL_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.

 - int `label_image_spacing` ()

Return the gap size between the label and the image.

 - void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.

 - `FL_Color` `labelcolor` () const

Gets the label color.

 - void `labelcolor` (`FL_Color` c)

Sets the label color.

 - `FL_Font` `labelfont` () const

Gets the font to use.

 - void `labelfont` (`FL_Font` f)

Sets the font to use.

 - `FL_Fonsize` `labelsize` () const

Gets the font size in pixels.

 - void `labelsize` (`FL_Fonsize` pix)

Sets the font size in pixels.

 - `FL_Labeltype` `labeltype` () const

Gets the label type.

 - void `labeltype` (`FL_Labeltype` a)

Sets the label type.

 - void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.

 - bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.

- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FI_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FI_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)
Changes the size or position of the widget.
- [FI_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const

- Finds the x/y offset of the current widget relative to the top-level window.*
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (FI_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `FI_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `FI_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~FI_Widget ()`
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- `static void default_callback (FI_Widget *widget, void *data)`
The default callback for all widgets that don't set a callback.
- `static unsigned int label_shortcut (const char *t)`
Returns the Unicode value of the '&x' shortcut in a given text.
- `static int test_shortcut (const char *, const bool require_alt=false)`
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
 ,
OVERRIDE = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
 ,
MENU_WINDOW = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
 ,
GROUP_RELATIVE = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
 = 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from FI_Slider

- void **draw** () **FL_OVERRIDE**
Draws the widget.
- void **draw** (int, int, int, int)
- int **handle** (int, int, int, int, int)

Protected Member Functions inherited from FI_Valuator

- FI_Valuator** (int X, int Y, int W, int H, const char *L)
Creates a new FI_Valuator widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void **value_damage** ()
Asks for partial redraw.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.

- void **draw_box** ([Fl_Boxtype](#) t, [Fl_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([Fl_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [Fl_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

The documentation for this class was generated from the following files:

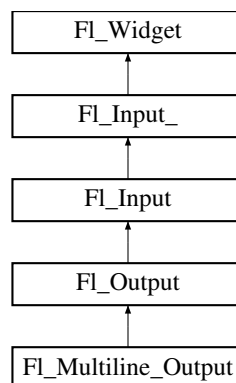
- [Fl_Nice_Slider.H](#)
- [Fl_Slider.cxx](#)

11.94 Fl_Output Class Reference

This widget displays a piece of text.

```
#include <Fl_Output.H>
```

Inheritance diagram for Fl_Output:



Public Member Functions

- [FI_Output](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [FI_Output](#) widget using the given position, size, and label string.

Public Member Functions inherited from [FI_Input](#)

- [FI_Input](#) (int, int, int, int, const char *l=0)
Creates a new [FI_Input](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [FI_Input_](#)

- int [append](#) (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool [can_redo](#) () const
Check if there is a redo action available.
- bool [can_undo](#) () const
Check if the last operation can be undone.
- int [copy](#) (int clipboard)
Put the current selection into the clipboard.
- int [copy_cuts](#) ()
Copies the yank buffer to the clipboard.
- [FI_Color](#) [cursor_color](#) () const
Gets the color of the cursor.
- void [cursor_color](#) ([FI_Color](#) n)
Sets the color of the cursor.
- int [cut](#) ()
Deletes the current selection.
- int [cut](#) (int a, int b)
*Deletes all characters between index *a* and *b*.*
- int [cut](#) (int n)
*Deletes the next *n* bytes rounded to characters before or after the cursor.*
- double [dvalue](#) () const
Returns the widget text interpreted as a floating point number.
- [FI_Input_](#) (int, int, int, int, const char *l=0)
Creates a new [FI_Input_](#) widget.
- unsigned int [index](#) (int i) const
*Returns the character at index *i*.*
- int [input_type](#) () const
Gets the input field type.
- void [input_type](#) (int t)
Sets the input field type.
- int [insert](#) (const char *t, int l=0)
Inserts text at the cursor position.
- int [insert_position](#) () const
Gets the position of the text cursor.
- int [insert_position](#) (int p)
Sets the cursor position and mark.
- int [insert_position](#) (int p, int m)
Sets the index for the cursor and mark.

- `int ivalue () const`
Returns the widget text interpreted as a signed integer.
- `int mark () const`
Gets the current selection mark.
- `int mark (int m)`
Sets the current selection mark.
- `int maximum_size () const`
Gets the maximum length of the input field in characters.
- `void maximum_size (int m)`
Sets the maximum length of the input field in characters.
- `int position () const`
- `int position (int p)`
- `int position (int p, int m)`
- `int readonly () const`
Gets the read-only state of the input field.
- `void readonly (int b)`
Sets the read-only state of the input field.
- `int redo ()`
Redo previous undo operation.
- `int replace (int b, int e, const char *text, int ilen=0)`
*Deletes text from *b* to *e* and inserts the new string *text*.*
- `void resize (int, int, int, int) FL_OVERRIDE`
Changes the size of the widget.
- `int shortcut () const`
Return the shortcut key associated with this widget.
- `void shortcut (int s)`
Sets the shortcut key associated with this widget.
- `int size () const`
*Returns the number of bytes in *value()*.*
- `void size (int W, int H)`
Sets the width and height of this widget.
- `int static_value (const char *)`
Changes the widget text.
- `int static_value (const char *, int)`
Changes the widget text.
- `int tab_nav () const`
Gets whether the Tab key causes focus navigation in multiline input fields or not.
- `void tab_nav (int val)`
*Sets whether the Tab key does focus navigation, or inserts tab characters into *FL_Multiline_Input*.*
- `FL_Color textcolor () const`
Gets the color of the text in the input field.
- `void textcolor (FL_Color n)`
Sets the color of the text in the input field.
- `FL_Font textfont () const`
Gets the font of the text in the input field.
- `void textfont (FL_Font s)`
Sets the font of the text in the input field.
- `FL_Fontsize textsize () const`
Gets the size of the text in the input field.
- `void textsize (FL_Fontsize s)`
Sets the size of the text in the input field.

- int [undo](#) ()
Undoes previous changes to the text buffer.
- const char * [value](#) () const
Returns the text displayed in the widget.
- int [value](#) (const char *)
Changes the widget text.
- int [value](#) (const char *, int)
Changes the widget text.
- int [value](#) (double value)
Changes the widget text to a floating point number ("%g").
- int [value](#) (int value)
Changes the widget text to a signed integer number.
- int [wrap](#) () const
Gets the word wrapping state of the input field.
- void [wrap](#) (int b)
Sets the word wrapping state of the input field.
- ~[FI_Input](#) ()
Destroys the widget.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)

- Sets the image to use as part of the widget label when in the active state.*

 - void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const

Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)

Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const

Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()

Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.
- void [clear_output](#) ()

Sets a widget to accept input.
- void [clear_visible](#) ()

Hides the widget.
- void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const

Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)

Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)

Sets the background and selection color of the widget.
- [FI_Color](#) [color2](#) () const

For back compatibility only.
- void [color2](#) (unsigned a)

For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const

Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)

Sets the current label.
- void [copy_tooltip](#) (const char *text)

Sets the current tooltip text.
- uchar [damage](#) () const

Returns non-zero if [draw\(\)](#) needs to be called.

- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)

- Sets the current label pointer.*

 - void `label` (`FI_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const

Gets the label color.
- void `labelcolor` (`FI_Color` c)

Sets the label color.
- `FI_Font` `labelfont` () const

Gets the font to use.
- void `labelfont` (`FI_Font` f)

Sets the font to use.
- `FI_Fonsize` `labelsize` () const

Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)

Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

Gets the label type.
- void `labeltype` (`FI_Labeltype` a)

Sets the label type.
- void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FI_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FI_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- `FI_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`FI_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.

- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const

- Gets the widget width.*
 - [Fl_When](#) [when](#) () const
 - Returns the conditions under which the callback is called.*
 - void [when](#) (uchar i)
 - Sets the flags used to decide when a callback is called.*
 - [Fl_Window](#) * [window](#) () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
 - int [x](#) () const
 - Gets the widget position in its window.*
 - int [y](#) () const
 - Gets the widget position in its window.*
 - virtual ~[Fl_Widget](#) ()
 - Destroys the widget.*

Additional Inherited Members

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)
 - The default callback for all widgets that don't set a callback.*
- static unsigned int [label_shortcut](#) (const char *t)
 - Returns the Unicode value of the '&x' shortcut in a given text.*
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
 - Returns true if the given text t contains the entered '&x' shortcut.*

Static Public Attributes inherited from [Fl_Input](#)

- static const char * [copy_menu_text](#) = "Copy"
 - [this text may be customized at run-time]*
- static const char * [cut_menu_text](#) = "Cut"
 - [this text may be customized at run-time]*
- static const char * [paste_menu_text](#) = "Paste"
 - [this text may be customized at run-time]*

Protected Types inherited from [Fl_Widget](#)

- enum {
 - [INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
 - [FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
 - ,
 - [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
 - ,
 - [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
 - ,
 - [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#) = 1<<19 ,
 - [NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
 - [AUTO_DELETE_USER_DATA](#) = 1<<23 ,
 - [MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
 - [USERFLAG1](#) = 1<<31 }
 - flags possible values enumeration.*

Protected Member Functions inherited from FI_Input

- void **draw** () **FL_OVERRIDE**
Draws the widget.
- int **handle_key** ()
Handles a keystroke.
- int **handle_rmb** ()
Handle right mouse button down events.

Protected Member Functions inherited from FI_Input_

- int **apply_undo** ()
Apply the current undo/redo operation.
- void **drawtext** (int, int, int, int)
Draws the text in the passed bounding box.
- void **drawtext** (int, int, int, int, bool draw_active)
Draws the text in the passed bounding box.
- void **handle_mouse** (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int **handletext** (int e, int, int, int, int)
Handles all kinds of text field related events.
- int **line_end** (int i) const
Finds the end of a line.
- int **line_start** (int i) const
Finds the start of a line.
- int **linesPerPage** ()
- void **maybe_do_callback** (FI_Callback_Reason reason=FL_REASON_UNKNOWN)
- int **up_down_position** (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int **word_end** (int i) const
Finds the end of a word.
- int **word_start** (int i) const
Finds the start of a word.
- int **xscroll** () const
- int **yscroll** () const
- void **yscroll** (int yOffset)

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const

- Draws a focus rectangle around the widget.*
- void `draw_focus` (`FI_Boxtype` t, int x, int y, int w, int h, `FI_Color` bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `FI_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

11.94.1 Detailed Description

This widget displays a piece of text.

When you set the `value()` , `FI_Output` does a `strcpy()` to its own storage, which is useful for program-generated values. The user may select portions of the text using the mouse and paste the contents into other fields or programs.

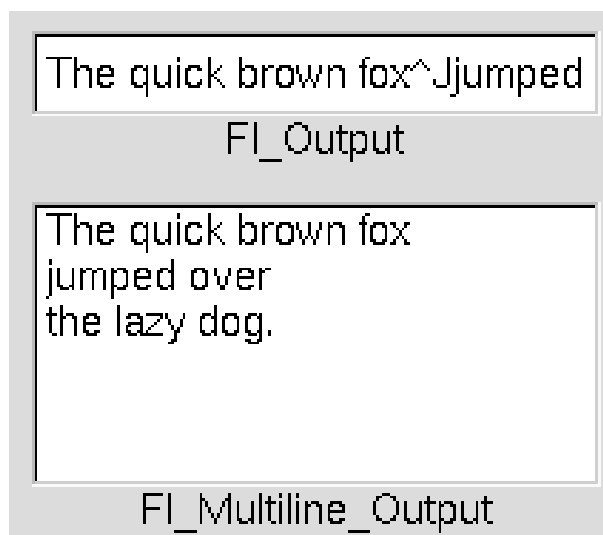


Figure 11.34 `FI_Output`

There is a single subclass, `FI_Multiline_Output`, which allows you to display multiple lines of text. `FI_Multiline_Output` does not provide scroll bars. If a more complete text editing widget is needed, use `FI_Text_Display` instead.

The text may contain any characters except `\0`, and will correctly display anything, using `^X` notation for unprintable control characters and `\nnn` notation for unprintable characters with the high bit set. It assumes the font can draw any characters in the ISO-Latin1 character set.

11.94.2 Constructor & Destructor Documentation

11.94.2.1 Fl_Output()

```
Fl_Output::Fl_Output (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Output](#) widget using the given position, size, and label string.

The default boxtype is `FL_DOWN_BOX`.

Inherited destructor destroys the widget and any value associated with it.

The documentation for this class was generated from the following files:

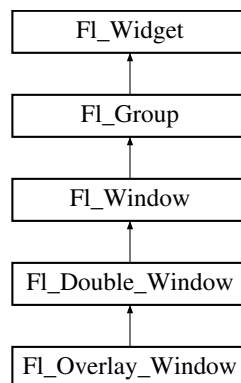
- `Fl_Output.H`
- `Fl_Input.cxx`

11.95 Fl_Overlay_Window Class Reference

This window provides double buffering and also the ability to draw the "overlay" which is another picture placed on top of the main image.

```
#include <Fl_Overlay_Window.H>
```

Inheritance diagram for `Fl_Overlay_Window`:



Public Member Functions

- `Fl_Overlay_Window * as_overlay_window () FL_OVERRIDE`
Return non-null if this is an `Fl_Overlay_Window` object.
- `int can_do_overlay ()`
Returns non-zero if there's hardware overlay support.
- `virtual void draw_overlay ()=0`
You must subclass `Fl_Overlay_Window` and provide this method.
- `void flush () FL_OVERRIDE`
Forces the window to be drawn, this window is also made current and calls `draw()`.
- `void hide () FL_OVERRIDE`
Makes a widget invisible.
- `void redraw_overlay ()`
Call this to indicate that the overlay data has changed and needs to be redrawn.
- `void resize (int, int, int, int) FL_OVERRIDE`
Changes the size or position of the widget.
- `void show () FL_OVERRIDE`
Makes a widget visible.

- void **show** (int a, char **b)
*Same as [FI_Window::show\(int a, char **b\)](#)*
- **~FI_Overlay_Window** ()
Destroys the window and all child widgets.

Public Member Functions inherited from [FI_Double_Window](#)

- [FI_Double_Window](#) * **as_double_window** () [FL_OVERRIDE](#)
Return non-null if this is an [FI_Double_Window](#) object.
- **FI_Double_Window** (int W, int H, const char *l=0)
Creates a new [FI_Double_Window](#) widget using the given position, size, and label (title) string.
- **FI_Double_Window** (int X, int Y, int W, int H, const char *l=0)
*See [FI_Double_Window::FI_Double_Window\(int w, int h, const char *label = 0\)](#)*
- void **show** (int a, char **b)
*Same as [FI_Window::show\(int a, char **b\)](#)*
- **~FI_Double_Window** ()
The destructor also deletes all the children.

Public Member Functions inherited from [FI_Window](#)

- void **allow_expand_outside_parent** ()
Allow this subwindow to expand outside the area of its parent window.
- [FI_Window](#) const * **as_window** () const [FL_OVERRIDE](#)
- [FI_Window](#) * **as_window** () [FL_OVERRIDE](#)
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- unsigned int **border** () const
Returns whether the window possesses a border.
- void **border** (int b)
Sets whether or not the window manager border is around the window.
- void **clear_border** ()
Fast inline function to turn the window manager border off.
- void **clear_modal_states** ()
Clears the "modal" flags and converts a "modal" or "non-modal" window back into a "normal" window.
- void **copy_label** (const char *a)
Sets the window titlebar label to a copy of a character string.
- void **cursor** (const [FI_RGB_Image](#) *, int, int)
Changes the cursor for this window using the provided image as cursor's shape.
- void **cursor** ([FI_Cursor](#) c, [FI_Color](#), [FI_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void **cursor** ([FI_Cursor](#))
Changes the cursor for this window.
- int **decorated_h** () const
Returns the window height including any window title bar and any frame added by the window manager.
- int **decorated_w** () const
Returns the window width including any frame added by the window manager.
- void **default_cursor** ([FI_Cursor](#) c, [FI_Color](#), [FI_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void **default_cursor** ([FI_Cursor](#))
Sets the default window cursor.
- void **draw_backdrop** ()
Draw the background image if one is set and is aligned inside.
- [FI_Window](#) (int w, int h, const char *title=0)

- Creates a window from the given width `w`, height `h`, and `title`.*

 - `FL_Window` (int `x`, int `y`, int `w`, int `h`, const char `*title=0`)

Creates a window from the given position (`x`, `y`), size (`w`, `h`) and `title`.
- void `free_position` ()

Undoes the effect of a previous `resize()` or `show()` so that the next time `show()` is called the window manager is free to position the window.
- void `fullscreen` ()

Makes the window completely fill one or more screens, without any window manager border visible.
- unsigned int `fullscreen_active` () const

Returns non zero if FULLSCREEN flag is set, 0 otherwise.
- void `fullscreen_off` ()

Turns off any side effects of `fullscreen()`
- void `fullscreen_off` (int `X`, int `Y`, int `W`, int `H`)

Turns off any side effects of `fullscreen()` and does `resize(x,y,w,h)`.
- void `fullscreen_screens` (int `top`, int `bottom`, int `left`, int `right`)

Sets which screens should be used when this window is in fullscreen mode.
- uchar `get_size_range` (int `*minw`, int `*minh`, int `*maxw=NULL`, int `*maxh=NULL`, int `*dw=NULL`, int `*dh=NULL`, int `*aspect=NULL`)

Gets the allowable range to which the user can resize this window.
- int `handle` (int) `FL_OVERRIDE`

Handles the specified event.
- void `hide` () `FL_OVERRIDE`

Removes the window from the screen.
- void `hotspot` (const `FL_Widget` &`p`, int `offscreen=0`)

See void `FL_Window::hotspot`(int `x`, int `y`, int `offscreen = 0`)
- void `hotspot` (const `FL_Widget` *, int `offscreen=0`)

See void `FL_Window::hotspot`(int `x`, int `y`, int `offscreen = 0`)
- void `hotspot` (int `x`, int `y`, int `offscreen=0`)

Positions the window so that the mouse is pointing at the given position, or at the center of the given widget, which may be the window itself.
- const void * `icon` () const

Gets the current icon window target dependent data.
- void `icon` (const `FL_RGB_Image` *)

Sets or resets a single window icon.
- void `icon` (const void *`ic`)

Platform-specific method to set the window icon usable on Windows and X11 only.
- void `iconize` ()

Iconifies the window.
- const char * `iconlabel` () const

See void `FL_Window::iconlabel`(const char)*
- void `iconlabel` (const char *)

Sets the icon label.
- void `icons` (const `FL_RGB_Image` *[], int)

Sets the window icons.
- void `icons` (HICON `big_icon`, HICON `small_icon`)

Sets the window icons using HICON handles (Windows platform only).
- const char * `label` () const

See void `FL_Window::label`(const char)*
- void `label` (const char *)

Sets the window title bar label.

- void **label** (const char *label, const char *iconlabel)
Sets the icon label.
- void **make_current** ()
Sets things up so that the drawing functions in <FL/fl_draw.H> will go into this window.
- void **maximize** ()
Maximizes a top-level window to its current screen.
- unsigned int **maximize_active** () const
Returns whether the window is currently maximized.
- unsigned int **menu_window** () const
Returns true if this window is a menu window.
- unsigned int **modal** () const
Returns true if this window is modal.
- unsigned int **non_modal** () const
Returns true if this window is modal or non-modal.
- **fl_uintptr_t os_id** ()
Returns a platform-specific identification of a shown window, or 0 if not shown.
- unsigned int **override** () const
Returns non zero if OVERRIDE flag is set, 0 otherwise.
- void **resize** (int X, int Y, int W, int H) **FL_OVERRIDE**
Changes the size and position of the window.
- int **screen_num** ()
The number of the screen containing the mapped window.
- void **screen_num** (int screen_num)
Set the number of the screen where to map the window.
- void **set_menu_window** ()
Marks the window as a menu window.
- void **set_modal** ()
*A "modal" window, when **shown()**, will prevent any events from being delivered to other windows in the same program, and will also remain on top of the other windows (if the X window manager supports the "transient for" property).*
- void **set_non_modal** ()
*A "non-modal" window (terminology borrowed from Microsoft Windows) acts like a **modal()** one in that it remains on top, but it has no effect on event delivery.*
- void **set_override** ()
Activates the flags NOBORDER|OVERRIDE.
- void **set_tooltip_window** ()
Marks the window as a tooltip window.
- const **Fl_Image** * **shape** ()
Returns the image controlling the window shape or NULL.
- void **shape** (const **Fl_Image** &b)
*Set the window's shape with an **Fl_Image**.*
- void **shape** (const **Fl_Image** *img)
Assigns a non-rectangular shape to the window.
- void **show** () **FL_OVERRIDE**
Puts the window on the screen.
- void **show** (int argc, char **argv)
*Puts the window on the screen with **show()** and parses command-line arguments.*
- int **shown** ()
*Returns non-zero if **show()** has been called (but not **hide()**).*
- void **size_range** (int minw, int minh, int maxw=0, int maxh=0, int dw=0, int dh=0, int aspect=0)
Sets the allowable range to which the user can resize this window.
- unsigned int **tooltip_window** () const

- Returns true if this window is a tooltip window.*

 - void [un_maximize](#) ()

Returns a previously maximized top-level window to its previous size.

- void [wait_for_expose](#) ()
- Waits for the window to be displayed after calling [show\(\)](#).*

 - int [x_root](#) () const

Gets the x position of the window on the screen.

- const char * [xclass](#) () const
- Returns the xclass for this window, or a default.*

 - void [xclass](#) (const char *c)

Sets the xclass for this window.

- int [y_root](#) () const
- Gets the y position of the window on the screen.*

 - virtual [~FL_Window](#) ()

The destructor also deletes all the children.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *& [_ddfdesign_kludge](#) ()
- This is for forms compatibility only.*

 - void [add](#) ([FL_Widget](#) &)

The widget is removed from its current group (if any) and then added to the end of this group.

- void [add](#) ([FL_Widget](#) *o)
- See void [FL_Group::add\(FL_Widget &w\)](#)*

 - void [add_resizable](#) ([FL_Widget](#) &o)

Adds a widget to the group and makes it the resizable widget.

- [FL_Widget](#) *const * [array](#) () const
- Returns a pointer to the internal array of children.*

 - [FL_Group](#) const * [as_group](#) () const [FL_OVERRIDE](#)
 - [FL_Group](#) * [as_group](#) () [FL_OVERRIDE](#)

Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).

- void [begin](#) ()
- Sets the current group so you can build the widget tree by just constructing the widgets.*

 - [FL_Widget](#) * [child](#) (int n) const

Returns the n'th child.

- int [children](#) () const
- Returns how many child widgets the group has.*

 - void [clear](#) ()

Deletes all child widgets from memory recursively.

- unsigned int [clip_children](#) ()
- Returns the current clipping mode.*

 - void [clip_children](#) (int c)

Controls whether the group widget clips the drawing of child widgets to its bounding box.

- virtual int [delete_child](#) (int n)
- Removes the widget at `index` from the group and deletes it.*

 - void [end](#) ()

Exactly the same as `current(this->parent())`.

- int [find](#) (const [FL_Widget](#) &o) const
- See int [FL_Group::find\(const FL_Widget *w\) const](#).*

 - int [find](#) (const [FL_Widget](#) *) const

Searches the children for the widget and returns the index.

- **FL_Group** (int, int, int, int, const char *=0)
Creates a new **FL_Group** widget using the given position, size, and label string.
- void **focus** (**FL_Widget** *W)
- void **forms_end** ()
This is for forms compatibility only.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** (**FL_Widget** &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** (**FL_Widget** &o, **FL_Widget** *before)
This does insert(w, find(before)).
- void **remove** (**FL_Widget** &)
Removes a widget from the group but does not delete it.
- void **remove** (**FL_Widget** *o)
Removes the widget o from the group.
- void **remove** (int index)
Removes the widget at index from the group but does not delete it.
- **FL_Widget** * **resizable** () const
Returns the group's resizable widget.
- void **resizable** (**FL_Widget** &o)
Sets the group's resizable widget.
- void **resizable** (**FL_Widget** *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual **~FL_Group** ()
The destructor also deletes all the children.

Public Member Functions inherited from **FL_Widget**

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FL_Align** **align** () const
Gets the label alignment.
- void **align** (**FL_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FL_Gl_Window** * **as_gl_window** ()
Returns an **FL_Gl_Window** pointer if this widget is an **FL_Gl_Window**.
- virtual class **FL_Gl_Window** const * **as_gl_window** () const
- void **bind_deimage** (**FL_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.

- void [bind_image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FL_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FL_Boxtype](#) new_box)
Sets the box type for the widget.
- [FL_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FL_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FL_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) ([uchar](#) c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FL_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FL_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FL_Color](#) bg, [FL_Color](#) sel)
Sets the background and selection color of the widget.
- [FL_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FL_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const

- Returns non-zero if [draw\(\)](#) needs to be called.*

 - void [damage](#) ([uchar](#) c)
 - Sets the damage bits for the widget.*
 - void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
 - Sets the damage bits for an area inside the widget.*
 - int [damage_resize](#) (int, int, int, int)
 - Internal use only.*
 - void [deactivate](#) ()
 - Deactivates the widget.*
 - [Fl_Image](#) * [deimage](#) ()
 - Gets the image that is used as part of the widget label when in the inactive state.*
 - const [Fl_Image](#) * [deimage](#) () const
 - Gets the image that is used as part of the widget label when in the inactive state.*
 - void [deimage](#) ([Fl_Image](#) &img)
 - Sets the image to use as part of the widget label when in the inactive state.*
 - void [deimage](#) ([Fl_Image](#) *img)
 - Sets the image to use as part of the widget label when in the inactive state.*
 - int [deimage_bound](#) () const
 - Returns whether the inactive image is managed by the widget.*
 - void [do_callback](#) ([Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
 - Calls the widget callback function with default arguments.*
 - void [do_callback](#) ([Fl_Widget](#) *widget, long arg, [Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
 - Calls the widget callback function with arbitrary arguments.*
 - void [do_callback](#) ([Fl_Widget](#) *widget, void *arg=0, [Fl_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
 - Calls the widget callback function with arbitrary arguments.*
 - void [draw_label](#) (int, int, int, int, [Fl_Align](#)) const
 - Draws the label in an arbitrary bounding box with an arbitrary alignment.*
 - int [h](#) () const
 - Gets the widget height.*
 - int [horizontal_label_margin](#) ()
 - Get the spacing between the label and the horizontal edge of the widget.*
 - void [horizontal_label_margin](#) (int px)
 - Set the spacing between the label and the horizontal edge of the widget.*
 - [Fl_Image](#) * [image](#) ()
 - Gets the image that is used as part of the widget label when in the active state.*
 - const [Fl_Image](#) * [image](#) () const
 - Gets the image that is used as part of the widget label when in the active state.*
 - void [image](#) ([Fl_Image](#) &img)
 - Sets the image to use as part of the widget label when in the active state.*
 - void [image](#) ([Fl_Image](#) *img)
 - Sets the image to use as part of the widget label when in the active state.*
 - int [image_bound](#) () const
 - Returns whether the image is managed by the widget.*
 - int [inside](#) (const [Fl_Widget](#) *wgt) const
 - Checks if this widget is a child of wgt.*
 - int [is_label_copied](#) () const
 - Returns whether the current label was assigned with [copy_label\(\)](#).*
 - const char * [label](#) () const
 - Gets the current label text.*
 - void [label](#) (const char *text)
 - Sets the current label pointer.*

- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [FL_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()

- Makes the widget visible.*

 - void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

Gets the current tooltip text.
- void [tooltip](#) (const char *text)

Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const

Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const

Gets the widget type.
- void [type](#) ([uchar](#) t)

Sets the widget type.
- int [use_accents_menu](#) ()

Returns non zero if `MAC_USE_ACCENTS_MENU` flag is set, 0 otherwise.
- void * [user_data](#) () const

Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)

Sets the user data for this widget.
- void [user_data](#) (void *v)

Sets the user data for this widget.
- int [vertical_label_margin](#) ()

Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const

Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const

Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)

Modifies keyboard focus navigation.
- int [visible_r](#) () const

Returns whether a widget and all its parents are visible.
- int [w](#) () const

Gets the widget width.
- [Fl_When](#) [when](#) () const

Returns the conditions under which the callback is called.

- void `when` (`uchar` i)
Sets the flags used to decide when a callback is called.
- `FL_Window * window` () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual `~FL_Widget` ()
Destroys the widget.

Protected Member Functions

- `FL_Overlay_Window` (int W, int H, const char *l=0)
See `FL_Overlay_Window::FL_Overlay_Window(int X, int Y, int W, int H, const char *l=0)`
- `FL_Overlay_Window` (int X, int Y, int W, int H, const char *l=0)
Creates a new `FL_Overlay_Window` widget using the given position, size, and label (title) string.

Protected Member Functions inherited from `FL_Window`

- void `default_size_range` ()
Protected method to calculate the default size range of a window.
- void `draw` () `FL_OVERRIDE`
Draws the widget.
- int `force_position` () const
Returns the internal state of the window's `FORCE_POSITION` flag.
- void `force_position` (int force)
Sets an internal flag that tells FLTK and the window manager to honor position requests.
- void `free_icons` ()
Deletes all icons previously attached to the window.
- int `is_resizable` ()
Protected method to determine whether a window is resizable.

Protected Member Functions inherited from `FL_Group`

- `FL_Rect * bounds` ()
Returns the internal array of widget sizes and positions.
- void `draw_child` (`FL_Widget` &widget) const
Forces a child to redraw.
- void `draw_children` ()
Draws all children of the group.
- void `draw_outside_label` (const `FL_Widget` &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int `on_insert` (`FL_Widget` *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int `on_move` (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void `on_remove` (int)
Allow derived groups to act when a child widget is removed from the group.
- int * `sizes` ()
Returns the internal array of widget sizes and positions.
- void `update_child` (`FL_Widget` &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Public Types inherited from [FI_Window](#)

- typedef struct HICON__ * **HICON**

Static Public Member Functions inherited from [FI_Window](#)

- static [FI_Window](#) * **current** ()
Returns the last window that was made current.
- static void **default_callback** ([FI_Window](#) *, void *v)
Back compatibility: Sets the default callback v for win to call on close event.
- static void **default_icon** (const [FI_RGB_Image](#) *)
Sets a single default window icon.
- static void **default_icons** (const [FI_RGB_Image](#) *[], int)
Sets the default window icons.

- static void [default_icons](#) (HICON big_icon, HICON small_icon)
Sets the default window icons (Windows platform only).
- static const char * [default_xclass](#) ()
Returns the default xclass.
- static void [default_xclass](#) (const char *)
Sets the default window xclass.
- static bool [is_a_rescale](#) ()
Returns true when a window is being rescaled.
- static char [show_next_window_iconic](#) ()
Returns the static flag whether the next window should be opened iconified.
- static void [show_next_window_iconic](#) (char stat)
Sets a static flag whether the next window should be opened iconified.

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * [current](#) ()
Returns the currently active group.
- static void [current](#) ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [FI_Widget](#)

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Static Protected Attributes inherited from [FI_Window](#)

- static [FI_Window](#) * [current_](#)
Stores the last window that was made current.

11.95.1 Detailed Description

This window provides double buffering and also the ability to draw the "overlay" which is another picture placed on top of the main image.

The overlay is designed to be a rapidly-changing but simple graphic such as a mouse selection box. [Fl_Overlay_Window](#) uses the overlay planes provided by your graphics hardware if they are available.

If no hardware support is found the overlay is simulated by drawing directly into the on-screen copy of the double-buffered window, and "erased" by copying the backbuffer over it again. This means the overlay will blink if you change the image in the window.

11.95.2 Constructor & Destructor Documentation

11.95.2.1 Fl_Overlay_Window()

```
Fl_Overlay_Window::Fl_Overlay_Window (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0) [protected]
```

Creates a new [Fl_Overlay_Window](#) widget using the given position, size, and label (title) string.

If the positions (x,y) are not given, then the window manager will choose them.

11.95.3 Member Function Documentation

11.95.3.1 as_overlay_window()

```
Fl_Overlay_Window * Fl_Overlay_Window::as_overlay_window () [inline], [virtual]
```

Return non-null if this is an [Fl_Overlay_Window](#) object.

Reimplemented from [Fl_Window](#).

11.95.3.2 draw_overlay()

```
virtual void Fl_Overlay_Window::draw_overlay () [pure virtual]
```

You must subclass [Fl_Overlay_Window](#) and provide this method.

It is just like a [draw\(\)](#) method, except it draws the overlay. The overlay will have already been "cleared" when this is called. You can use any of the routines described in [<FL/fl_draw.H>](#).

11.95.3.3 flush()

```
void Fl_Overlay_Window::flush () [virtual]
```

Forces the window to be drawn, this window is also made current and calls [draw\(\)](#).

Reimplemented from [Fl_Double_Window](#).

11.95.3.4 hide()

```
void Fl_Overlay_Window::hide () [virtual]
```

Makes a widget invisible.

See also

[show\(\)](#), [visible\(\)](#), [visible_r\(\)](#)

Reimplemented from [Fl_Double_Window](#).

11.95.3.5 redraw_overlay()

```
void Fl_Overlay_Window::redraw_overlay ()
```

Call this to indicate that the overlay data has changed and needs to be redrawn.

The overlay will be clear until the first time this is called, so if you want an initial display you must call this after calling [show\(\)](#).

11.95.3.6 `resize()`

```
void Fl_Overlay_Window::resize (
    int x,
    int y,
    int w,
    int h) [virtual]
```

Changes the size or position of the widget.

This is a virtual function so that the widget may implement its own handling of resizing. The default version does *not* call the `redraw()` method, but instead relies on the parent widget to do so because the parent may know a faster way to update the display, such as scrolling from the old position.

Some window managers under X11 call `resize()` a lot more often than needed. Please verify that the position or size of a widget did actually change before doing any extensive calculations.

`position(X, Y)` is a shortcut for `resize(X, Y, w(), h())`, and `size(W, H)` is a shortcut for `resize(x(), y(), W, H)`.

Parameters

in	<code>x,y</code>	new position relative to the parent window
in	<code>w,h</code>	new size

See also

[position\(int,int\)](#), [size\(int,int\)](#)

Reimplemented from [Fl_Double_Window](#).

11.95.3.7 `show()`

```
void Fl_Overlay_Window::show () [virtual]
```

Makes a widget visible.

An invisible widget never gets redrawn and does not get keyboard or mouse events, but can receive a few other events like `FL_SHOW`.

The `visible()` method returns true if the widget is set to be visible. The `visible_r()` method returns true if the widget and all of its parents are visible. A widget is only visible if `visible()` is true on it *and all of its parents*.

Changing it will send `FL_SHOW` or `FL_HIDE` events to the widget. *Do not change it if the parent is not visible, as this will send false `FL_SHOW` or `FL_HIDE` events to the widget.* `redraw()` is called if necessary on this or the parent.

See also

[hide\(\)](#), [visible\(\)](#), [visible_r\(\)](#)

Reimplemented from [Fl_Double_Window](#).

The documentation for this class was generated from the following files:

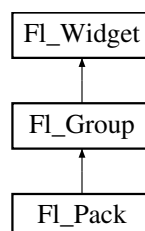
- `Fl_Overlay_Window.H`
- `Fl_Overlay_Window.cxx`

11.96 Fl_Pack Class Reference

This widget was designed to add the functionality of compressing and aligning widgets.

```
#include <Fl_Pack.H>
```

Inheritance diagram for `Fl_Pack`:



Public Types

- enum { **VERTICAL** = 0 , **HORIZONTAL** = 1 }

Public Member Functions

- void **clear** ()
Deletes all child widgets with [FL_Group::clear\(\)](#).
- [FL_Pack](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [FL_Pack](#) widget using the given position, size, and label string.
- [uchar horizontal](#) () const
Returns non-zero if [FL_Pack](#) alignment is horizontal.
- void **resize** (int X, int Y, int W, int H) [FL_OVERRIDE](#)
Override [FL_Group](#) resize behavior.
- int **spacing** () const
Gets the number of extra pixels of blank space that are added between the children.
- void **spacing** (int i)
Sets the number of extra pixels of blank space that are added between the children.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *& **_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void **add_resizable** ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * **array** () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * **as_group** () const [FL_OVERRIDE](#)
- [FL_Group](#) * **as_group** () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
*Removes the widget at *index* from the group and deletes it.*
- void **end** ()
*Exactly the same as *current(this->parent())*.*
- int **find** (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*

- int **find** (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char *=0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void **focus** ([FL_Widget](#) *W)
- void **forms_end** ()
This is for forms compatibility only.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** ([FL_Widget](#) &o, [FL_Widget](#) *before)
This does insert(w, find(before)).
- void **remove** ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void **remove** ([FL_Widget](#) *o)
Removes the widget o from the group.
- void **remove** (int index)
Removes the widget at index from the group but does not delete it.
- [FL_Widget](#) * **resizable** () const
Returns the group's resizable widget.
- void **resizable** ([FL_Widget](#) &o)
Sets the group's resizable widget.
- void **resizable** ([FL_Widget](#) *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual **~FL_Group** ()
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- [FL_Align](#) **align** () const
Gets the label alignment.
- void **align** ([FL_Align](#) alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FL_Gl_Window](#) * **as_gl_window** ()
Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).
- virtual class [FL_Gl_Window](#) const * **as_gl_window** () const

- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.

- int [contains](#) (const [FI_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [FI_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FI_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FI_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FI_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FI_Widget](#) *widget, long arg, [FI_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FI_Widget](#) *widget, void *arg=0, [FI_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FI_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FI_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FI_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FI_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const

- Returns whether the image is managed by the widget.*

 - int `inside` (const `FI_Widget` *wgt) const

Checks if this widget is a child of wgt.
- int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const

Gets the current label text.
- void `label` (const char *text)

Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const

Gets the label color.
- void `labelcolor` (`FI_Color` c)

Sets the label color.
- `FI_Font` `labelfont` () const

Gets the font to use.
- void `labelfont` (`FI_Font` f)

Sets the font to use.
- `FI_Fonsize` `labelsize` () const

Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)

Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

Gets the label type.
- void `labeltype` (`FI_Labeltype` a)

Sets the label type.
- void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FI_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FI_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- `FI_Color` `selection_color` () const

Gets the selection color.

- void [selection_color](#) ([FI_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const

- Returns whether a widget is visible.*
- unsigned int `visible_focus` () const
Checks whether this widget has a visible focus.
- void `visible_focus` (int v)
Modifies keyboard focus navigation.
- int `visible_r` () const
Returns whether a widget and all its parents are visible.
- int `w` () const
Gets the widget width.
- `FL_When` when () const
Returns the conditions under which the callback is called.
- void `when` (uchar i)
Sets the flags used to decide when a callback is called.
- `FL_Window` * `window` () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual `~FL_Widget` ()
Destroys the widget.

Protected Member Functions

- void `draw` () `FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from `FL_Group`

- `FL_Rect` * `bounds` ()
Returns the internal array of widget sizes and positions.
- void `draw_child` (`FL_Widget` &widget) const
Forces a child to redraw.
- void `draw_children` ()
Draws all children of the group.
- void `draw_outside_label` (const `FL_Widget` &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int `on_insert` (`FL_Widget` *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int `on_move` (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void `on_remove` (int)
Allow derived groups to act when a child widget is removed from the group.
- int * `sizes` ()
Returns the internal array of widget sizes and positions.
- void `update_child` (`FL_Widget` &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Group

- static FI_Group * **current** ()
Returns the currently active group.
- static void **current** (FI_Group *g)
Sets the current group.

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
 ,
[OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
 ,
[MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
 ,
[GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
 = 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.96.1 Detailed Description

This widget was designed to add the functionality of compressing and aligning widgets.

If [type\(\)](#) is [Fl_Pack::HORIZONTAL](#) all the children are resized to the height of the [Fl_Pack](#), and are moved next to each other horizontally. If [type\(\)](#) is not [Fl_Pack::HORIZONTAL](#) then the children are resized to the width and are stacked below each other. Then the [Fl_Pack](#) resizes itself to surround the child widgets.

You may want to put the [Fl_Pack](#) inside an [Fl_Scroll](#).

The '[resizable\(\)](#)' for [Fl_Pack](#) is set to NULL by default. Its behavior is slightly different than in a normal [Fl_Group](#) widget: only if the [resizable\(\)](#) widget is the last widget in the group it is extended to take the full available width or height, respectively, of the [Fl_Pack](#) group.

Note

You can nest [Fl_Pack](#) widgets or put them inside [Fl_Scroll](#) widgets or inside other group widgets but their behavior can sometimes be *"surprising"*. This is partly due to the fact that [Fl_Pack](#) widgets resize themselves during their [draw\(\)](#) operation, trying to react on their child widgets resizing themselves during **their** [draw\(\)](#) operations which can be confusing. If you want to achieve special resize behavior of nested group widgets it can sometimes be easier to derive your own specialized group widget than to try to make nested [Fl_Pack](#) widgets behave as expected.

See also

[Fl_Group::resizable\(\)](#)

11.96.2 Constructor & Destructor Documentation

11.96.2.1 [Fl_Pack\(\)](#)

```
Fl_Pack::Fl_Pack (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Pack](#) widget using the given position, size, and label string.

The default boxtype is [FL_NO_BOX](#).

The default [type\(\)](#) is [Fl_Pack::VERTICAL](#).

The destructor *also deletes all the children*. This allows a whole tree to be deleted at once, without having to keep a pointer to all the children in the user code. A kludge has been done so the [Fl_Pack](#) and all of its children can be automatic (local) variables, but you must declare the [Fl_Pack](#) *first*, so that it is destroyed last.

Parameters

in	<i>X,Y</i>	X and Y coordinates (position)
in	<i>W,H</i>	width and height, respectively
in	<i>L</i>	label (optional)

11.96.3 Member Function Documentation

11.96.3.1 clear()

```
void Fl_Pack::clear () [inline]
```

Deletes all child widgets with [Fl_Group::clear\(\)](#).

And sets to NULL the [resizable\(\)](#) widget.

11.96.3.2 draw()

```
void Fl_Pack::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Reimplemented from [Fl_Group](#).

11.96.3.3 horizontal()

```
uchar Fl_Pack::horizontal () const [inline]
```

Returns non-zero if [Fl_Pack](#) alignment is horizontal.

Returns

non-zero if [Fl_Pack](#) alignment is horizontal ([Fl_Pack::HORIZONTAL](#))

Note

Currently the return value is the same as [Fl_Group::type\(\)](#), but this may change in the future. Do not set any other values than the following with [Fl_Pack::type\(\)](#):

- [Fl_Pack::VERTICAL](#) (Default)
- [Fl_Pack::HORIZONTAL](#)

See class [Fl_Pack](#) documentation for details.

11.96.3.4 resize()

```
void Fl_Pack::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Override [Fl_Group](#) resize behavior.

Resizing an [Fl_Pack](#) will not resize any of its children, but trigger a redraw, which in turn recalculates the dimensions of all children.

Parameters

in	<i>X,Y,W,H</i>	new position and size of the Fl_Pack widget

Reimplemented from [Fl_Group](#).

The documentation for this class was generated from the following files:

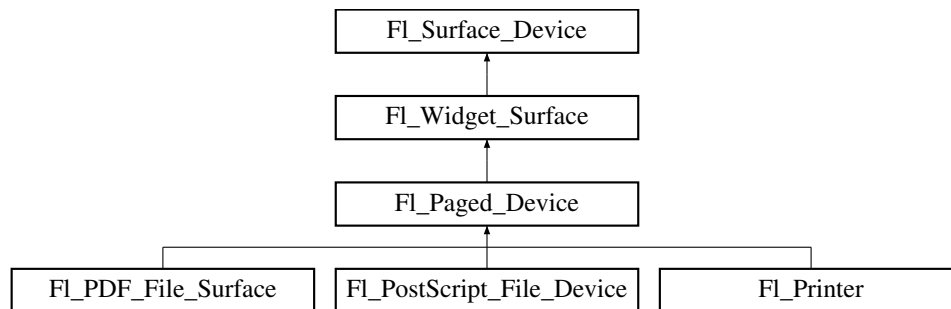
- [Fl_Pack.H](#)
- [Fl_Pack.cxx](#)

11.97 Fl_Paged_Device Class Reference

Represents page-structured drawing surfaces.

```
#include <Fl_Paged_Device.H>
```

Inheritance diagram for [Fl_Paged_Device](#):



Classes

- struct [page_format](#)
width, height and name of a page format

Public Types

- enum [Page_Format](#) {
[A0](#) = 0 , [A1](#) , [A2](#) , [A3](#) ,
[A4](#) , [A5](#) , [A6](#) , [A7](#) ,
[A8](#) , [A9](#) , [B0](#) , [B1](#) ,
[B2](#) , [B3](#) , [B4](#) , [B5](#) ,
[B6](#) , [B7](#) , [B8](#) , [B9](#) ,
[B10](#) , [C5E](#) , [DLE](#) , [EXECUTIVE](#) ,
[FOLIO](#) , [LEDGER](#) , [LEGAL](#) , [LETTER](#) ,
[TABLOID](#) , [ENVELOPE](#) , [MEDIA](#) = 0x1000 }
Possible page formats.
- enum [Page_Layout](#) { [PORTRAIT](#) = 0 , [LANDSCAPE](#) = 0x100 , [REVERSED](#) = 0x200 , [ORIENTATION](#) = 0x300
 }
Possible page layouts.

Public Member Functions

- virtual int [begin_job](#) (int pagecount=0, int *frompage=NULL, int *topage=NULL, char **perr_
 message=NULL)
Begins a print job.
- virtual int [begin_page](#) (void)
Begins a new printed page.
- virtual void [end_job](#) (void)
To be called at the end of a print job.
- virtual int [end_page](#) (void)

- To be called at the end of each page.*
- virtual void **margins** (int *left, int *top, int *right, int *bottom)
 - Computes the dimensions of margins that lie between the printable page area and the full page.*
- void **print_widget** (FI_Widget *widget, int delta_x=0, int delta_y=0)
 - Synonym of **draw**(FI_Widget*, int, int)*
- void **print_window** (FI_Window *win, int x_off=0, int y_off=0)
 - Synonym of **draw_decorated_window**(FI_Window*, int, int)*
- virtual void **rotate** (float angle)
 - Rotates the graphics operations relatively to paper.*
- virtual void **scale** (float scale_x, float scale_y=0.)
 - Changes the scaling of page coordinates.*
- int **start_job** (int pagecount=0, int *frompage=NULL, int *topage=NULL, char **perr_message=NULL)
 - Synonym of **begin_job**(int pagecount, int *frompage, int *topage, char **perr_message).*
- int **start_page** ()
 - Synonym of **begin_page**().*
- virtual ~**FI_Paged_Device** ()
 - The destructor.*

Public Member Functions inherited from FI_Widget_Surface

- void **draw** (FI_Widget *widget, int delta_x=0, int delta_y=0)
 - Draws the widget on the drawing surface.*
- void **draw_decorated_window** (FI_Window *win, int x_offset=0, int y_offset=0)
 - Draws a window with its title bar and frame if any.*
- virtual void **origin** (int *x, int *y)
 - Computes the coordinates of the current origin of graphics functions.*
- virtual void **origin** (int x, int y)
 - Sets the position of the origin of graphics in the drawable part of the drawing surface.*
- void **print_window_part** (FI_Window *win, int x, int y, int w, int h, int delta_x=0, int delta_y=0)
 - Draws a rectangular part of an on-screen window.*
- virtual int **printable_rect** (int *w, int *h)
 - Computes the width and height of the drawable area of the drawing surface.*
- virtual void **translate** (int x, int y)
 - Translates the current graphics origin accounting for the current rotation.*
- virtual void **untranslate** ()
 - Undoes the effect of a previous **translate**() call.*

Public Member Functions inherited from FI_Surface_Device

- FI_Graphics_Driver * **driver** ()
 - Returns the graphics driver of this drawing surface.*
- virtual bool **is_current** ()
 - Is this surface the current drawing surface?*
- virtual void **set_current** (void)
 - Make this surface the current drawing surface.*
- virtual ~**FI_Surface_Device** ()
 - The destructor.*

Static Public Attributes

- static const **page_format** **page_formats** [NO_PAGE_FORMATS]
 - width, height and name of all elements of the enum **Page_Format**.*

Protected Member Functions

- **FI_Paged_Device** ()

The constructor.

Protected Member Functions inherited from [FI_Widget_Surface](#)

- [FI_Widget_Surface](#) (FI_Graphics_Driver *d)

The constructor.

Protected Member Functions inherited from [FI_Surface_Device](#)

- void **driver** (FI_Graphics_Driver *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void [end_current](#) ()
FLTK calls this each time a surface ceases to be the current drawing surface.
- **FI_Surface_Device** (FI_Graphics_Driver *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Surface_Device](#)

- static [FI_Surface_Device](#) * [pop_current](#) ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void [push_current](#) ([FI_Surface_Device](#) *new_current)
Pushes new_current on top of the stack of current drawing surfaces, and makes it current.
- static [FI_Surface_Device](#) * [surface](#) ()
The current drawing surface.

Protected Attributes inherited from [FI_Widget_Surface](#)

- int **x_offset**
horizontal offset to the origin of graphics coordinates
- int **y_offset**
vertical offset to the origin of graphics coordinates

11.97.1 Detailed Description

Represents page-structured drawing surfaces.

This class has no public constructor: don't instantiate it; use [FI_Printer](#) or [FI_PostScript_File_Device](#) instead.

11.97.2 Member Enumeration Documentation

11.97.2.1 Page_Format

```
enum Fl\_Paged\_Device::Page\_Format
```

Possible page formats.

All paper formats with pre-defined width and height. The [FI_Paged_Device::page_formats](#) array gives these widths and heights.

Enumerator

A0	A0 format.
A1	A1 format.
A2	A2 format.
A3	A3 format.

Enumerator

A4	A4 format.
A5	A5 format.
A6	A6 format.
A7	A7 format.
A8	A8 format.
A9	A9 format.
B0	B0 format.
B1	B1 format.
B2	B2 format.
B3	B3 format.
B4	B4 format.
B5	B5 format.
B6	B6 format.
B7	B7 format.
B8	B8 format.
B9	B9 format.
B10	B10 format.
EXECUTIVE	Executive format.
FOLIO	Folio format.
LEDGER	Ledger format.
LEGAL	Legal format.
LETTER	Letter format.
TABLOID	Tabloid format.

11.97.2.2 Page_Layout

enum `Fl_Paged_Device::Page_Layout`

Possible page layouts.

Enumerator

PORTRAIT	Portrait orientation.
LANDSCAPE	Landscape orientation.
REVERSED	Reversed orientation.
ORIENTATION	orientation

11.97.3 Member Function Documentation

11.97.3.1 begin_job()

```
int Fl_Paged_Device::begin_job (
    int pagecount = 0,
    int * frompage = NULL,
    int * topage = NULL,
    char ** perr_message = NULL) [virtual]
```

Begins a print job.

Parameters

in	<i>pagecount</i>	the total number of pages of the job (or 0 if you don't know the number of pages)
out	<i>frompage</i>	if non-null, *frompage is set to the first page the user wants printed

Parameters

out	<i>topage</i>	if non-null, *topage is set to the last page the user wants printed
out	<i>perr_message</i>	if non-null and if the returned value is 2, *perr_message is set to a string describing the error. That string can be deleted after use.

Returns

0 if OK, 1 if user cancelled the job, 2 if any error.

Reimplemented in [FI_PDF_File_Surface](#), [FI_PostScript_File_Device](#), and [FI_Printer](#).

11.97.3.2 begin_page()

```
int Fl_Paged_Device::begin_page (
    void ) [virtual]
```

Begins a new printed page.

The page coordinates are initially in points, i.e., 1/72 inch, and with origin at the top left of the printable page area. This function also makes this surface the current drawing surface with [FI_Surface_Device::push_current\(\)](#).

Note

[begin_page\(\)](#) calls [FI_Surface_Device::push_current\(\)](#) and leaves this device as the active surface. If any calls between [begin_page\(\)](#) and [end_page\(\)](#) open dialog boxes or will otherwise draw into FLTK windows, those calls must be put between a call to [FI_Surface_Device::pop_current\(\)](#) and a call to [FI_Surface_Device::push_current\(\)](#), or the content of the dialog box will be rendered to the printer instead of the screen.

Returns

0 if OK, non-zero if any error

Reimplemented in [FI_PDF_File_Surface](#), [FI_PostScript_File_Device](#), and [FI_Printer](#).

11.97.3.3 end_job()

```
void Fl_Paged_Device::end_job (
    void ) [virtual]
```

To be called at the end of a print job.

Reimplemented in [FI_PDF_File_Surface](#), [FI_PostScript_File_Device](#), and [FI_Printer](#).

11.97.3.4 end_page()

```
int Fl_Paged_Device::end_page (
    void ) [virtual]
```

To be called at the end of each page.

This function also stops this surface from being the current drawing surface with [FI_Surface_Device::pop_current\(\)](#).

Note

[end_page\(\)](#) calls [FI_Surface_Device::pop_current\(\)](#). If any calls between [begin_page\(\)](#) and [end_page\(\)](#) open dialog boxes or will otherwise draw into FLTK windows, those calls must be put between a call to [FI_Surface_Device::pop_current\(\)](#) and a call to [FI_Surface_Device::push_current\(\)](#).

Returns

0 if OK, non-zero if any error.

Reimplemented in [FI_PDF_File_Surface](#), [FI_PostScript_File_Device](#), and [FI_Printer](#).

11.97.3.5 margins()

```
void Fl_Paged_Device::margins (
    int * left,
    int * top,
    int * right,
    int * bottom) [virtual]
```

Computes the dimensions of margins that lie between the printable page area and the full page. Values are in the same unit as that used by FLTK drawing functions. They are changed by [scale\(\)](#) calls.

Parameters

out	<i>left</i>	If non-null, *left is set to the left margin size.
out	<i>top</i>	If non-null, *top is set to the top margin size.
out	<i>right</i>	If non-null, *right is set to the right margin size.
out	<i>bottom</i>	If non-null, *bottom is set to the bottom margin size.

Reimplemented in [Fl_PDF_File_Surface](#), [Fl_PostScript_File_Device](#), and [Fl_Printer](#).

11.97.3.6 rotate()

```
void Fl_Paged_Device::rotate (
    float angle) [virtual]
```

Rotates the graphics operations relatively to paper. The rotation is centered on the current graphics origin. Successive [rotate\(\)](#) calls don't combine their effects.

Parameters

<i>angle</i>	Rotation angle in counter-clockwise degrees.

Reimplemented in [Fl_PDF_File_Surface](#), [Fl_PostScript_File_Device](#), and [Fl_Printer](#).

11.97.3.7 scale()

```
void Fl_Paged_Device::scale (
    float scale_x,
    float scale_y = 0.) [virtual]
```

Changes the scaling of page coordinates.

This function also resets the origin of graphics functions at top left of printable page area. After a [scale\(\)](#) call, do a [printable_rect\(\)](#) call to get the new dimensions of the printable page area. Successive [scale\(\)](#) calls don't combine their effects.

Parameters

<i>scale_x</i>	Horizontal dimensions of plot are multiplied by this quantity.
<i>scale_y</i>	Same as above, vertically. The value 0. is equivalent to setting <code>scale_y = scale_x</code> . Thus, <code>scale(factor);</code> is equivalent to <code>scale(factor, factor);</code>

Reimplemented in [Fl_PDF_File_Surface](#), [Fl_PostScript_File_Device](#), and [Fl_Printer](#).

11.97.3.8 start_job()

```
int Fl_Paged_Device::start_job (
    int pagecount = 0,
    int * frompage = NULL,
    int * topage = NULL,
    char ** perr_message = NULL) [inline]
```

Synonym of [begin_job\(int pagecount, int *frompage, int *topage, char **perr_message\)](#).
For API compatibility with FLTK 1.3.x

11.97.3.9 start_page()

```
int Fl_Paged_Device::start_page () [inline]
```

Synonym of [begin_page\(\)](#).

For API compatibility with FLTK 1.3.x

The documentation for this class was generated from the following files:

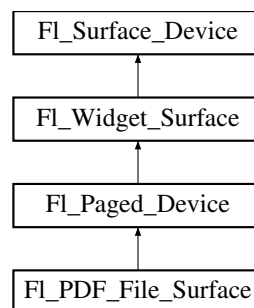
- [Fl_Paged_Device.H](#)
- [Fl_Paged_Device.cxx](#)

11.98 Fl_PDF_File_Surface Class Reference

To send graphical output to a PDF file.

```
#include <Fl_PDF_File_Surface.H>
```

Inheritance diagram for Fl_PDF_File_Surface:



Public Member Functions

- int [begin_document](#) (const char *pathname, enum [Fl_Paged_Device::Page_Format](#) format=[Fl_Paged_Device::A4](#), enum [Fl_Paged_Device::Page_Layout](#) layout=[Fl_Paged_Device::PORTRAIT](#), char **perr=NULL)
Prepare to draw to a PDF document identified by its pathname.
- int [begin_job](#) (const char *defaultfilename, char **perr=NULL)
Prepare to draw to a PDF document identified with a file chooser.
- int [begin_job](#) (int, int *, int *, char **) [FL_OVERRIDE](#)
Don't use for this class.
- int [begin_page](#) (void) [FL_OVERRIDE](#)
Begins a new printed page.
- void [end_job](#) (void) [FL_OVERRIDE](#)
To be called at the end of a print job.
- int [end_page](#) (void) [FL_OVERRIDE](#)
To be called at the end of each page.
- bool [is_current](#) () [FL_OVERRIDE](#)
Is this surface the current drawing surface?
- void [margins](#) (int *left, int *top, int *right, int *bottom) [FL_OVERRIDE](#)
Computes the dimensions of margins that lie between the printable page area and the full page.
- void [origin](#) (int *x, int *y) [FL_OVERRIDE](#)
Computes the coordinates of the current origin of graphics functions.
- void [origin](#) (int x, int y) [FL_OVERRIDE](#)
Sets the position of the origin of graphics in the drawable part of the drawing surface.
- const char * [pdf_filename](#) ()

- Returns the name of the PDF document.*
- int [printable_rect](#) (int *w, int *h) [FL_OVERRIDE](#)
Computes the width and height of the drawable area of the drawing surface.
- void [rotate](#) (float angle) [FL_OVERRIDE](#)
Rotates the graphics operations relatively to paper.
- void [scale](#) (float s_x, float s_y=0) [FL_OVERRIDE](#)
Changes the scaling of page coordinates.
- void [set_current](#) () [FL_OVERRIDE](#)
Make this surface the current drawing surface.
- void [translate](#) (int x, int y) [FL_OVERRIDE](#)
Translates the current graphics origin accounting for the current rotation.
- void [untranslate](#) () [FL_OVERRIDE](#)
Undoes the effect of a previous [translate\(\)](#) call.

Public Member Functions inherited from [FI_Paged_Device](#)

- void [print_widget](#) ([FI_Widget](#) *widget, int delta_x=0, int delta_y=0)
Synonym of [draw\(FI_Widget, int, int\)](#)*
- void [print_window](#) ([FI_Window](#) *win, int x_off=0, int y_off=0)
Synonym of [draw_decorated_window\(FI_Window, int, int\)](#)*
- int [start_job](#) (int pagecount=0, int *frompage=NULL, int *topage=NULL, char **perr_message=NULL)
*Synonym of [begin_job\(int pagecount, int *frompage, int *topage, char **perr_message\)](#).*
- int [start_page](#) ()
Synonym of [begin_page\(\)](#).
- virtual ~[FI_Paged_Device](#) ()
The destructor.

Public Member Functions inherited from [FI_Widget_Surface](#)

- void [draw](#) ([FI_Widget](#) *widget, int delta_x=0, int delta_y=0)
Draws the widget on the drawing surface.
- void [draw_decorated_window](#) ([FI_Window](#) *win, int x_offset=0, int y_offset=0)
Draws a window with its title bar and frame if any.
- void [print_window_part](#) ([FI_Window](#) *win, int x, int y, int w, int h, int delta_x=0, int delta_y=0)
Draws a rectangular part of an on-screen window.

Public Member Functions inherited from [FI_Surface_Device](#)

- [FI_Graphics_Driver](#) * [driver](#) ()
Returns the graphics driver of this drawing surface.
- virtual ~[FI_Surface_Device](#) ()
The destructor.

Static Public Attributes

These attributes are useful for the Wayland/X11 platform only.

- static const char * [format_dialog_title](#) = "PDF document settings"
Localizable text of the "PDF document settings" dialog.
- static const char * [format_dialog_page_size](#) = "Page Size:"
Localizable text of the "PDF document settings" dialog.
- static const char * [format_dialog_orientation](#) = "Orientation:"
Localizable text of the "PDF document settings" dialog.
- static const char * [format_dialog_default](#) = "Set as default"
Localizable text of the "PDF document settings" dialog.

Static Public Attributes inherited from [FI_Paged_Device](#)

- static const [page_format](#) **page_formats** [[NO_PAGE_FORMATS](#)]
width, height and name of all elements of the enum [Page_Format](#).

Additional Inherited Members

Public Types inherited from [FI_Paged_Device](#)

- enum [Page_Format](#) {
[A0](#) = 0 , [A1](#) , [A2](#) , [A3](#) ,
[A4](#) , [A5](#) , [A6](#) , [A7](#) ,
[A8](#) , [A9](#) , [B0](#) , [B1](#) ,
[B2](#) , [B3](#) , [B4](#) , [B5](#) ,
[B6](#) , [B7](#) , [B8](#) , [B9](#) ,
[B10](#) , [C5E](#) , [DLE](#) , [EXECUTIVE](#) ,
[FOLIO](#) , [LEDGER](#) , [LEGAL](#) , [LETTER](#) ,
[TABLOID](#) , [ENVELOPE](#) , [MEDIA](#) = 0x1000 }
Possible page formats.
- enum [Page_Layout](#) { [PORTRAIT](#) = 0 , [LANDSCAPE](#) = 0x100 , [REVERSED](#) = 0x200 , [ORIENTATION](#) = 0x300 }
Possible page layouts.

Static Public Member Functions inherited from [FI_Surface_Device](#)

- static [FI_Surface_Device](#) * [pop_current](#) ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void [push_current](#) ([FI_Surface_Device](#) *new_current)
Pushes new_current on top of the stack of current drawing surfaces, and makes it current.
- static [FI_Surface_Device](#) * [surface](#) ()
The current drawing surface.

Protected Member Functions inherited from [FI_Paged_Device](#)

- [FI_Paged_Device](#) ()
The constructor.

Protected Member Functions inherited from [FI_Widget_Surface](#)

- [FI_Widget_Surface](#) ([FI_Graphics_Driver](#) *d)
The constructor.

Protected Member Functions inherited from [FI_Surface_Device](#)

- void **driver** ([FI_Graphics_Driver](#) *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void [end_current](#) ()
FLTK calls this each time a surface ceases to be the current drawing surface.
- [FI_Surface_Device](#) ([FI_Graphics_Driver](#) *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

Protected Attributes inherited from [FI_Widget_Surface](#)

- int **x_offset**
horizontal offset to the origin of graphics coordinates
- int **y_offset**
vertical offset to the origin of graphics coordinates

11.98.1 Detailed Description

To send graphical output to a PDF file.

Class `Fl_PDF_File_Surface` is used exactly as the `Fl_Printer` class except for its 2 member functions `begin_job()` and `begin_document()`.

Platform notes:

- Windows: requires "Microsoft Print to PDF" available in Windows 10 and later.
- Wayland/X11: requires the FLTK library was built with `FLTK_USE_PANGO=1`.
- macOS: requires macOS 10.9 or later.

If the running platform doesn't fulfill the requirement above, the program runs but doesn't output any PDF.

11.98.2 Member Function Documentation

11.98.2.1 `begin_document()`

```
int Fl_PDF_File_Surface::begin_document (
    const char * pathname,
    enum Fl_Paged_Device::Page_Format format = Fl_Paged_Device::A4,
    enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT,
    char ** perr = NULL)
```

Prepare to draw to a PDF document identified by its pathname.

Parameters

<i>pathname</i>	Path name for the PDF document
<i>format</i>	The paper format for the PDF document
<i>layout</i>	The orientation for the PDF document
<i>perr</i>	NULL or address of a string that receives a message in case of error. To be deleted[] after use.

Returns

0 for success, 2 when an error occurred.

11.98.2.2 `begin_job()` [1/2]

```
int Fl_PDF_File_Surface::begin_job (
    const char * defaultfilename,
    char ** perr = NULL)
```

Prepare to draw to a PDF document identified with a file chooser.

A dialog opens to select the location and name of the output PDF document as well as its page format and orientation.

Parameters

<i>defaultfilename</i>	Default name for the PDF document
<i>perr</i>	NULL or address of a string that receives a message in case of error. To be deleted[] after use.

Returns

0 for success, 1 when the user cancelled the operation, 2 when an error occurred.

11.98.2.3 begin_job() [2/2]

```
int Fl_PDF_File_Surface::begin_job (
    int ,
    int * ,
    int * ,
    char ** ) [inline], [virtual]
```

Don't use for this class.

Reimplemented from [Fl_Paged_Device](#).

11.98.2.4 begin_page()

```
int Fl_PDF_File_Surface::begin_page (
    void ) [inline], [virtual]
```

Begins a new printed page.

The page coordinates are initially in points, i.e., 1/72 inch, and with origin at the top left of the printable page area. This function also makes this surface the current drawing surface with [Fl_Surface_Device::push_current\(\)](#).

Note

[begin_page\(\)](#) calls [Fl_Surface_Device::push_current\(\)](#) and leaves this device as the active surface. If any calls between [begin_page\(\)](#) and [end_page\(\)](#) open dialog boxes or will otherwise draw into FLTK windows, those calls must be put between a call to [Fl_Surface_Device::pop_current\(\)](#) and a call to [Fl_Surface_Device::push_current\(\)](#), or the content of the dialog box will be rendered to the printer instead of the screen.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Paged_Device](#).

11.98.2.5 end_job()

```
void Fl_PDF_File_Surface::end_job (
    void ) [inline], [virtual]
```

To be called at the end of a print job.

Reimplemented from [Fl_Paged_Device](#).

11.98.2.6 end_page()

```
int Fl_PDF_File_Surface::end_page (
    void ) [inline], [virtual]
```

To be called at the end of each page.

This function also stops this surface from being the current drawing surface with [Fl_Surface_Device::pop_current\(\)](#).

Note

[end_page\(\)](#) calls [Fl_Surface_Device::pop_current\(\)](#). If any calls between [begin_page\(\)](#) and [end_page\(\)](#) open dialog boxes or will otherwise draw into FLTK windows, those calls must be put between a call to [Fl_Surface_Device::pop_current\(\)](#) and a call to [Fl_Surface_Device::push_current\(\)](#).

Returns

0 if OK, non-zero if any error.

Reimplemented from [Fl_Paged_Device](#).

11.98.2.7 is_current()

```
bool Fl_PDF_File_Surface::is_current () [inline], [virtual]
```

Is this surface the current drawing surface?

Reimplemented from [Fl_Surface_Device](#).

11.98.2.8 margins()

```
void Fl_PDF_File_Surface::margins (
    int * left,
    int * top,
    int * right,
    int * bottom) [inline], [virtual]
```

Computes the dimensions of margins that lie between the printable page area and the full page. Values are in the same unit as that used by FLTK drawing functions. They are changed by [scale\(\)](#) calls.

Parameters

out	<i>left</i>	If non-null, *left is set to the left margin size.
out	<i>top</i>	If non-null, *top is set to the top margin size.
out	<i>right</i>	If non-null, *right is set to the right margin size.
out	<i>bottom</i>	If non-null, *bottom is set to the bottom margin size.

Reimplemented from [Fl_Paged_Device](#).

11.98.2.9 origin() [1/2]

```
void Fl_PDF_File_Surface::origin (
    int * x,
    int * y) [inline], [virtual]
```

Computes the coordinates of the current origin of graphics functions.

Parameters

out	<i>x,y</i>	If non-null, *x and *y are set to the horizontal and vertical coordinates of the graphics origin.

Reimplemented from [Fl_Widget_Surface](#).

11.98.2.10 origin() [2/2]

```
void Fl_PDF_File_Surface::origin (
    int x,
    int y) [inline], [virtual]
```

Sets the position of the origin of graphics in the drawable part of the drawing surface. Arguments should be expressed relatively to the result of a previous [printable_rect\(\)](#) call. That is, `printable_rect(&w, &h); origin(w/2, 0);` sets the graphics origin at the top center of the drawable area. Successive [origin\(\)](#) calls don't combine their effects. Origin() calls are not affected by [rotate\(\)](#) calls (for classes derived from [Fl_Paged_Device](#)).

Parameters

in	<i>x,y</i>	Horizontal and vertical positions in the drawing surface of the desired origin of graphics.

Reimplemented from [Fl_Widget_Surface](#).

11.98.2.11 printable_rect()

```
int Fl_PDF_File_Surface::printable_rect (
    int * w,
    int * h) [inline], [virtual]
```

Computes the width and height of the drawable area of the drawing surface. Values are in the same unit as that used by FLTK drawing functions and are unchanged by calls to [origin\(\)](#). If the object is derived from class [Fl_Paged_Device](#), values account for the user-selected paper type and print orientation and are changed by [scale\(\)](#) calls.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Widget_Surface](#).

11.98.2.12 rotate()

```
void Fl_PDF_File_Surface::rotate (
    float angle) [inline], [virtual]
```

Rotates the graphics operations relatively to paper.

The rotation is centered on the current graphics origin. Successive [rotate\(\)](#) calls don't combine their effects.

Parameters

<i>angle</i>	Rotation angle in counter-clockwise degrees.

Reimplemented from [Fl_Paged_Device](#).

11.98.2.13 scale()

```
void Fl_PDF_File_Surface::scale (
    float scale_x,
    float scale_y = 0) [inline], [virtual]
```

Changes the scaling of page coordinates.

This function also resets the origin of graphics functions at top left of printable page area. After a [scale\(\)](#) call, do a [printable_rect\(\)](#) call to get the new dimensions of the printable page area. Successive [scale\(\)](#) calls don't combine their effects.

Parameters

<i>scale_x</i>	Horizontal dimensions of plot are multiplied by this quantity.
<i>scale_y</i>	Same as above, vertically. The value 0. is equivalent to setting <code>scale_y = scale_x</code> . Thus, <code>scale(factor);</code> is equivalent to <code>scale(factor, factor);</code>

Reimplemented from [Fl_Paged_Device](#).

11.98.2.14 set_current()

```
void Fl_PDF_File_Surface::set_current (
    void ) [inline], [virtual]
```

Make this surface the current drawing surface.

This surface will receive all future graphics requests.

Since FLTK 1.4.0 the preferred API to change the current drawing surface is [Fl_Surface_Device::push_current\(\)](#) / [Fl_Surface_Device::pop_current\(\)](#).

Note

It is recommended to use this function only as follows :

- The current drawing surface is the display;
- make current another surface, e.g., an [Fl_Printer](#) or an [Fl_Image_Surface](#) object, calling [set_current\(\)](#) on this object;
- draw to that surface;
- make the display current again with [Fl_Display_Device::display_device\(\)->set_current\(\)](#); don't do any other call to [set_current\(\)](#) before this one.

Other scenarios of drawing surface changes should be performed via [Fl_Surface_Device::push_current\(\)](#) and [Fl_Surface_Device::pop_current\(\)](#).

Reimplemented from [Fl_Surface_Device](#).

11.98.2.15 translate()

```
void Fl_PDF_File_Surface::translate (
    int x,
    int y) [inline], [virtual]
```

Translates the current graphics origin accounting for the current rotation.

Each [translate\(\)](#) call must be matched by an [untranslate\(\)](#) call. Successive [translate\(\)](#) calls add up their effects.

Reimplemented from [Fl_Widget_Surface](#).

11.98.2.16 untranslate()

```
void Fl_PDF_File_Surface::untranslate (
    void ) [inline], [virtual]
```

Undoes the effect of a previous [translate\(\)](#) call.

Reimplemented from [Fl_Widget_Surface](#).

The documentation for this class was generated from the following files:

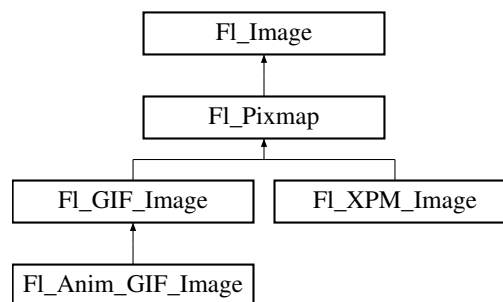
- [Fl_PDF_File_Surface.H](#)
- [Fl_Device.cxx](#)

11.99 Fl_Pixmap Class Reference

The [Fl_Pixmap](#) class supports caching and drawing of colormap (pixmap) images, including transparency.

```
#include <Fl_Pixmap.H>
```

Inheritance diagram for [Fl_Pixmap](#):



Public Member Functions

- int **cache_h** ()
- int **cache_w** ()
- void **color_average** ([Fl_Color](#) c, float i) [FL_OVERRIDE](#)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [Fl_Image](#) * **copy** () const
- [Fl_Image](#) * **copy** (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void **desaturate** () [FL_OVERRIDE](#)
The [desaturate\(\)](#) method converts an image to grayscale.
- void **draw** (int X, int Y)
- void **draw** (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [Fl_Pixmap](#) (char *const *D)
The constructors create a new pixmap from the specified XPM data.
- [Fl_Pixmap](#) (const char *const *D)
The constructors create a new pixmap from the specified XPM data.
- [Fl_Pixmap](#) (const uchar *const *D)

The constructors create a new pixmap from the specified XPM data.

- [FL_Pixmap](#) ([uchar](#) *const *D)

The constructors create a new pixmap from the specified XPM data.

- void [label](#) ([FL_Menu_Item](#) *m) [FL_OVERRIDE](#)

This method is an obsolete way to set the image attribute of a menu item.

- void [label](#) ([FL_Widget](#) *w) [FL_OVERRIDE](#)

This method is an obsolete way to set the image attribute of a widget or menu item.

- void [uncache](#) () [FL_OVERRIDE](#)

If the image has been cached for display, delete the cache data.

- virtual [~FL_Pixmap](#) ()

The destructor frees all memory and server resources that are used by the pixmap.

Public Member Functions inherited from [FL_Image](#)

- virtual class [FL_Shared_Image](#) * [as_shared_image](#) ()

Returns whether an image is an [FL_Shared_Image](#) or not.

- [FL_Image](#) * [copy](#) () const

Creates a copy of the image in the same size.

- int [count](#) () const

Returns the number of data values associated with the image.

- int [d](#) () const

Returns the image depth.

- const char *const * [data](#) () const

Returns a pointer to the current image data array.

- int [data_h](#) () const

Returns the height of the image data.

- int [data_w](#) () const

Returns the width of the image data.

- void [draw](#) (int X, int Y)

Draws the image to the current drawing surface.

- int [fail](#) () const

Returns a value that is not 0 if there is currently no image available.

- [FL_Image](#) (int W, int H, int D)

The constructor creates an empty image with the specified width, height, and depth.

- int [h](#) () const

Returns the current image drawing height in FLTK units.

- void [inactive](#) ()

The [inactive\(\)](#) method calls [color_average\(FL_BACKGROUND_COLOR, 0.33f\)](#) to produce an image that appears grayed out.

- int [ld](#) () const

Returns the current line data size in bytes.

- virtual void [release](#) ()

Releases an [FL_Image](#) - the same as 'delete this'.

- virtual void [scale](#) (int width, int height, int proportional=1, int can_expand=0)

Sets the drawing size of the image.

- int [w](#) () const

Returns the current image drawing width in FLTK units.

- virtual [~FL_Image](#) ()

The destructor is a virtual method that frees all memory used by the image.

Public Attributes

- int [alloc_data](#)

Protected Member Functions

- void **measure** ()

Protected Member Functions inherited from [FI_Image](#)

- void **d** (int D)
Sets the current image depth.
- void **data** (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void **draw_empty** (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int **draw_scaled** (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void **h** (int H)
Sets the height of the image data.
- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Friends

- class **FI_Graphics_Driver**

Additional Inherited Members**Static Public Member Functions inherited from [FI_Image](#)**

- static [FI_Labeltype](#) **define_FL_IMAGE_LABEL** ()
- static [FI_RGB_Scaling](#) **RGB_scaling** ()
Returns the currently used RGB image scaling method.
- static void **RGB_scaling** ([FI_RGB_Scaling](#))
Sets the RGB image scaling method used for copy(int, int).
- static [FI_RGB_Scaling](#) **scaling_algorithm** ()
Gets what algorithm is used when resizing a source image to draw it.
- static void **scaling_algorithm** ([FI_RGB_Scaling](#) algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Static Public Attributes inherited from [FI_Image](#)

- static const int **ERR_FILE_ACCESS** = -2
- static const int **ERR_FORMAT** = -3
- static const int **ERR_MEMORY_ACCESS** = -4
- static const int **ERR_NO_IMAGE** = -1
- static bool **register_images_done** = false
True after [fl_register_images\(\)](#) was called, false before.

Static Protected Member Functions inherited from [FI_Image](#)

- static void **labeltype** (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void **measure** (const [FI_Label](#) *lo, int &lw, int &lh)

11.99.1 Detailed Description

The [FI_Pixmap](#) class supports caching and drawing of colormap (pixmap) images, including transparency.

11.99.2 Constructor & Destructor Documentation

11.99.2.1 Fl_Pixmap() [1/4]

```
Fl_Pixmap::Fl_Pixmap (
    char *const * D) [inline], [explicit]
```

The constructors create a new pixmap from the specified XPM data.

11.99.2.2 Fl_Pixmap() [2/4]

```
Fl_Pixmap::Fl_Pixmap (
    uchar *const * D) [inline], [explicit]
```

The constructors create a new pixmap from the specified XPM data.

11.99.2.3 Fl_Pixmap() [3/4]

```
Fl_Pixmap::Fl_Pixmap (
    const char *const * D) [inline], [explicit]
```

The constructors create a new pixmap from the specified XPM data.

11.99.2.4 Fl_Pixmap() [4/4]

```
Fl_Pixmap::Fl_Pixmap (
    const uchar *const * D) [inline], [explicit]
```

The constructors create a new pixmap from the specified XPM data.

11.99.3 Member Function Documentation

11.99.3.1 color_average()

```
void Fl_Pixmap::color_average (
    Fl_Color c,
    float i) [virtual]
```

The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.

The first argument specifies the FLTK color to be used.

The second argument specifies the amount of the original image to combine with the color, so a value of 1.0 results in no color blend, and a value of 0.0 results in a constant image of the specified color.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented from [Fl_Image](#).

11.99.3.2 copy()

```
Fl_Image * Fl_Pixmap::copy (
    int W,
    int H) const [virtual]
```

Creates a resized copy of the image.

It is recommended not to call this member function to reduce the size of an image to the size of the area where this image will be drawn, and to use [Fl_Image::scale\(\)](#) instead.

The new image should be released when you are done with it.

Note: since FLTK 1.4.0 you can use [Fl_Image::release\(\)](#) for all types of images (i.e. all subclasses of [Fl_Image](#)) instead of operator *delete* for [Fl_Image](#)'s and [Fl_Image::release\(\)](#) for [Fl_Shared_Image](#)'s.

The new image data will be converted to the requested size. RGB images are resized using the algorithm set by [Fl_Image::RGB_scaling\(\)](#).

For the new image the following equations are true:

- $w() == \text{data\_w}() == \bar{W}$
- $h() == \text{data\_h}() == \bar{H}$

Parameters

in	$\bar{W}, \bar{H}$	Requested width and height of the new image

Note

The returned image can be safely cast to the same image type as that of the source image provided this type is one of [Fl_RGB_Image](#), [Fl_SVG_Image](#), [Fl_Pixmap](#), [Fl_Bitmap](#), [Fl_Tiled_Image](#), [Fl_Anim_GIF_Image](#) and [Fl_Shared_Image](#). Returned objects copied from images of other, derived, image classes belong to the parent class appearing in this list. For example, the copy of an [Fl_GIF_Image](#) is an object of class [Fl_Pixmap](#).

Since FLTK 1.4.0 this method is 'const'. If you derive your own class from [Fl_Image](#) or any subclass your overridden methods of '[Fl_Image::copy\(\) const](#)' and '[Fl_Image::copy\(int, int\) const](#)' **must** also be 'const' for inheritance to work properly. This is different than in FLTK 1.3.x and earlier where these methods have not been 'const'.

Reimplemented from [Fl_Image](#).

11.99.3.3 desaturate()

```
void Fl_Pixmap::desaturate () [virtual]
```

The [desaturate\(\)](#) method converts an image to grayscale.

If the image contains an alpha channel (depth = 4), the alpha channel is preserved.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented from [Fl_Image](#).

11.99.3.4 draw()

```
void Fl_Pixmap::draw (
    int X,
    int Y,
    int W,
    int H,
    int cx = 0,
    int cy = 0) [virtual]
```

Draws the image to the current drawing surface with a bounding box.

Arguments X, Y, W, H specify a bounding box for the image, with the origin (upper-left corner) of the image offset by the cx and cy arguments.

In other words: `fl_push_clip(X,Y,W,H)` is applied, the image is drawn with its upper-left corner at X-cx, Y-cy and its own width and height, `fl_pop_clip()` is applied.

Reimplemented from [Fl_Image](#).

11.99.3.5 label() [1/2]

```
void Fl_Pixmap::label (
    Fl_Menu_Item * m) [virtual]
```

This method is an obsolete way to set the image attribute of a menu item.

Deprecated Please use [Fl_Menu_Item::image\(\)](#) instead.

Reimplemented from [Fl_Image](#).

11.99.3.6 label() [2/2]

```
void Fl_Pixmap::label (
    Fl_Widget * widget) [virtual]
```

This method is an obsolete way to set the image attribute of a widget or menu item.

Deprecated Please use [Fl_Widget::image\(\)](#) or [Fl_Widget::deimage\(\)](#) instead.

Reimplemented from [Fl_Image](#).

11.99.3.7 uncache()

```
void Fl_Pixmap::uncache () [virtual]
```

If the image has been cached for display, delete the cache data.

This allows you to change the data used for the image and then redraw it without recreating an image object.

Reimplemented from [Fl_Image](#).

The documentation for this class was generated from the following files:

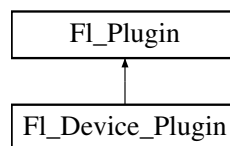
- [Fl_Pixmap.H](#)
- [Fl_Pixmap.cxx](#)

11.100 FI_Plugin Class Reference

[Fl_Plugin](#) allows link-time and run-time integration of binary modules.

```
#include <Fl_Plugin.H>
```

Inheritance diagram for [Fl_Plugin](#):



Public Member Functions

- [Fl_Plugin](#) (const char *class, const char *name)
Create a plugin.
- virtual ~[Fl_Plugin](#) ()
Clear the plugin and remove it from the database.

11.100.1 Detailed Description

[Fl_Plugin](#) allows link-time and run-time integration of binary modules.

[Fl_Plugin](#) and [Fl_Plugin_Manager](#) provide a small and simple solution for linking C++ classes at run-time, or optionally linking modules at compile time without the need to change the main application.

[Fl_Plugin_Manager](#) uses static initialization to create the plugin interface early during startup. Plugins are stored in a temporary database, organized in classes.

Plugins should derive a new class from [Fl_Plugin](#) as a base:

```
class My_Plugin : public Fl_Plugin {
public:
    My_Plugin() : Fl_Plugin("effects", "blur") { }
    void do_something(...);
};
```

```
My_Plugin blur_plugin();
```

Plugins can be put into modules and either linked before distribution, or loaded from dynamically linkable files. An

[Fl_Plugin_Manager](#) is used to list and access all currently loaded plugins.

```
Fl_Plugin_Manager mgr("effects");
int i, n = mgr.plugins();
for (i=0; i<n; i++) {
    My_Plugin *pin = (My_Plugin*)mgr.plugin(i);
    pin->do_something();
}
```

11.100.2 Constructor & Destructor Documentation

11.100.2.1 FI_Plugin()

```
FI_Plugin::FI_Plugin (
    const char * klass,
    const char * name)
```

Create a plugin.

Parameters

in	<i>klass</i>	plugins are grouped in classes
in	<i>name</i>	every plugin should have a unique name

The documentation for this class was generated from the following files:

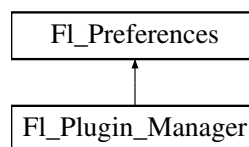
- FI_Plugin.H
- FI_Preferences.cxx

11.101 FI_Plugin_Manager Class Reference

[FI_Plugin_Manager](#) manages link-time and run-time plugin binaries.

```
#include <FI_Plugin.H>
```

Inheritance diagram for FI_Plugin_Manager:



Public Member Functions

- [FI_Preferences::ID addPlugin](#) (const char **name*, [FI_Plugin](#) **plugin*)
This function adds a new plugin to the database.
- **FI_Plugin_Manager** (const char **klass*)
Manage all plugins belonging to one class.
- [FI_Plugin](#) * **plugin** (const char **name*)
Return the address of a plugin by name.
- [FI_Plugin](#) * **plugin** (int *index*)
Return the address of a plugin by index.
- int **plugins** ()
Return the number of plugins in the klass.
- [~FI_Plugin_Manager](#) ()
Remove the plugin manager.

Public Member Functions inherited from [FI_Preferences](#)

- char **clear** ()
Delete all groups and all entries.
- char **delete_all_entries** ()
Delete all entries.
- char **delete_all_groups** ()
Delete all groups.
- char [delete_entry](#) (const char **entry*)

- Deletes a single name/value pair.*

 - char `delete_group` (const char *group)
- Deletes a group.*

 - int `dirty` ()
- Check if there were changes to the database that need to be written to disk.*

 - int `entries` ()
- Returns the number of entries (name/value pairs) in a group.*

 - const char * `entry` (int index)
- Returns the name of an entry.*

 - char `entry_exists` (const char *key)
- Returns non-zero if an entry with this name exists.*

 - `Root filename` (char *buffer, size_t buffer_size)
- Return the file name and path to the preference file.*

 - `FI_Preferences` (const char *path, const char *vendor, const char *application)
- Deprecated: Use this constructor to create or read a preference file at an arbitrary position in the file system.*

 - `FI_Preferences` (const char *path, const char *vendor, const char *application, `Root` flags)
- Use this constructor to create or read a preference file at an arbitrary position in the file system.*

 - **`FI_Preferences`** (const `FI_Preferences` &)
- Create another reference to a Preferences group.*

 - `FI_Preferences` (`FI_Preferences` &parent, const char *group)
- Generate or read a new group of entries within another group.*

 - `FI_Preferences` (`FI_Preferences` &parent, int groupIndex)
- Open a child group using a given index.*

 - `FI_Preferences` (`FI_Preferences` *parent, const char *group)
- Create or access a group of preferences using a name.*

 - `FI_Preferences` (`FI_Preferences` *parent, int groupIndex)
- Create a new dataset access point using a dataset ID.*

 - `FI_Preferences` (`Root` root, const char *vendor, const char *application)
- The constructor creates a group that manages key/value pairs and child groups.*

 - int `flush` ()
- Writes preferences to disk if they were modified.*

 - char `get` (const char *entry, char *&value, const char *defaultValue)
- Reads an entry from the group.*

 - char `get` (const char *entry, char *value, const char *defaultValue, int maxSize)
- Reads an entry from the group.*

 - char `get` (const char *entry, double &value, double defaultValue)
- Reads an entry from the group.*

 - char `get` (const char *entry, float &value, float defaultValue)
- Reads an entry from the group.*

 - char `get` (const char *entry, int &value, int defaultValue)
- Reads an entry from the group.*

 - char `get` (const char *entry, std::string &value, const std::string &defaultValue)
- Reads an entry from the group.*

 - char `get` (const char *entry, void *&value, const void *defaultValue, int defaultSize)
- Reads an entry from the group.*

 - char `get` (const char *entry, void *value, const void *defaultValue, int defaultSize, int *size)
- Reads a binary entry from the group, encoded in hexadecimal blocks.*

 - char `get` (const char *entry, void *value, const void *defaultValue, int defaultSize, int maxSize)
- Reads a binary entry from the group, encoded in hexadecimal blocks.*

 - char `get_userdata_path` (char *path, int pathlen)

- Creates a path that is related to the preference file and that is usable for additional application data.*

 - `const char * group (int num_group)`
Returns the name of the Nth (`num_group`) group.
 - `char group\_exists (const char *key)`
Returns non-zero if a group with this name exists.
 - `int groups ()`
Returns the number of groups that are contained within a group.
 - `ID id ()`
Return an ID that can later be reused to open more references to this dataset.
 - `const char * name ()`
Return the name of this entry.
 - `const char * path ()`
Return the full path to this entry.
 - `char set (const char *entry, const char *value)`
Sets an entry (name/value pair).
 - `char set (const char *entry, const std::string &value)`
Sets an entry (name/value pair).
 - `char set (const char *entry, const void *value, int size)`
Sets an entry (name/value pair).
 - `char set (const char *entry, double value)`
Sets an entry (name/value pair).
 - `char set (const char *entry, double value, int precision)`
Sets an entry (name/value pair).
 - `char set (const char *entry, float value)`
Sets an entry (name/value pair).
 - `char set (const char *entry, float value, int precision)`
Sets an entry (name/value pair).
 - `char set (const char *entry, int value)`
Sets an entry (name/value pair).
 - `int size (const char *entry)`
Returns the size of the value part of an entry.
 - `virtual ~FI\_Preferences ()`
The destructor removes allocated resources.

Static Public Member Functions

- `static int load (const char *filename)`
Load a module from disk.
- `static int loadAll (const char *dirpath, const char *pattern=0)`
Use this function to load a whole directory full of modules.
- `static void removePlugin (FI\_Preferences::ID id)`
Remove any plugin.

Static Public Member Functions inherited from [FI_Preferences](#)

- `static unsigned int file\_access ()`
Return the current file access permissions for the FLTK preferences system.
- `static void file\_access (unsigned int flags)`
Tell the FLTK preferences system which files in the file system it may read, create, or write.
- `static Root filename (char *buffer, size_t buffer_size, Root root, const char *vendor, const char *application)`
Determine the file name and path to preferences that would be opened with these parameters.
- `static const char * new\_UUID ()`

Returns a UUID as generated by the system.

- static char **remove** (ID id_)

Remove the group with this ID from a database.

Additional Inherited Members

Public Types inherited from FI_Preferences

- typedef void * ID

Every FI_Preferences-Group has a unique ID.

- enum Root {
UNKNOWN_ROOT_TYPE = -1 , SYSTEM = 0 , USER , MEMORY ,
ROOT_MASK = 0x00FF , CORE = 0x0100 , C_LOCALE = 0x1000 , CLEAR = 0x2000 ,
SYSTEM_L = SYSTEM | C_LOCALE , USER_L = USER | C_LOCALE , CORE_SYSTEM_L = CORE |
SYSTEM_L , CORE_USER_L = CORE | USER_L ,
CORE_SYSTEM = CORE | SYSTEM , CORE_USER = CORE | USER }

Define the scope of the preferences.

Static Public Attributes inherited from FI_Preferences

- static const unsigned int **ALL** = ALL_READ_OK | ALL_WRITE_OK
Set this to give FLTK and applications permission to read, write, and create preference files.
- static const unsigned int **ALL_READ_OK** = USER_READ_OK | SYSTEM_READ_OK | CORE_READ_OK
Set this to allow FLTK and applications to read preference files.
- static const unsigned int **ALL_WRITE_OK** = USER_WRITE_OK | SYSTEM_WRITE_OK | CORE_WRITE_OK
Set this to allow FLTK and applications to create and write preference files.
- static const unsigned int **APP_OK** = SYSTEM_OK | USER_OK
Set this if it is OK for applications to read, create, and write any kind of preference files.
- static const unsigned int **CORE_OK** = CORE_READ_OK | CORE_WRITE_OK
Set this if it is OK for FLTK to read, create, or write preference files.
- static const unsigned int **CORE_READ_OK** = 0x0010
Set this if it is OK for FLTK to read preference files.
- static const unsigned int **CORE_WRITE_OK** = 0x0020
Set this if it is OK for FLTK to create or write preference files.
- static const unsigned int **NONE** = 0x0000
Set this if no call to FI_Preferences shall access the file system.
- static const unsigned int **SYSTEM_OK** = SYSTEM_READ_OK | SYSTEM_WRITE_OK
Set this if it is OK for applications to read, create, and write system wide preference files.
- static const unsigned int **SYSTEM_READ_OK** = 0x0004
Set this if it is OK for applications to read system wide preference files.
- static const unsigned int **SYSTEM_WRITE_OK** = 0x0008
Set this if it is OK for applications to create and write system wide preference files.
- static const unsigned int **USER_OK** = USER_READ_OK | USER_WRITE_OK
Set this if it is OK for applications to read, create, and write user preference files.
- static const unsigned int **USER_READ_OK** = 0x0001
Set this if it is OK for applications to read user preference files.
- static const unsigned int **USER_WRITE_OK** = 0x0002
Set this if it is OK for applications to create and write user preference files.

Protected Attributes inherited from FI_Preferences

- Node * node
- RootNode * rootNode

11.101.1 Detailed Description

[Fl_Plugin_Manager](#) manages link-time and run-time plugin binaries.

See also

[Fl_Plugin](#)

11.101.2 Constructor & Destructor Documentation

11.101.2.1 ~Fl_Plugin_Manager()

```
Fl_Plugin_Manager::~Fl_Plugin_Manager ()
```

Remove the plugin manager.

Calling this does not remove the database itself or any plugins. It just removes the reference to the database.

11.101.3 Member Function Documentation

11.101.3.1 addPlugin()

```
Fl_Preferences::ID Fl_Plugin_Manager::addPlugin (
    const char * name,
    Fl_Plugin * plugin)
```

This function adds a new plugin to the database.

There is no need to call this function explicitly. Every [Fl_Plugin](#) constructor will call this function at initialization time.

11.101.3.2 load()

```
int Fl_Plugin_Manager::load (
    const char * filename) [static]
```

Load a module from disk.

A module must be a dynamically linkable file for the given operating system. When loading a module, its +init function will be called which in turn calls the constructor of all statically initialized [Fl_Plugin](#) classes and adds them to the database.

11.101.3.3 loadAll()

```
int Fl_Plugin_Manager::loadAll (
    const char * dirpath,
    const char * pattern = 0) [static]
```

Use this function to load a whole directory full of modules.

Parameters

<i>dirpath</i>	Pathname of a directory. It must end with the platform's directory separator character (i.e., '\ ' under Windows, '/' otherwise).
<i>pattern</i>	A filename pattern to catch all modules of interest in the targeted directory (e.g., "{*.so,*.dll,*.dylib}"), or NULL to catch all files in the directory.

11.101.3.4 removePlugin()

```
void Fl_Plugin_Manager::removePlugin (
    Fl_Preferences::ID id) [static]
```

Remove any plugin.

There is no need to call this function explicitly. Every [Fl_Plugin](#) destructor will call this function at destruction time.

The documentation for this class was generated from the following files:

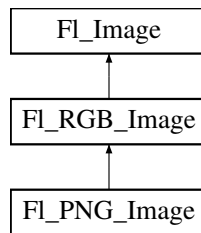
- [Fl_Plugin.H](#)
- [Fl_Preferences.cxx](#)

11.102 FI_PNG_Image Class Reference

The [FI_PNG_Image](#) class supports loading, caching, and drawing of Portable Network Graphics (PNG) image files.

```
#include <FI_PNG_Image.H>
```

Inheritance diagram for [FI_PNG_Image](#):



Public Member Functions

- [FI_PNG_Image](#) (const char *filename)
The constructor loads the named PNG image from the given png filename.
- [FI_PNG_Image](#) (const char *name_png, const unsigned char *buffer, int datasize)
Constructor that reads a PNG image from memory.

Public Member Functions inherited from [FI_RGB_Image](#)

- virtual [FI_SVG_Image](#) * [as_svg_image](#) ()
Returns whether an image is an [FI_SVG_Image](#) or not.
- int [cache_h](#) ()
- int [cache_w](#) ()
- void [color_average](#) ([FI_Color](#) c, float i) [FL_OVERRIDE](#)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [FI_Image](#) * [copy](#) () const
- [FI_Image](#) * [copy](#) (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void [desaturate](#) () [FL_OVERRIDE](#)
The [desaturate\(\)](#) method converts an image to grayscale.
- void [draw](#) (int X, int Y)
- void [draw](#) (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [FI_RGB_Image](#) (const [FI_Pixmap](#) *pxm, [FI_Color](#) bg=[FL_GRAY](#))
The constructor creates a new RGBA image from the specified [FI_Pixmap](#).
- [FI_RGB_Image](#) (const uchar *bits, int bits_length, int W, int H, int D, int LD)
The constructor creates a new image from the specified data.
- [FI_RGB_Image](#) (const uchar *bits, int W, int H, int D=3, int LD=0)
The constructor creates a new image from the specified data.
- void [label](#) ([FI_Menu_Item](#) *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.
- void [label](#) ([FI_Widget](#) *w) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a widget or menu item.
- virtual void [normalize](#) ()
Makes sure the object is fully initialized.
- void [uncache](#) () [FL_OVERRIDE](#)
If the image has been cached for display, delete the cache data.
- ~[FI_RGB_Image](#) () [FL_OVERRIDE](#)
The destructor frees all memory and server resources that are used by the image.

Public Member Functions inherited from FI_Image

- virtual class FI_Shared_Image * **as_shared_image** ()
Returns whether an image is an FI_Shared_Image or not.
- FI_Image * **copy** () const
Creates a copy of the image in the same size.
- int **count** () const
Returns the number of data values associated with the image.
- int **d** () const
Returns the image depth.
- const char *const * **data** () const
Returns a pointer to the current image data array.
- int **data_h** () const
Returns the height of the image data.
- int **data_w** () const
Returns the width of the image data.
- void **draw** (int X, int Y)
Draws the image to the current drawing surface.
- int **fail** () const
Returns a value that is not 0 if there is currently no image available.
- FI_Image (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int **h** () const
Returns the current image drawing height in FLTK units.
- void **inactive** ()
*The **inactive()** method calls **color_average(FI_BACKGROUND_COLOR, 0.33f)** to produce an image that appears grayed out.*
- int **ld** () const
Returns the current line data size in bytes.
- virtual void **release** ()
Releases an FI_Image - the same as 'delete this'.
- virtual void **scale** (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- int **w** () const
Returns the current image drawing width in FLTK units.
- virtual ~FI_Image ()
The destructor is a virtual method that frees all memory used by the image.

Friends

- class FI_ICO_Image

Additional Inherited Members

Static Public Member Functions inherited from FI_RGB_Image

- static size_t **max_size** ()
Returns the maximum allowed image size in bytes when creating an FI_RGB_Image object.
- static void **max_size** (size_t size)
Sets the maximum allowed image size in bytes when creating an FI_RGB_Image object.

Static Public Member Functions inherited from [FI_Image](#)

- static [FI_Labeltype](#) **define_FL_IMAGE_LABEL** ()
- static [FI_RGB_Scaling](#) **RGB_scaling** ()
Returns the currently used RGB image scaling method.
- static void [RGB_scaling](#) ([FI_RGB_Scaling](#))
Sets the RGB image scaling method used for copy(int, int).
- static [FI_RGB_Scaling](#) **scaling_algorithm** ()
Gets what algorithm is used when resizing a source image to draw it.
- static void [scaling_algorithm](#) ([FI_RGB_Scaling](#) algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Public Attributes inherited from [FI_RGB_Image](#)

- int **alloc_array**
If non-zero, the object's data array is delete[]'d when deleting the object.
- const [uchar](#) * **array**
Points to the start of the object's data array.

Static Public Attributes inherited from [FI_Image](#)

- static const int **ERR_FILE_ACCESS** = -2
- static const int **ERR_FORMAT** = -3
- static const int **ERR_MEMORY_ACCESS** = -4
- static const int **ERR_NO_IMAGE** = -1
- static bool **register_images_done** = false
True after [fl_register_images\(\)](#) was called, false before.

Protected Member Functions inherited from [FI_Image](#)

- void **d** (int D)
Sets the current image depth.
- void **data** (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void **draw_empty** (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int **draw_scaled** (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void **h** (int H)
Sets the height of the image data.
- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Static Protected Member Functions inherited from [FI_Image](#)

- static void **labeltype** (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void **measure** (const [FI_Label](#) *lo, int &lw, int &lh)

11.102.1 Detailed Description

The [FI_PNG_Image](#) class supports loading, caching, and drawing of Portable Network Graphics (PNG) image files. The class loads color-mapped and full-color images and handles color- and alpha-based transparency.

11.102.2 Constructor & Destructor Documentation

11.102.2.1 FI_PNG_Image() [1/2]

```
FI_PNG_Image::FI_PNG_Image (
    const char * filename)
```

The constructor loads the named PNG image from the given png filename.

The destructor frees all memory and server resources that are used by the image.

Use [FI_Image::fail\(\)](#) to check if [FI_PNG_Image](#) failed to load. [fail\(\)](#) returns `ERR_FILE_ACCESS` if the file could not be opened or read, `ERR_FORMAT` if the PNG format could not be decoded, and `ERR_NO_IMAGE` if the image could not be loaded for another reason.

Parameters

in	filename	Name of PNG file to read

11.102.2.2 FI_PNG_Image() [2/2]

```
FI_PNG_Image::FI_PNG_Image (
    const char * name_png,
    const unsigned char * buffer,
    int maxsize)
```

Constructor that reads a PNG image from memory.

Construct an image from a block of memory inside the application. Fluid offers "binary Data" chunks as a great way to add image data into the C++ source code. `name_png` can be NULL. If a name is given, the image is added to the list of shared images (see: [FI_Shared_Image](#)) and will be available by that name.

Parameters

name_png	A name given to this image or NULL
buffer	Pointer to the start of the PNG image in memory
maxsize	Size in bytes of the memory buffer containing the PNG image

The documentation for this class was generated from the following files:

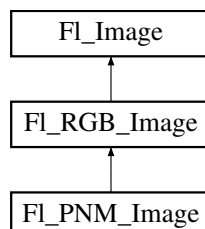
- [FI_PNG_Image.H](#)
- [FI_PNG_Image.cxx](#)

11.103 FI_PNM_Image Class Reference

The [FI_PNM_Image](#) class supports loading, caching, and drawing of Portable Anymap (PNM, PBM, PGM, PPM) image files.

```
#include <FI_PNM_Image.H>
```

Inheritance diagram for [FI_PNM_Image](#):



Public Member Functions

- [FI_PNM_Image](#) (const char *filename)
The constructor loads the named PNM image.

Public Member Functions inherited from [FI_RGB_Image](#)

- virtual [FI_SVG_Image](#) * [as_svg_image](#) ()
Returns whether an image is an [FI_SVG_Image](#) or not.
- int [cache_h](#) ()
- int [cache_w](#) ()
- void [color_average](#) ([FI_Color](#) c, float i) [FL_OVERRIDE](#)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [FI_Image](#) * [copy](#) () const
- [FI_Image](#) * [copy](#) (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void [desaturate](#) () [FL_OVERRIDE](#)
The [desaturate\(\)](#) method converts an image to grayscale.
- void [draw](#) (int X, int Y)
- void [draw](#) (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [FI_RGB_Image](#) (const [FI_Pixmap](#) *pxm, [FI_Color](#) bg=[FL_GRAY](#))
The constructor creates a new RGBA image from the specified [FI_Pixmap](#).
- [FI_RGB_Image](#) (const [uchar](#) *bits, int bits_length, int W, int H, int D, int LD)
The constructor creates a new image from the specified data.
- [FI_RGB_Image](#) (const [uchar](#) *bits, int W, int H, int D=3, int LD=0)
The constructor creates a new image from the specified data.
- void [label](#) ([FI_Menu_Item](#) *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.
- void [label](#) ([FI_Widget](#) *w) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a widget or menu item.
- virtual void [normalize](#) ()
Makes sure the object is fully initialized.
- void [uncache](#) () [FL_OVERRIDE](#)
If the image has been cached for display, delete the cache data.
- ~[FI_RGB_Image](#) () [FL_OVERRIDE](#)
The destructor frees all memory and server resources that are used by the image.

Public Member Functions inherited from [FI_Image](#)

- virtual class [FI_Shared_Image](#) * [as_shared_image](#) ()
Returns whether an image is an [FI_Shared_Image](#) or not.
- [FI_Image](#) * [copy](#) () const
Creates a copy of the image in the same size.
- int [count](#) () const
Returns the number of data values associated with the image.
- int [d](#) () const
Returns the image depth.
- const char *const * [data](#) () const
Returns a pointer to the current image data array.
- int [data_h](#) () const
Returns the height of the image data.
- int [data_w](#) () const
Returns the width of the image data.
- void [draw](#) (int X, int Y)
Draws the image to the current drawing surface.
- int [fail](#) () const

- Returns a value that is not 0 if there is currently no image available.*
- [FI_Image](#) (int W, int H, int D)
 - The constructor creates an empty image with the specified width, height, and depth.*
- int [h](#) () const
 - Returns the current image drawing height in FLTK units.*
- void [inactive](#) ()
 - The [inactive\(\)](#) method calls [color_average\(FI_BACKGROUND_COLOR, 0.33f\)](#) to produce an image that appears grayed out.*
- int [ld](#) () const
 - Returns the current line data size in bytes.*
- virtual void [release](#) ()
 - Releases an [FI_Image](#) - the same as 'delete this'.*
- virtual void [scale](#) (int width, int height, int proportional=1, int can_expand=0)
 - Sets the drawing size of the image.*
- int [w](#) () const
 - Returns the current image drawing width in FLTK units.*
- virtual [~FI_Image](#) ()
 - The destructor is a virtual method that frees all memory used by the image.*

Additional Inherited Members

Static Public Member Functions inherited from [FI_RGB_Image](#)

- static size_t [max_size](#) ()
 - Returns the maximum allowed image size in bytes when creating an [FI_RGB_Image](#) object.*
- static void [max_size](#) (size_t size)
 - Sets the maximum allowed image size in bytes when creating an [FI_RGB_Image](#) object.*

Static Public Member Functions inherited from [FI_Image](#)

- static [FI_Labeltype](#) [define_FL_IMAGE_LABEL](#) ()
- static [FI_RGB_Scaling](#) [RGB_scaling](#) ()
 - Returns the currently used RGB image scaling method.*
- static void [RGB_scaling](#) ([FI_RGB_Scaling](#))
 - Sets the RGB image scaling method used for copy(int, int).*
- static [FI_RGB_Scaling](#) [scaling_algorithm](#) ()
 - Gets what algorithm is used when resizing a source image to draw it.*
- static void [scaling_algorithm](#) ([FI_RGB_Scaling](#) algorithm)
 - Sets what algorithm is used when resizing a source image to draw it.*

Public Attributes inherited from [FI_RGB_Image](#)

- int [alloc_array](#)
 - If non-zero, the object's data array is delete[]'d when deleting the object.*
- const [uchar](#) * [array](#)
 - Points to the start of the object's data array.*

Static Public Attributes inherited from [FI_Image](#)

- static const int [ERR_FILE_ACCESS](#) = -2
- static const int [ERR_FORMAT](#) = -3
- static const int [ERR_MEMORY_ACCESS](#) = -4
- static const int [ERR_NO_IMAGE](#) = -1
- static bool [register_images_done](#) = false
 - True after [fl_register_images\(\)](#) was called, false before.*

Protected Member Functions inherited from [FI_Image](#)

- void **d** (int D)
Sets the current image depth.
- void **data** (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void **draw_empty** (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int **draw_scaled** (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void **h** (int H)
Sets the height of the image data.
- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Static Protected Member Functions inherited from [FI_Image](#)

- static void **labeltype** (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void **measure** (const [FI_Label](#) *lo, int &lw, int &lh)

11.103.1 Detailed Description

The [FI_PNM_Image](#) class supports loading, caching, and drawing of Portable Anymap (PNM, PBM, PGM, PPM) image files.

The class loads bitmap, grayscale, and full-color images in both ASCII and binary formats.

11.103.2 Constructor & Destructor Documentation

11.103.2.1 [FI_PNM_Image\(\)](#)

```
FI_PNM_Image::FI_PNM_Image (
    const char * filename)
```

The constructor loads the named PNM image.

The destructor frees all memory and server resources that are used by the image.

Use [FI_Image::fail\(\)](#) to check if [FI_PNM_Image](#) failed to load. [fail\(\)](#) returns `ERR_FILE_ACCESS` if the file could not be opened or read, `ERR_FORMAT` if the PNM format could not be decoded, and `ERR_NO_IMAGE` if the image could not be loaded for another reason.

Parameters

<code>in</code>	<code>filename</code>	a full path and name pointing to a valid jpeg file.

The documentation for this class was generated from the following files:

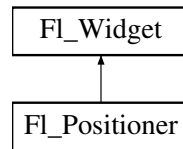
- [FI_PNM_Image.H](#)
- [FI_PNM_Image.cxx](#)

11.104 [FI_Positioner](#) Class Reference

This class is provided for Forms compatibility.

```
#include <FI_Positioner.H>
```

Inheritance diagram for [FI_Positioner](#):



Public Member Functions

- `Fl_Positioner` (int `x`, int `y`, int `w`, int `h`, const char *`l`=0)
 - Creates a new `Fl_Positioner` widget using the given position, size, and label string.*
- int `handle` (int) `FL_OVERRIDE`
 - Handles the specified event.*
- int `value` (double, double)
 - Returns the current position in `x` and `y`.*
- void `xbounds` (double, double)
 - Sets the `X` axis bounds.*
- double `xmaximum` () const
 - Gets the `X` axis maximum.*
- void `xmaximum` (double `a`)
 - Same as `xbounds(xminimum(), a)`*
- double `xminimum` () const
 - Gets the `X` axis minimum.*
- void `xminimum` (double `a`)
 - Same as `xbounds(a, xmaximum())`*
- void `xstep` (double `a`)
 - Sets the stepping value for the `X` axis.*
- double `xvalue` () const
 - Gets the `X` axis coordinate.*
- int `xvalue` (double)
 - Sets the `X` axis coordinate.*
- void `ybounds` (double, double)
 - Sets the `Y` axis bounds.*
- double `ymaximum` () const
 - Gets the `Y` axis maximum.*
- void `ymaximum` (double `a`)
 - Same as `ybounds(yminimum(), a)`*
- double `yminimum` () const
 - Gets the `Y` axis minimum.*
- void `yminimum` (double `a`)
 - Same as `ybounds(a, ymaximum())`*
- void `ystep` (double `a`)
 - Sets the stepping value for the `Y` axis.*
- double `yvalue` () const
 - Gets the `Y` axis coordinate.*
- int `yvalue` (double)
 - Sets the `Y` axis coordinate.*

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color color` () const

Gets the background color of the widget.
- void `color` (FL_Color bg)

Sets the background color of the widget.
- void `color` (FL_Color bg, FL_Color sel)

Sets the background and selection color of the widget.
- `FL_Color color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const FL_Widget *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([FL_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.
- `void draw (int, int, int, int)`
- `int handle (int, int, int, int, int)`

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
 ,
[OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
 ,
[MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
 ,
[GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
 = 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.104.1 Detailed Description

This class is provided for Forms compatibility.

It provides 2D input. It would be useful if this could be put atop another widget so that the crosshairs are on top, but this is not implemented. The color of the crosshairs is [selection_color\(\)](#).

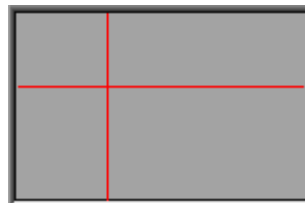


Figure 11.35 [Fl_Positioner](#)

11.104.2 Constructor & Destructor Documentation

11.104.2.1 [Fl_Positioner\(\)](#)

```
Fl_Positioner::Fl_Positioner (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Positioner](#) widget using the given position, size, and label string. The default boxtype is [FL_NO_BOX](#).

11.104.3 Member Function Documentation

11.104.3.1 [draw\(\)](#)

```
void Fl_Positioner::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.104.3.2 handle()

```
int Fl_Positioner::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited handle() method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your handle() method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the handle() method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

The documentation for this class was generated from the following files:

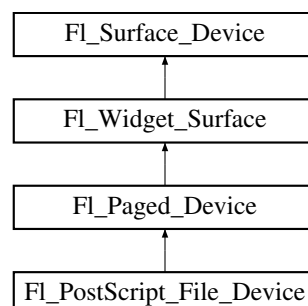
- Fl_Positioner.H
- Fl_Positioner.cxx

11.105 Fl_PostScript_File_Device Class Reference

To send graphical output to a PostScript file.

```
#include <Fl_PostScript.H>
```

Inheritance diagram for Fl_PostScript_File_Device:



Public Member Functions

- int [begin_job](#) (FILE *ps_output, int pagecount=0, enum [Fl_Paged_Device::Page_Format](#) format=[Fl_Paged_Device::A4](#), enum [Fl_Paged_Device::Page_Layout](#) layout=[Fl_Paged_Device::PORTRAIT](#))

- Begins the session where all graphics requests will go to FILE pointer.*

 - int [begin_job](#) (int pagecount, int *from, int *to, char **perr_message) [FL_OVERRIDE](#)

Don't use with this class.
- int [begin_job](#) (int pagecount=0, enum [FI_Paged_Device::Page_Format](#) format=[FI_Paged_Device::A4](#), enum [FI_Paged_Device::Page_Layout](#) layout=[FI_Paged_Device::PORTRAIT](#))

Begins the session where all graphics requests will go to a local PostScript file.
- int [begin_page](#) (void) [FL_OVERRIDE](#)

Begins a new printed page.
- void [close_command](#) ([FI_PostScript_Close_Command](#) cmd)

Sets the function [end_job\(\)](#) calls to close the [file\(\)](#)
- void [end_current](#) () [FL_OVERRIDE](#)

FLTK calls this each time a surface ceases to be the current drawing surface.
- void [end_job](#) (void) [FL_OVERRIDE](#)

Completes all PostScript output.
- int [end_page](#) (void) [FL_OVERRIDE](#)

To be called at the end of each page.
- FILE * [file](#) ()

Returns the underlying FILE receiving all PostScript data.*
- [FI_PostScript_File_Device](#) ()

The constructor.
- void [margins](#) (int *left, int *top, int *right, int *bottom) [FL_OVERRIDE](#)

Computes the dimensions of margins that lie between the printable page area and the full page.
- void [origin](#) (int *x, int *y) [FL_OVERRIDE](#)

Computes the coordinates of the current origin of graphics functions.
- void [origin](#) (int x, int y) [FL_OVERRIDE](#)

Sets the position of the origin of graphics in the drawable part of the drawing surface.
- int [printable_rect](#) (int *w, int *h) [FL_OVERRIDE](#)

Computes the width and height of the drawable area of the drawing surface.
- void [rotate](#) (float angle) [FL_OVERRIDE](#)

Rotates the graphics operations relatively to paper.
- void [scale](#) (float scale_x, float scale_y=0.) [FL_OVERRIDE](#)

Changes the scaling of page coordinates.
- void [set_current](#) () [FL_OVERRIDE](#)

Make this surface the current drawing surface.
- int [start_job](#) (FILE *ps_output, int pagecount=0, enum [FI_Paged_Device::Page_Format](#) format=[FI_Paged_Device::A4](#), enum [FI_Paged_Device::Page_Layout](#) layout=[FI_Paged_Device::PORTRAIT](#))

Synonym of [begin_job\(\)](#).
- int [start_job](#) (int pagecount=0, enum [FI_Paged_Device::Page_Format](#) format=[FI_Paged_Device::A4](#), enum [FI_Paged_Device::Page_Layout](#) layout=[FI_Paged_Device::PORTRAIT](#))

Synonym of [begin_job\(\)](#).
- void [translate](#) (int x, int y) [FL_OVERRIDE](#)

Translates the current graphics origin accounting for the current rotation.
- void [untranslate](#) (void) [FL_OVERRIDE](#)

Undoes the effect of a previous [translate\(\)](#) call.
- ~[FI_PostScript_File_Device](#) ()

The destructor.

Public Member Functions inherited from FI_Paged_Device

- void **print_widget** ([FI_Widget](#) *widget, int delta_x=0, int delta_y=0)
Synonym of [draw\(FI_Widget, int, int\)](#)*
- void **print_window** ([FI_Window](#) *win, int x_off=0, int y_off=0)
Synonym of [draw_decorated_window\(FI_Window, int, int\)](#)*
- int **start_job** (int pagecount=0, int *frompage=NULL, int *topage=NULL, char **perr_message=NULL)
*Synonym of [begin_job\(int pagecount, int *frompage, int *topage, char **perr_message\)](#).*
- int **start_page** ()
Synonym of [begin_page\(\)](#).
- virtual ~**FI_Paged_Device** ()
The destructor.

Public Member Functions inherited from FI_Widget_Surface

- void **draw** ([FI_Widget](#) *widget, int delta_x=0, int delta_y=0)
Draws the widget on the drawing surface.
- void **draw_decorated_window** ([FI_Window](#) *win, int x_offset=0, int y_offset=0)
Draws a window with its title bar and frame if any.
- void **print_window_part** ([FI_Window](#) *win, int x, int y, int w, int h, int delta_x=0, int delta_y=0)
Draws a rectangular part of an on-screen window.

Public Member Functions inherited from FI_Surface_Device

- [FI_Graphics_Driver](#) * **driver** ()
Returns the graphics driver of this drawing surface.
- virtual bool **is_current** ()
Is this surface the current drawing surface?
- virtual ~**FI_Surface_Device** ()
The destructor.

Static Public Attributes

- static const char * **file_chooser_title**
Label of the PostScript file chooser window.

Static Public Attributes inherited from FI_Paged_Device

- static const [page_format](#) **page_formats** [[NO_PAGE_FORMATS](#)]
width, height and name of all elements of the enum [Page_Format](#).

Protected Member Functions

- [FI_PostScript_Graphics_Driver](#) * **driver** ()
Returns the PostScript driver of this drawing surface.

Protected Member Functions inherited from FI_Paged_Device

- **FI_Paged_Device** ()
The constructor.

Protected Member Functions inherited from FI_Widget_Surface

- [FI_Widget_Surface](#) ([FI_Graphics_Driver](#) *d)
The constructor.

Protected Member Functions inherited from [FI_Surface_Device](#)

- void **driver** ([FI_Graphics_Driver](#) *graphics_driver)
Sets the graphics driver of this drawing surface.
- **FI_Surface_Device** ([FI_Graphics_Driver](#) *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

Additional Inherited Members

Public Types inherited from [FI_Paged_Device](#)

- enum [Page_Format](#) {
A0 , A1 , A2 , A3 ,
A4 , A5 , A6 , A7 ,
A8 , A9 , B0 , B1 ,
B2 , B3 , B4 , B5 ,
B6 , B7 , B8 , B9 ,
B10 , C5E , DLE , EXECUTIVE ,
FOLIO , LEDGER , LEGAL , LETTER ,
TABLOID , ENVELOPE , MEDIA = 0x1000 }
Possible page formats.
- enum [Page_Layout](#) { PORTRAIT = 0 , LANDSCAPE = 0x100 , REVERSED = 0x200 , ORIENTATION = 0x300 }
Possible page layouts.

Static Public Member Functions inherited from [FI_Surface_Device](#)

- static [FI_Surface_Device](#) * pop_current ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void push_current ([FI_Surface_Device](#) *new_current)
Pushes new_current on top of the stack of current drawing surfaces, and makes it current.
- static [FI_Surface_Device](#) * surface ()
The current drawing surface.

Protected Attributes inherited from [FI_Widget_Surface](#)

- int **x_offset**
horizontal offset to the origin of graphics coordinates
- int **y_offset**
vertical offset to the origin of graphics coordinates

11.105.1 Detailed Description

To send graphical output to a PostScript file.

This class is used exactly as the [FI_Printer](#) class except for the [begin_job\(\)](#) call, two variants of which are usable and allow to specify what page format and layout are desired.

Processing of text: Text uses vectorial fonts under the X11 + Pango and the Wayland platforms. With other platforms, only text restricted to the Latin alphabet (and a few other characters listed in the table below) and to FLTK standard fonts is vectorized. All other unicode characters or all other fonts (FL_FREE_FONT and above) are output as a bitmap. FLTK standard fonts are output using the corresponding PostScript standard fonts. The latin alphabet means all unicode characters between U+0020 and U+017F, or, in other words, the ASCII, Latin-1 Supplement and Latin Extended-A charts.

Char	Codepoint	Name	Char	Codepoint	Name	Char	Codepoint	Name
<i>f</i>	U+0192	florin	,	U+201A	quotesinglbase	™	U+2122	trademark
^	U+02C6	circumflex	“	U+201C	quotedblleft	∂	U+2202	partialdiff
ˇ	U+02C7	caron	”	U+201D	quotedblright	Δ	U+2206	Delta
˘	U+02D8	breve	„	U+201E	quotedblbase	Σ	U+2211	summation
˙	U+02D9	dotaccent	†	U+2020	dagger	√	U+221A	radical
°	U+02DA	ring	‡	U+2021	daggerdbl	∞	U+221E	infinity
˛	U+02DB	ogonek	•	U+2022	bullet	≠	U+2260	notequal
˜	U+02DC	tilde	...	U+2026	ellipsis	≤	U+2264	lessequal
˝	U+02DD	hungarumlaut	‰	U+2030	perthousand	≥	U+2265	greaterequal
—	U+2013	endash	‹	U+2039	guilsinglleft	◊	U+25CA	lozenge
—	U+2014	emdash	›	U+203A	guilsinglright	fi	U+FB01	fi
‘	U+2018	quoteleft	/	U+2044	fraction	fl	U+FB02	fl
’	U+2019	quoteright	€	U+20AC	Euro	🍏	U+F8FF	apple (macOS only)

Figure 11.36 Extra characters supported by standard PostScript fonts

Processing of transparent [Fl_RGB_Image](#) objects: Under the X11 + Pango and the Wayland platforms, these objects are output with their exact transparency. With other platforms, these objects are drawn blended to white color. Class [Fl_EPS_File_Surface](#) 's constructor allows to set another background color for blending.

11.105.2 Member Function Documentation

11.105.2.1 `begin_job()` [1/3]

```
int Fl_PostScript_File_Device::begin_job (
    FILE * ps_output,
    int pagecount = 0,
    enum Fl_Paged_Device::Page_Format format = Fl_Paged_Device::A4,
    enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT)
```

Begins the session where all graphics requests will go to FILE pointer.

This member function prevents `end_job()` from closing `ps_output`, so the user can check with `ferror(ps_output)` for output errors.

Parameters

<i>ps_output</i>	A writable FILE pointer that will receive PostScript output and that should not be closed until after end_job() has been called.
<i>pagecount</i>	The total number of pages to be created. Use 0 if this number is unknown when this function is called.
<i>format</i>	Desired page format.
<i>layout</i>	Desired page layout.

Returns

always 0.

11.105.2.2 `begin_job()` [2/3]

```
int Fl_PostScript_File_Device::begin_job (
    int pagecount,
    int * from,
```

```
int * to,
char ** perr_message) [virtual]
```

Don't use with this class.

Reimplemented from [Fl_Paged_Device](#).

11.105.2.3 begin_job() [3/3]

```
int Fl_PostScript_File_Device::begin_job (
    int pagecount = 0,
    enum Fl_Paged_Device::Page_Format format = Fl_Paged_Device::A4,
    enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT)
```

Begins the session where all graphics requests will go to a local PostScript file.

Opens a file dialog to select an output PostScript file. This member function makes [end_job\(\)](#) close the resulting PostScript file and display an alert message with [fl_alert\(\)](#) in case of any output error.

Parameters

<i>pagecount</i>	The total number of pages to be created. Use 0 if this number is unknown when this function is called.
<i>format</i>	Desired page format.
<i>layout</i>	Desired page layout.

Returns

0 if OK, 1 if user cancelled the file dialog, 2 if fopen failed on user-selected output file.

11.105.2.4 begin_page()

```
int Fl_PostScript_File_Device::begin_page (
    void ) [virtual]
```

Begins a new printed page.

The page coordinates are initially in points, i.e., 1/72 inch, and with origin at the top left of the printable page area. This function also makes this surface the current drawing surface with [Fl_Surface_Device::push_current\(\)](#).

Note

[begin_page\(\)](#) calls [Fl_Surface_Device::push_current\(\)](#) and leaves this device as the active surface. If any calls between [begin_page\(\)](#) and [end_page\(\)](#) open dialog boxes or will otherwise draw into FLTK windows, those calls must be put between a call to [Fl_Surface_Device::pop_current\(\)](#) and a call to [Fl_Surface_Device::push_current\(\)](#), or the content of the dialog box will be rendered to the printer instead of the screen.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Paged_Device](#).

11.105.2.5 end_current()

```
void Fl_PostScript_File_Device::end_current () [virtual]
```

FLTK calls this each time a surface ceases to be the current drawing surface.

This member function is mostly of interest to developers of new [Fl_Surface_Device](#) derived classes. It allows to perform surface-specific operations necessary when this surface ceases to be current. Each implementation should end with a call to [Fl_Surface_Device::end_current\(\)](#).

Reimplemented from [Fl_Surface_Device](#).

11.105.2.6 end_job()

```
void Fl_PostScript_File_Device::end_job (
    void ) [virtual]
```

Completes all PostScript output.

This also closes with `fclose()` the underlying [file\(\)](#) unless [close_command\(\)](#) was used to set another function.

Reimplemented from [Fl_Paged_Device](#).

11.105.2.7 end_page()

```
int Fl_PostScript_File_Device::end_page (
    void ) [virtual]
```

To be called at the end of each page.

This function also stops this surface from being the current drawing surface with [Fl_Surface_Device::pop_current\(\)](#).

Note

[end_page\(\)](#) calls [Fl_Surface_Device::pop_current\(\)](#). If any calls between [begin_page\(\)](#) and [end_page\(\)](#) open dialog boxes or will otherwise draw into FLTK windows, those calls must be put between a call to [Fl_Surface_Device::pop_current\(\)](#) and a call to [Fl_Surface_Device::push_current\(\)](#).

Returns

0 if OK, non-zero if any error.

Reimplemented from [Fl_Paged_Device](#).

11.105.2.8 margins()

```
void Fl_PostScript_File_Device::margins (
    int * left,
    int * top,
    int * right,
    int * bottom) [virtual]
```

Computes the dimensions of margins that lie between the printable page area and the full page.

Values are in the same unit as that used by FLTK drawing functions. They are changed by [scale\(\)](#) calls.

Parameters

out	<i>left</i>	If non-null, *left is set to the left margin size.
out	<i>top</i>	If non-null, *top is set to the top margin size.
out	<i>right</i>	If non-null, *right is set to the right margin size.
out	<i>bottom</i>	If non-null, *bottom is set to the bottom margin size.

Reimplemented from [Fl_Paged_Device](#).

11.105.2.9 origin() [1/2]

```
void Fl_PostScript_File_Device::origin (
    int * x,
    int * y) [virtual]
```

Computes the coordinates of the current origin of graphics functions.

Parameters

out	<i>x,y</i>	If non-null, *x and *y are set to the horizontal and vertical coordinates of the graphics origin.

Reimplemented from [Fl_Widget_Surface](#).

11.105.2.10 origin() [2/2]

```
void Fl_PostScript_File_Device::origin (
    int x,
    int y) [virtual]
```

Sets the position of the origin of graphics in the drawable part of the drawing surface.

Arguments should be expressed relatively to the result of a previous [printable_rect\(\)](#) call. That is, `printable_rect(&w, &h); origin(w/2, 0);` sets the graphics origin at the top center of the drawable area. Successive [origin\(\)](#) calls don't combine their effects. [Origin\(\)](#) calls are not affected by [rotate\(\)](#) calls (for classes derived from [Fl_Paged_Device](#)).

Parameters

<i>in</i>	<i>x,y</i>	Horizontal and vertical positions in the drawing surface of the desired origin of graphics.

Reimplemented from [Fl_Widget_Surface](#).

11.105.2.11 printable_rect()

```
int Fl_PostScript_File_Device::printable_rect (
    int * w,
    int * h) [virtual]
```

Computes the width and height of the drawable area of the drawing surface.

Values are in the same unit as that used by FLTK drawing functions and are unchanged by calls to [origin\(\)](#). If the object is derived from class [Fl_Paged_Device](#), values account for the user-selected paper type and print orientation and are changed by [scale\(\)](#) calls.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Widget_Surface](#).

11.105.2.12 rotate()

```
void Fl_PostScript_File_Device::rotate (
    float angle) [virtual]
```

Rotates the graphics operations relatively to paper.

The rotation is centered on the current graphics origin. Successive [rotate\(\)](#) calls don't combine their effects.

Parameters

<i>angle</i>	Rotation angle in counter-clockwise degrees.

Reimplemented from [Fl_Paged_Device](#).

11.105.2.13 scale()

```
void Fl_PostScript_File_Device::scale (
    float scale_x,
    float scale_y = 0.) [virtual]
```

Changes the scaling of page coordinates.

This function also resets the origin of graphics functions at top left of printable page area. After a [scale\(\)](#) call, do a [printable_rect\(\)](#) call to get the new dimensions of the printable page area. Successive [scale\(\)](#) calls don't combine their effects.

Parameters

<i>scale_x</i>	Horizontal dimensions of plot are multiplied by this quantity.

<code>scale_↵ _y</code>	Same as above, vertically. The value 0. is equivalent to setting <code>scale_y = scale_x</code> . Thus, <code>scale(factor)</code> ; is equivalent to <code>scale(factor, factor)</code> ;

Reimplemented from [Fl_Paged_Device](#).

11.105.2.14 set_current()

```
void Fl_PostScript_File_Device::set_current (
    void ) [virtual]
```

Make this surface the current drawing surface.

This surface will receive all future graphics requests.

Since FLTK 1.4.0 the preferred API to change the current drawing surface is [Fl_Surface_Device::push_current\(\)](#) / [Fl_Surface_Device::pop_current\(\)](#).

Note

It is recommended to use this function only as follows :

- The current drawing surface is the display;
- make current another surface, e.g., an [Fl_Printer](#) or an [Fl_Image_Surface](#) object, calling [set_current\(\)](#) on this object;
- draw to that surface;
- make the display current again with [Fl_Display_Device::display_device\(\)->set_current\(\)](#);
don't do any other call to [set_current\(\)](#) before this one.

Other scenarios of drawing surface changes should be performed via [Fl_Surface_Device::push_current\(\)](#) and [Fl_Surface_Device::pop_current\(\)](#).

Reimplemented from [Fl_Surface_Device](#).

11.105.2.15 start_job() [1/2]

```
int Fl_PostScript_File_Device::start_job (
    FILE * ps_output,
    int pagecount = 0,
    enum Fl_Paged_Device::Page_Format format = Fl_Paged_Device::A4,
    enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT) [inline]
```

Synonym of [begin_job\(\)](#).

For API compatibility with FLTK 1.3.x

11.105.2.16 start_job() [2/2]

```
int Fl_PostScript_File_Device::start_job (
    int pagecount = 0,
    enum Fl_Paged_Device::Page_Format format = Fl_Paged_Device::A4,
    enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT) [inline]
```

Synonym of [begin_job\(\)](#).

For API compatibility with FLTK 1.3.x

11.105.2.17 translate()

```
void Fl_PostScript_File_Device::translate (
    int x,
    int y) [virtual]
```

Translates the current graphics origin accounting for the current rotation.

Each [translate\(\)](#) call must be matched by an [untranslate\(\)](#) call. Successive [translate\(\)](#) calls add up their effects.

Reimplemented from [Fl_Widget_Surface](#).

11.105.2.18 untranslate()

```
void Fl_PostScript_File_Device::untranslate (
    void ) [virtual]
```

Undoes the effect of a previous [translate\(\)](#) call.

Reimplemented from [Fl_Widget_Surface](#).

The documentation for this class was generated from the following file:

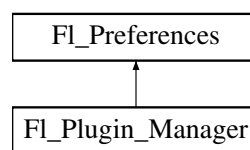
- [Fl_PostScript.H](#)

11.106 Fl_Preferences Class Reference

[Fl_Preferences](#) store user settings between application starts.

```
#include <Fl_Preferences.H>
```

Inheritance diagram for [Fl_Preferences](#):



Classes

- struct [Entry](#)
- class [Name](#)
 - 'Name' provides a simple method to create numerical or more complex procedural names for entries and groups on the fly.*
- class [Node](#)
- class [RootNode](#)

Public Types

- typedef void * [ID](#)
 - Every Fl_Preferences-Group has a unique ID.*
- enum [Root](#) {
 - [UNKNOWN_ROOT_TYPE](#) = -1 , [SYSTEM](#) = 0 , [USER](#) , [MEMORY](#) ,
 - [ROOT_MASK](#) = 0x00FF , [CORE](#) = 0x0100 , [C_LOCALE](#) = 0x1000 , [CLEAR](#) = 0x2000 ,
 - [SYSTEM_L](#) = SYSTEM | C_LOCALE , [USER_L](#) = USER | C_LOCALE , [CORE_SYSTEM_L](#) = CORE |
 - [SYSTEM_L](#) , [CORE_USER_L](#) = CORE | USER_L ,
 - [CORE_SYSTEM](#) = CORE | SYSTEM , [CORE_USER](#) = CORE | USER }
 - Define the scope of the preferences.*

Public Member Functions

- char [clear](#) ()
 - Delete all groups and all entries.*
- char [delete_all_entries](#) ()
 - Delete all entries.*
- char [delete_all_groups](#) ()
 - Delete all groups.*
- char [delete_entry](#) (const char *entry)
 - Deletes a single name/value pair.*
- char [delete_group](#) (const char *group)
 - Deletes a group.*
- int [dirty](#) ()

- Check if there were changes to the database that need to be written to disk.*

 - `int entries ()`
Returns the number of entries (name/value pairs) in a group.
 - `const char * entry (int index)`
Returns the name of an entry.
 - `char entry_exists (const char *key)`
Returns non-zero if an entry with this name exists.
 - `Root filename (char *buffer, size_t buffer_size)`
Return the file name and path to the preference file.
 - `FI_Preferences (const char *path, const char *vendor, const char *application)`
Deprecated: Use this constructor to create or read a preference file at an arbitrary position in the file system.
 - `FI_Preferences (const char *path, const char *vendor, const char *application, Root flags)`
Use this constructor to create or read a preference file at an arbitrary position in the file system.
 - `FI_Preferences (const FI_Preferences &)`
Create another reference to a Preferences group.
 - `FI_Preferences (FI_Preferences &parent, const char *group)`
Generate or read a new group of entries within another group.
 - `FI_Preferences (FI_Preferences &parent, int groupIndex)`
Open a child group using a given index.
 - `FI_Preferences (FI_Preferences *parent, const char *group)`
Create or access a group of preferences using a name.
 - `FI_Preferences (FI_Preferences *parent, int groupIndex)`
 - `FI_Preferences (ID id)`
Create a new dataset access point using a dataset ID.
 - `FI_Preferences (Root root, const char *vendor, const char *application)`
The constructor creates a group that manages key/value pairs and child groups.
 - `int flush ()`
Writes preferences to disk if they were modified.
 - `char get (const char *entry, char *&value, const char *defaultValue)`
Reads an entry from the group.
 - `char get (const char *entry, char *value, const char *defaultValue, int maxSize)`
Reads an entry from the group.
 - `char get (const char *entry, double &value, double defaultValue)`
Reads an entry from the group.
 - `char get (const char *entry, float &value, float defaultValue)`
Reads an entry from the group.
 - `char get (const char *entry, int &value, int defaultValue)`
Reads an entry from the group.
 - `char get (const char *entry, std::string &value, const std::string &defaultValue)`
Reads an entry from the group.
 - `char get (const char *entry, void *&value, const void *defaultValue, int defaultSize)`
Reads an entry from the group.
 - `char get (const char *entry, void *value, const void *defaultValue, int defaultSize, int *size)`
Reads a binary entry from the group, encoded in hexadecimal blocks.
 - `char get (const char *entry, void *value, const void *defaultValue, int defaultSize, int maxSize)`
Reads a binary entry from the group, encoded in hexadecimal blocks.
 - `char get_userdata_path (char *path, int pathlen)`
Creates a path that is related to the preference file and that is usable for additional application data.
 - `const char * group (int num_group)`
Returns the name of the Nth (num_group) group.
 - `char group_exists (const char *key)`

- Returns non-zero if a group with this name exists.*

 - int **groups** ()

Returns the number of groups that are contained within a group.

- ID **id** ()
- Return an ID that can later be reused to open more references to this dataset.*

 - const char * **name** ()

Return the name of this entry.

- const char * **path** ()
- Return the full path to this entry.*

 - char **set** (const char ***entry**, const char *value)

Sets an entry (name/value pair).

- char **set** (const char ***entry**, const std::string &value)
- Sets an entry (name/value pair).*

 - char **set** (const char ***entry**, const void *value, int **size**)

Sets an entry (name/value pair).

- char **set** (const char ***entry**, double value)
- Sets an entry (name/value pair).*

 - char **set** (const char ***entry**, double value, int precision)

Sets an entry (name/value pair).

- char **set** (const char ***entry**, float value)
- Sets an entry (name/value pair).*

 - char **set** (const char ***entry**, float value, int precision)

Sets an entry (name/value pair).

- char **set** (const char ***entry**, int value)
- Sets an entry (name/value pair).*

 - int **size** (const char ***entry**)

Returns the size of the value part of an entry.

- virtual ~**FL_Preferences** ()
- The destructor removes allocated resources.*

Static Public Member Functions

- static unsigned int **file_access** ()
- Return the current file access permissions for the FLTK preferences system.*

 - static void **file_access** (unsigned int flags)

Tell the FLTK preferences system which files in the file system it may read, create, or write.

- static **Root filename** (char *buffer, size_t buffer_size, **Root** root, const char *vendor, const char *application)
- Determine the file name and path to preferences that would be opened with these parameters.*

 - static const char * **new_UUID** ()

Returns a UUID as generated by the system.

- static char **remove** (ID id_)
- Remove the group with this ID from a database.*

Static Public Attributes

- static const unsigned int **ALL** = **ALL_READ_OK** | **ALL_WRITE_OK**
- Set this to give FLTK and applications permission to read, write, and create preference files.*

 - static const unsigned int **ALL_READ_OK** = **USER_READ_OK** | **SYSTEM_READ_OK** | **CORE_READ_OK**

Set this to allow FLTK and applications to read preference files.

- static const unsigned int **ALL_WRITE_OK** = **USER_WRITE_OK** | **SYSTEM_WRITE_OK** | **CORE_WRITE_OK**
- Set this to allow FLTK and applications to create and write preference files.*

 - static const unsigned int **APP_OK** = **SYSTEM_OK** | **USER_OK**

- Set this if it is OK for applications to read, create, and write any kind of preference files.*
- static const unsigned int **CORE_OK** = **CORE_READ_OK** | **CORE_WRITE_OK**
- Set this if it is OK for FLTK to read, create, or write preference files.*
- static const unsigned int **CORE_READ_OK** = 0x0010
- Set this if it is OK for FLTK to read preference files.*
- static const unsigned int **CORE_WRITE_OK** = 0x0020
- Set this if it is OK for FLTK to create or write preference files.*
- static const unsigned int **NONE** = 0x0000
- Set this if no call to [FI_Preferences](#) shall access the file system.*
- static const unsigned int **SYSTEM_OK** = **SYSTEM_READ_OK** | **SYSTEM_WRITE_OK**
- Set this if it is OK for applications to read, create, and write system wide preference files.*
- static const unsigned int **SYSTEM_READ_OK** = 0x0004
- Set this if it is OK for applications to read system wide preference files.*
- static const unsigned int **SYSTEM_WRITE_OK** = 0x0008
- Set this if it is OK for applications to create and write system wide preference files.*
- static const unsigned int **USER_OK** = **USER_READ_OK** | **USER_WRITE_OK**
- Set this if it is OK for applications to read, create, and write user preference files.*
- static const unsigned int **USER_READ_OK** = 0x0001
- Set this if it is OK for applications to read user preference files.*
- static const unsigned int **USER_WRITE_OK** = 0x0002
- Set this if it is OK for applications to create and write user preference files.*

Protected Attributes

- [Node](#) * **node**
- [RootNode](#) * **rootNode**

Friends

- class **Node**
- class **RootNode**

11.106.1 Detailed Description

[FI_Preferences](#) store user settings between application starts.

FLTK Preferences are similar to the Registry on Windows and Preferences on MacOS, providing a simple method to store customizable user settings between application launches. A typical use is storing the last window position or a history of previously used documents.

Preferences are organized in a hierarchy of groups. Every group can contain more groups and any number of key/value pairs. Keys can be text strings containing ASCII letters, digits, periods, and underscores. Forward slashes in a key name are treated as subgroups, i.e. the key 'window/width' would actually refer to the key 'width' inside the group 'window'.

Keys have a unique name within their group. A value can be any string including control characters 0x00 to 0x1f, 0x7f, and UTF-8 octets.

Several methods allow setting and getting numerical values and binary data.

Preferences files are the same across platforms. User comments in preference files are preserved. Filenames are unique for each application by using a vendor/application naming scheme. The developer app must provide default values for all entries to ensure proper operation should preferences be corrupted or not yet exist.

Note

The format of preferences files is not part of the FLTK specification and intentionally undocumented. The only valid way to read or write prefs files is via the API from your app. The fact that the current implementation looks like human-readable text is purely coincidental and may change at any time. Preferences files are not meant to be created or edited "by hand."

FLTK preferences are not meant to replace a fully featured database. No merging of data takes place. If several instances of an app access the same database at the same time, only the most recent changes will persist. Preferences should not be used to store document data. The .prefs file should be kept small for performance reasons. One application can have multiple preferences files. Extensive binary data however should be stored in separate files: see [Fl_Preferences::get_userdata_path\(\)](#).

[Fl_Preferences](#) are not thread-safe. They can temporarily change the locale on some platforms during read and write access, which also changes it temporarily in other threads of the same app.

Typically a preferences database is read at startup, and then reopened and written at app shutdown:

```
int appWindowWidth, appWindowHeight;

void launch() {
    Fl_Preferences app(Fl_Preferences::USER_L, "matthiasm.com", "hello");
    // 'app' constructor will be called, reading data from .prefs file
    Fl_Preferences window(app, "window");
    window.get("width", appWindowWidth, 800);
    window.get("height", appWindowHeight, 600);
    // 'app' destructor will be called. This will write data to the
    // .prefs file if any preferences were changed or added
}

void quit() {
    Fl_Preferences app(Fl_Preferences::USER_L, "matthiasm.com", "hello");
    Fl_Preferences window(app, "window");
    window.set("width", appWindowWidth);
    window.set("height", appWindowHeight);
}
```

See also

[Fl_Preferences::Fl_Preferences\(Root root, const char *vendor, const char *application\)](#)

As a special case, [Fl_Preferences](#) can be memory mapped and not be associated with a file on disk. See [Fl_Preferences::Fl_Preferences\(Fl_Preferences *parent, const char *group\)](#) and [Fl_Preferences::MEMORY](#) for more details on memory mapped preferences.

Note

Starting with FLTK 1.3, preferences databases are expected to be in UTF-8 encoding. Previous databases were stored in the current character set or code page which renders them incompatible for text entries using international characters.

Starting with FLTK 1.4, searching a valid path to store the preference files has changed slightly. Please see [Fl_Preferences::Fl_Preferences\(Root, const char*, const char*\)](#) for details.

Starting with FLTK 1.4, preference files should be created with `SYSTEM_L` or `USER_L` to be interchangeable between computers with differing locale settings. The legacy modes, `LOCAL` and `SYSTEM`, will read and write floating point values using the decimal point of the current locale. As a result, a fp-value would be written '3,1415' on a German machine, and would be read back as '3.0' on a US machine because the comma would not be recognized as an alternative decimal point.

11.106.2 Member Typedef Documentation

11.106.2.1 ID

void* [Fl_Preferences::ID](#)

Every [Fl_Preferences-Group](#) has a unique ID.

ID's can be retrieved from an [Fl_Preferences-Group](#) and can then be used to create more [Fl_Preference](#) references to the same data set, as long as the database remains open.

11.106.3 Member Enumeration Documentation

11.106.3.1 Root

enum [Fl_Preferences::Root](#)

Define the scope of the preferences.

Enumerator

UNKNOWN_ROOT_TYPE	Returned if storage could not be determined.

Enumerator

SYSTEM	Preferences are used system-wide. Deprecated, see SYSTEM_L.
USER	Preferences apply only to the current user. Deprecated, see USER_L.
MEMORY	Returned if querying memory mapped preferences.
ROOT_MASK	Mask for the values above.
CORE	OR'd by FLTK to read and write core library preferences and options.
C_LOCALE	This flag should always be set to ensure that floating point values are written and read correctly independently of the current locale.
CLEAR	Don't read a possibly existing database. Instead, start with an empty set of preferences.
SYSTEM_L	Preferences are used system-wide, locale independent.
USER_L	Preferences apply only to the current user, locale independent.
CORE_SYSTEM_L	Same as CORE SYSTEM C_LOCALE.
CORE_USER_L	Same as CORE USER C_LOCALE.
CORE_SYSTEM	Deprecated, same as CORE SYSTEM. Use CORE_SYSTEM_L instead.
CORE_USER	Deprecated, same as CORE USER. Use CORE_USER_L instead.

11.106.4 Constructor & Destructor Documentation

11.106.4.1 Fl_Preferences() [1/8]

```
Fl_Preferences::Fl_Preferences (
    Root root,
    const char * vendor,
    const char * application)
```

The constructor creates a group that manages key/value pairs and child groups.

Preferences can be stored per user using the root type `Fl_Preferences::USER_L`, or stored system-wide using `Fl_Preferences::SYSTEM_L`.

Groups and key/value pairs can be read and written randomly. Reading undefined values will return the default value. Writing undefined values will create all required groups and key/value pairs.

This constructor creates the *base* instance for all following entries and reads the database from disk into memory if it exists. The vendor argument is a unique text string identifying the development team or vendor of an application. A domain name or an EMail address (replacing the '@' with a '.') are great unique names, e.g. "research.matthiasm.com" or "fluid.fltk.org". The application argument can be the working title or final name of your application. Both vendor and application must be valid UNIX path segments as they become parts of the preference file path and may contain forward slashes to create deeper file structures.

Note

On **Windows**, the directory is constructed by querying the *Common AppData* or *AppData* key of the `Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders` registry entry. The filename and path is then constructed as `$(query)/$(vendor)/$(application).prefs`. If the query call fails, data will be stored in RAM only. It will be lost when the app exits.

In FLTK versions before 1.4.0, if querying the registry failed,

preferences would be written to `C:\FLTK\$(vendor)\$(application).prefs`.

Note

On **Linux**, the `USER` directory is constructed by reading `$HOME`. If `$HOME` is not set or not pointing to an existing directory, FLTK will check the path member of the `passwd` struct returned by `getpwuid(getuid())`. If all attempts fail, data will be stored in RAM only and be lost when the app exits.

The `SYSTEM` preference filename is hardcoded as `/etc/ftk/${vendor}/${application}.prefs`. For backward compatibility, the old `USER.prefs` file naming scheme `${directory}/.ftk/${vendor}/${application}` is checked first. If that file does not exist, the environment variable `$XDG_CONFIG_HOME` is read as a base directory. If `$XDG_CONFIG_HOME` is not set, the base directory defaults to `$HOME/.config/`.

The user preferences will be stored in `${directory}/${vendor}/${application}.prefs`. The user data path will be `${directory}/${vendor}/${application}/`.

In FLTK versions before 1.4.0, if `$HOME` was not set, the `USER` path would be empty, generating `${vendor}/${application}.prefs`, which was used relative to the current working directory.

Note

On **macOS**, the `USER` directory is constructed by reading `$HOME`. If `$HOME` is not set or not pointing to an existing directory, we check the path returned by `NSHomeDirectory()`, and finally checking the path member of the `passwd` struct returned by `getpwuid(getuid())`. If all attempts fail, data will be stored in RAM only and be lost when the app exits. The filename and path is then constructed as `${directory}/Library/Preferences/${vendor}/${application}.prefs`. The `SYSTEM` directory is hardcoded as `/Library/Preferences/${vendor}/${application}.prefs`.

In FLTK versions before 1.4.0, if `$HOME` was not set, the `USER` path

would be `NULL`, generating `<null>/Library/Preferences/${vendor}/${application}.prefs`, which would silently fail to create a preference file.

Parameters

in	<i>root</i>	can be <code>USER_L</code> or <code>SYSTEM_L</code> for user specific or system wide preferences, add the <code>CLEAR</code> flag to start with a clean set of preferences instead of reading them from a preexisting database
in	<i>vendor</i>	unique text describing the company or author of this file, must be a valid filepath segment
in	<i>application</i>	unique text describing the application, must be a valid filepath segment

See also

[Fl_Preferences\(Fl_Preferences *parent, const char *group\)](#) with parent [set](#) to `NULL`

11.106.4.2 Fl_Preferences() [2/8]

```
Fl_Preferences::Fl_Preferences (
    const char * path,
    const char * vendor,
    const char * application,
    Root flags)
```

Use this constructor to create or read a preference file at an arbitrary position in the file system.

The file name is generated in the form `${path}/${application}.prefs`. If `application` is `NULL`, `path` is taken literally as the file path and name.

```
// Example: read from an existing database and write modifications when flushed
// or destructor is called
Fl_Preferences database("/user/matt/test.prefs", "org.fltk.test", NULL,
    Fl_Preferences::C_LOCALE);
```

```
// Example: create a new preferences file with an empty data set
Fl_Preferences database("/user/matt/test.prefs", "org.fltk.test", NULL,
    (Fl_Preferences::Root) (Fl_Preferences::C_LOCALE|Fl_Preferences::CLEAR));
```

Note

the `C_LOCALE` flag is not set by default for backward compatibility, but it is highly recommended to set it when opening a database.

Parameters

in	<i>path</i>	path to the directory that contains the preference file
in	<i>vendor</i>	unique text describing the company or author of this file, must be a valid file path segment
in	<i>application</i>	unique text describing the application, must be a valid filename or NULL
in	<i>flags</i>	C_LOCALE to make the preferences file independent of the current locale, add the CLEAR flag to start with a clean set of preferences instead of reading from the database

11.106.4.3 Fl_Preferences() [3/8]

```
Fl_Preferences::Fl_Preferences (
    Fl_Preferences & parent,
    const char * group)
```

Generate or read a new group of entries within another group.

Use the `group` argument to name the group that you would like to access. `Group` can also contain a path to a group further down the hierarchy by separating group names with a forward slash '/'.

Parameters

in	<i>parent</i>	reference object for the new group
in	<i>group</i>	name of the group to access (may contain '/'s)

11.106.4.4 Fl_Preferences() [4/8]

```
Fl_Preferences::Fl_Preferences (
    Fl_Preferences * parent,
    const char * group)
```

Create or access a group of preferences using a name.

Parent should point to a previously created parent preferences group to create a preferences hierarchy.

If `parent` is set to NULL, an unnamed database will be accessed that exists only in local memory and is not associated with a file on disk. The root type of this database is set to `Fl_Preferences::MEMORY`.

- the memory database is *not* shared among multiple instances of the same app
- memory databases are *not* thread safe
- all data will be lost when the app quits

```
void some_function() {
    Fl_Preferences guide( NULL, "Guide" );
    guide.set("answer", 42);
}
void other_function() {
    int x;
    Fl_Preferences guide( NULL, "Guide" );
    guide.get("answer", x, -1);
}
```

FLTK uses the memory database to manage plugins. See [Fl_Plugin](#).

Parameters

in	<i>parent</i>	the parameter parent is a pointer to the parent group. If <code>parent</code> is NULL, the new preferences item refers to an application internal database ("runtime prefs") which exists only once, and remains in RAM only until the application quits. This database is used to manage plugins and other data indexes by strings. Runtime prefs are <i>not</i> thread-safe.
in	<i>group</i>	a group name that is used as a key into the database

See also

[Fl_Preferences\(Fl_Preferences&, const char *group \)](#)

11.106.4.5 Fl_Preferences() [5/8]

```
Fl_Preferences::Fl_Preferences (
    Fl_Preferences & parent,
    int groupIndex)
```

Open a child group using a given index.

Use the `groupIndex` argument to find the group that you would like to access. If the given index is invalid (negative or too high), a new group is created with a UUID as a name.

The index needs to be fixed. It is currently backward. Index 0 points to the last member in the 'list' of preferences.

Parameters

in	<i>parent</i>	reference object for the new group
in	<i>groupIndex</i>	zero based index into child groups

11.106.4.6 Fl_Preferences() [6/8]

```
Fl_Preferences::Fl_Preferences (
    Fl_Preferences * parent,
    int groupIndex)
```

See also

[Fl_Preferences\(Fl_Preferences&, int groupIndex \)](#)

11.106.4.7 Fl_Preferences() [7/8]

```
Fl_Preferences::Fl_Preferences (
    Fl_Preferences::ID id)
```

Create a new dataset access point using a dataset ID.

ID's are a great way to remember shortcuts to database entries that are deeply nested in a preferences database, as long as the database root is not deleted. An ID can be retrieved from any [Fl_Preferences](#) dataset, and can then be used to create multiple new references to the same dataset.

ID's can be very helpful when put into the `user_data()` field of widget callbacks.

11.106.4.8 ~Fl_Preferences()

```
Fl_Preferences::~~Fl_Preferences () [virtual]
```

The destructor removes allocated resources.

When used on the *base* preferences group, the destructor flushes all changes to the preference file and deletes all internal databases.

The destructor does not remove any data from the database. It merely deletes your reference to the database.

11.106.4.9 Fl_Preferences() [8/8]

```
Fl_Preferences::Fl_Preferences (
    const char * path,
    const char * vendor,
    const char * application)
```

Deprecated: Use this constructor to create or read a preference file at an arbitrary position in the file system.

Deprecated "since 1.4.0 - use `Fl_Preferences(path, vendor, application, flags)` instead"

This constructor should no longer be used because the generated database uses the current locale, making it impossible to exchange floating point settings between machines with different language settings.

Use `Fl_Preferences(path, vendor, application, Fl_Preferences::C_LOCALE)` in new projects and `Fl_Preferences(path, vendor, application, 0)` if you must keep backward compatibility.

See also

[Fl_Preferences\(const char *path, const char *vendor, const char *application, Root flags \)](#)

11.106.5 Member Function Documentation

11.106.5.1 delete_entry()

```
char Fl_Preferences::delete_entry (
    const char * key)
```

Deletes a single name/value pair.

This function removes the entry *key* from the database.

Parameters

in	<i>key</i>	name of entry to delete

Returns

0 if deleting the entry failed

11.106.5.2 delete_group()

```
char Fl_Preferences::delete_group (
    const char * group)
```

Deletes a group.

Removes a group and all keys and groups within that group from the database.

Parameters

in	<i>group</i>	name of the group to delete

Returns

0 if call failed

11.106.5.3 dirty()

```
int Fl_Preferences::dirty ()
```

Check if there were changes to the database that need to be written to disk.

Returns

1 if the database will be written to disk by *flush* or destructor.

0 if the database is unchanged since the last write operation.

-1 if there is an internal database error.

11.106.5.4 entries()

```
int Fl_Preferences::entries ()
```

Returns the number of entries (name/value pairs) in a group.

Returns

number of entries

11.106.5.5 entry()

```
const char * Fl_Preferences::entry (  
    int index)
```

Returns the name of an entry.

There is no guaranteed order of entry names. The index must be within the range given by [entries\(\)](#).

Parameters

in	<i>index</i>	number indexing the requested entry

Returns

pointer to value cstring

11.106.5.6 entry_exists()

```
char Fl_Preferences::entry_exists (
    const char * key)
```

Returns non-zero if an entry with this name exists.

Parameters

in	<i>key</i>	name of entry that is searched for

Returns

0 if entry was not found

11.106.5.7 file_access() [1/2]

```
unsigned int Fl_Preferences::file_access () [static]
```

Return the current file access permissions for the FLTK preferences system.

See also

[Fl_Preferences::file_access\(unsigned int\)](#)

11.106.5.8 file_access() [2/2]

```
void Fl_Preferences::file_access (
    unsigned int flags) [static]
```

Tell the FLTK preferences system which files in the file system it may read, create, or write.

The FLTK core library will try to read or even create or write preference files when calling [Fl::option\(\)](#), [Fl_File_Chooser](#), the printing panel, and possibly some other internal functions. If your application wants to keep FLTK from touching the file system, call this function before making any other FLTK calls:

```
// neither FLTK nor the app may read, create, or write preference files
Fl_Preferences::file_access( Fl_Preferences::NONE );
```

or

```
// FLTK may not read, create, or write preference files, but the application may
Fl_Preferences::file_access( Fl_Preferences::APP_OK );
```

All flags can be combined using an OR operator. If flags are not set, that specific access to the file system will not be allowed. By default, all access is granted. To clear one or more flags from the default setting, use:

```
Fl_Preferences::file_access( Fl_Preferences::file_access()
    &~ Fl_Preferences::SYSTEM_WRITE );
```

If preferences are created using a filename (instead of [Fl_Preferences::USER](#) or [Fl_Preferences::SYSTEM](#)), file access is handled as if the [Fl_Preferences::USER](#) flag was set.

See also

[Fl_Preferences::NONE](#) and others for a list of flags.

[Fl_Preferences::file_access\(\)](#)

11.106.5.9 filename() [1/2]

```
Fl_Preferences::Root Fl_Preferences::filename (  
    char * buffer,  
    size_t buffer_size)
```

Return the file name and path to the preference file.

If the preferences have not changed or have not been flushed, the file or directory may not have been created yet.

Parameters

out	<i>buffer</i>	write the resulting path into this buffer
in	<i>buffer_size</i>	size of the <i>buffer</i> in bytes

Returns

the root type at creation type, or MEMORY for runtime prefs, it does not return CORE or LOCALE flags.

11.106.5.10 filename() [2/2]

```
Fl_Preferences::Root Fl_Preferences::filename (
    char * buffer,
    size_t buffer_size,
    Root root,
    const char * vendor,
    const char * application) [static]
```

Determine the file name and path to preferences that would be opened with these parameters.

Find the possible location of a preference file on disk without touching any of the pathname components. This can be used to check if a preference file already exists.

Parameters

out	<i>buffer</i>	write the resulting path into this buffer
in	<i>buffer_size</i>	size of the <i>buffer</i> in bytes
in	<i>root</i>	can be USER_L or SYSTEM_L for user specific or system wide preferences
in	<i>vendor</i>	unique text describing the company or author of this file, must be a valid filepath segment
in	<i>application</i>	unique text describing the application, must be a valid filepath segment

Returns

the input root value, or [Fl_Preferences::UNKNOWN_ROOT_TYPE](#) if the path could not be determined.

See also

[Fl_Preferences\(Root root, const char *vendor, const char *application \)](#)

11.106.5.11 flush()

```
int Fl_Preferences::flush ()
```

Writes preferences to disk if they were modified.

This method can be used to verify that writing a preference file went well. Deleting the base preferences object will also write the contents of the database to disk.

Returns

-1 if anything went wrong, i.e. file could not be opened, permissions blocked writing, etc.

0 if the file was written to disk. This does not check if the disk ran out of space and the file is truncated.

1 if no data was written to the database and no write attempt to disk was made.

11.106.5.12 get() [1/9]

```
char Fl_Preferences::get (  
    const char * key,  
    char *& text,  
    const char * defaultValue)
```

Reads an entry from the group.

A default value must be supplied. The return value indicates if the value was available (non-zero) or the default was used (0). [get\(\)](#) allocates memory of sufficient size to hold the value. The buffer must be free'd by the developer using 'free(value)'.

Parameters

in	<i>key</i>	name of entry
out	<i>text</i>	returned from preferences or default value if none was set
in	<i>defaultValue</i>	default value to be used if no preference was set

Returns

0 if the default value was used

11.106.5.13 get() [2/9]

```
char Fl_Preferences::get (  
    const char * key,  
    char * text,  
    const char * defaultValue,  
    int maxSize)
```

Reads an entry from the group.

A default value must be supplied. The return value indicates if the value was available (non-zero) or the default was used (0). 'maxSize' is the maximum length of text that will be read. The text buffer must allow for one additional byte for a trailing zero.

Parameters

in	<i>key</i>	name of entry
out	<i>text</i>	returned from preferences or default value if none was set
in	<i>defaultValue</i>	default value to be used if no preference was set
in	<i>maxSize</i>	maximum length of value plus one byte for a trailing zero

Returns

0 if the default value was used

11.106.5.14 get() [3/9]

```
char Fl_Preferences::get (  
    const char * key,  
    double & value,  
    double defaultValue)
```

Reads an entry from the group.

A default value must be supplied. The return value indicates if the value was available (non-zero) or the default was used (0).

Parameters

in	<i>key</i>	name of entry
out	<i>value</i>	returned from preferences or default value if none was set
in	<i>defaultValue</i>	default value to be used if no preference was set

Returns

0 if the default value was used

11.106.5.15 get() [4/9]

```
char Fl_Preferences::get (  
    const char * key,  
    float & value,  
    float defaultValue)
```

Reads an entry from the group.

A default value must be supplied. The return value indicates if the value was available (non-zero) or the default was used (0).

Parameters

in	<i>key</i>	name of entry
out	<i>value</i>	returned from preferences or default value if none was set
in	<i>defaultValue</i>	default value to be used if no preference was set

Returns

0 if the default value was used

11.106.5.16 get() [5/9]

```
char Fl_Preferences::get (  
    const char * key,  
    int & value,  
    int defaultValue)
```

Reads an entry from the group.

A default value must be supplied. The return value indicates if the value was available (non-zero) or the default was used (0).

Parameters

in	<i>key</i>	name of entry
out	<i>value</i>	returned from preferences or default value if none was set
in	<i>defaultValue</i>	default value to be used if no preference was set

Returns

0 if the default value was used

11.106.5.17 get() [6/9]

```
char Fl_Preferences::get (
    const char * key,
    std::string & value,
    const std::string & defaultValue)
```

Reads an entry from the group.

A default value must be supplied. The return value indicates if the value was available (non-zero) or the default was used (0). NUL characters in std::string are treated as the end of the text.

Parameters

in	<i>key</i>	name of entry
out	<i>value</i>	returned from preferences or default value if none was set
in	<i>defaultValue</i>	default value to be used if no preference was set

Returns

0 if the default value was used

11.106.5.18 get() [7/9]

```
char Fl_Preferences::get (
    const char * key,
    void *& data,
    const void * defaultValue,
    int defaultSize)
```

Reads an entry from the group.

A default value must be supplied. The return value indicates if the value was available (non-zero) or the default was used (0). [get\(\)](#) allocates memory of sufficient size to hold the value. The buffer must be free'd by the developer using 'free(value)'.

Parameters

in	<i>key</i>	name of entry
out	<i>data</i>	returned from preferences or default value if none was set
in	<i>defaultValue</i>	default value to be used if no preference was set
in	<i>defaultSize</i>	size of default value array

Returns

0 if the default value was used

11.106.5.19 get() [8/9]

```
char Fl_Preferences::get (
    const char * key,
    void * data,
    const void * defaultValue,
    int defaultSize,
    int * maxSize)
```

Reads a binary entry from the group, encoded in hexadecimal blocks.

A binary (not hex) default value can be supplied. The return value indicates if the value was available (non-zero) or the default was used (0). *maxSize* is the maximum length of text that will be read and returns the actual number of bytes read.

Parameters

in	<i>key</i>	name of entry
out	<i>data</i>	value returned from preferences or default value if none was set
in	<i>defaultValue</i>	default value to be used if no preference was set
in	<i>defaultSize</i>	size of default value array
in, out	<i>maxSize</i>	maximum length of value and actual number of bytes set

Returns

0 if the default value was used

11.106.5.20 `get()` [9/9]

```
char Fl_Preferences::get (
    const char * key,
    void * data,
    const void * defaultValue,
    int defaultSize,
    int maxSize)
```

Reads a binary entry from the group, encoded in hexadecimal blocks.

Parameters

in	<i>key</i>	name of entry
out	<i>data</i>	value returned from preferences or default value if none was set
in	<i>defaultValue</i>	default value
in	<i>defaultSize</i>	size of default value array
in	<i>maxSize</i>	maximum length of value, to receive the number of bytes read, use the function below instead.

Returns

0 if the default value was used

See also

[Fl_Preferences::get\(const char *key, void *data, const void *defaultValue, int defaultSize, int *maxSize \)](#)

11.106.5.21 `get_userdata_path()`

```
char Fl_Preferences::get_userdata_path (
    char * path,
    int pathlen)
```

Creates a path that is related to the preference file and that is usable for additional application data.

This function creates a directory that is named after the preferences database without the .prefs extension and located in the same directory. It then fills the given buffer with the complete path name.

There is no way to verify that the path name fit into the buffer. If the name is too long, it will be clipped.

This function can be used with direct paths that don't end in .prefs . `getUserDataPath()` will remove any extension and end the path with a / . If the file name has no extension, `getUserDataPath()` will append .data/ to the path name.

Example:

```
Fl_Preferences prefs( USER, "matthiasm.com", "test" );
char path[FL_PATH_MAX];
prefs.getUserdataPath( path, FL_PATH_MAX );
```

creates the preferences database in the directory (User 'matt' on Linux):

```
/Users/matt/.fltk/matthiasm.com/test.prefs
```

..and returns the userdata path:

```
/Users/matt/.fltk/matthiasm.com/test/
```

Parameters

out	<i>path</i>	buffer for user data path
in	<i>pathlen</i>	size of path buffer (should be at least <code>FL_PATH_MAX</code>)

Returns

- 1 if there is no filename (*path* will be unmodified)
- 1 if *pathlen* is 0 (*path* will be unmodified)
- 1 if a path was created successfully, *path* will contain the path name ending in a '/'
- 0 if path was not created for some reason; *path* will contain the pathname that could not be created

See also

[Fl_Preferences::Fl_Preferences\(Root, const char*, const char*\)](#)

11.106.5.22 group()

```
const char * Fl_Preferences::group (
    int num_group)
```

Returns the name of the Nth (*num_group*) group.

There is no guaranteed order of group names. The index must be within the range given by [groups\(\)](#).

Parameters

in	<i>num_group</i>	number indexing the requested group

Returns

'C' string pointer to the group name

11.106.5.23 group_exists()

```
char Fl_Preferences::group_exists (
    const char * key)
```

Returns non-zero if a group with this name exists.

Group names are relative to the [Fl_Preferences](#) node and can contain a path. "." describes the current node, "/" describes the topmost node. By preceding a groupname with a "/" its path becomes relative to the topmost node.

Parameters

in	<i>key</i>	name of group that is searched for

Returns

0 if no group by that name was found

11.106.5.24 groups()

```
int Fl_Preferences::groups ()
```

Returns the number of groups that are contained within a group.

Returns

0 for no groups at all

11.106.5.25 new_UUID()

```
const char * Fl_Preferences::new_UUID () [static]
```

Returns a UUID as generated by the system.

A UUID is a "universally unique identifier" which is commonly used in configuration files to create identities. A UUID in ASCII looks like this: 937C4900-51AA-4C11-8DD3-7AB59944F03E. It has always 36 bytes plus a trailing zero.

Returns

a pointer to a static buffer containing the new UUID in ASCII format. The buffer is overwritten during every call to this function!

11.106.5.26 set() [1/8]

```
char Fl_Preferences::set (
    const char * key,
    const char * text)
```

Sets an entry (name/value pair).

The return value indicates if there was a problem storing the data in memory. However it does not reflect if the value was actually stored in the preference file.

Parameters

in	<i>key</i>	name of entry
in	<i>text</i>	set this entry to value

Returns

0 if setting the value failed

11.106.5.27 set() [2/8]

```
char Fl_Preferences::set (
    const char * entry,
    const std::string & value)
```

Sets an entry (name/value pair).

The return value indicates if there was a problem storing the data in memory. However it does not reflect if the value was actually stored in the preference file. NUL characters in std::string are treated as the end of the text.

Parameters

in	<i>entry</i>	name of entry
in	<i>value</i>	set this entry to value (stops at the first nul character).

Returns

0 if setting the value failed

11.106.5.28 set() [3/8]

```
char Fl_Preferences::set (
    const char * key,
    const void * data,
    int dsize)
```

Sets an entry (name/value pair).

The return value indicates if there was a problem storing the data in memory. However it does not reflect if the value was actually stored in the preference file.

Parameters

in	<i>key</i>	name of entry
in	<i>data</i>	set this entry to <i>value</i>
in	<i>dsize</i>	size of data array

Returns

0 if setting the value failed

11.106.5.29 set() [4/8]

```
char Fl_Preferences::set (  
    const char * key,  
    double value)
```

Sets an entry (name/value pair).

The return value indicates if there was a problem storing the data in memory. However it does not reflect if the value was actually stored in the preference file.

Parameters

in	<i>key</i>	name of entry
in	<i>value</i>	set this entry to <i>value</i>

Returns

0 if setting the value failed

11.106.5.30 set() [5/8]

```
char Fl_Preferences::set (  
    const char * key,  
    double value,  
    int precision)
```

Sets an entry (name/value pair).

The return value indicates if there was a problem storing the data in memory. However it does not reflect if the value was actually stored in the preference file.

Parameters

in	<i>key</i>	name of entry
in	<i>value</i>	set this entry to <i>value</i>
in	<i>precision</i>	number of decimal digits to represent value

Returns

0 if setting the value failed

11.106.5.31 set() [6/8]

```
char Fl_Preferences::set (  
    const char * key,  
    float value)
```

Sets an entry (name/value pair).

The return value indicates if there was a problem storing the data in memory. However it does not reflect if the value was actually stored in the preference file.

Parameters

in	<i>key</i>	name of entry
in	<i>value</i>	set this entry to <i>value</i>

Returns

0 if setting the value failed

11.106.5.32 set() [7/8]

```
char Fl_Preferences::set (  
    const char * key,  
    float value,  
    int precision)
```

Sets an entry (name/value pair).

The return value indicates if there was a problem storing the data in memory. However it does not reflect if the value was actually stored in the preference file.

Parameters

in	<i>key</i>	name of entry
in	<i>value</i>	set this entry to <i>value</i>
in	<i>precision</i>	number of decimal digits to represent value

Returns

0 if setting the value failed

11.106.5.33 set() [8/8]

```
char Fl_Preferences::set (  
    const char * key,  
    int value)
```

Sets an entry (name/value pair).

The return value indicates if there was a problem storing the data in memory. However it does not reflect if the value was actually stored in the preference file.

Parameters

in	<i>key</i>	name of entry
in	<i>value</i>	set this entry to <i>value</i>

Returns

0 if setting the value failed

11.106.5.34 size()

```
int Fl_Preferences::size (  
    const char * key)
```

Returns the size of the value part of an entry.

Parameters

in	key	name of entry

Returns

size of value

11.106.6 Member Data Documentation

11.106.6.1 CORE_READ_OK

```
const unsigned int Fl_Preferences::CORE_READ_OK = 0x0010 [static]
```

Set this if it is OK for FLTK to read preference files.

USER_READ_OK and/or SYSTEM_READ_OK must also be set.

11.106.6.2 CORE_WRITE_OK

```
const unsigned int Fl_Preferences::CORE_WRITE_OK = 0x0020 [static]
```

Set this if it is OK for FLTK to create or write preference files.

USER_WRITE_OK and/or SYSTEM_WRITE_OK must also be set.

11.106.6.3 NONE

```
const unsigned int Fl_Preferences::NONE = 0x0000 [static]
```

Set this if no call to [Fl_Preferences](#) shall access the file system.

See also

[Fl_Preferences::file_access\(unsigned int\)](#)

[Fl_Preferences::file_access\(\)](#)

The documentation for this class was generated from the following files:

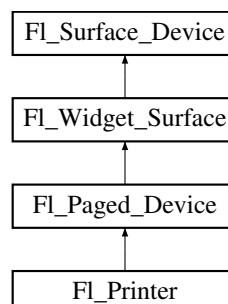
- [Fl_Preferences.H](#)
- [Fl_Preferences.cxx](#)

11.107 Fl_Printer Class Reference

OS-independent print support.

```
#include <Fl_Printer.H>
```

Inheritance diagram for [Fl_Printer](#):



Public Member Functions

- [int begin_job](#) (int pagecount=0, int *frompage=NULL, int *topage=NULL, char **perr_message=NULL)
[FL_OVERRIDE](#)
Begins a print job.

- int [begin_page](#) (void) [FL_OVERRIDE](#)
Begins a new printed page.
- void [end_job](#) (void) [FL_OVERRIDE](#)
To be called at the end of a print job.
- int [end_page](#) (void) [FL_OVERRIDE](#)
To be called at the end of each page.
- **FI_Printer** (void)
The constructor.
- bool [is_current](#) () [FL_OVERRIDE](#)
Is this surface the current drawing surface?
- void [margins](#) (int *left, int *top, int *right, int *bottom) [FL_OVERRIDE](#)
Computes the dimensions of margins that lie between the printable page area and the full page.
- void [origin](#) (int *x, int *y) [FL_OVERRIDE](#)
Computes the coordinates of the current origin of graphics functions.
- void [origin](#) (int x, int y) [FL_OVERRIDE](#)
Sets the position of the origin of graphics in the drawable part of the drawing surface.
- int [printable_rect](#) (int *w, int *h) [FL_OVERRIDE](#)
Computes the width and height of the drawable area of the drawing surface.
- void [rotate](#) (float angle) [FL_OVERRIDE](#)
Rotates the graphics operations relatively to paper.
- void [scale](#) (float scale_x, float scale_y=0.) [FL_OVERRIDE](#)
Changes the scaling of page coordinates.
- void [set_current](#) (void) [FL_OVERRIDE](#)
Make this surface the current drawing surface.
- void [translate](#) (int x, int y) [FL_OVERRIDE](#)
Translates the current graphics origin accounting for the current rotation.
- void [untranslate](#) (void) [FL_OVERRIDE](#)
Undoes the effect of a previous [translate\(\)](#) call.
- **~FI_Printer** (void)
The destructor.

Public Member Functions inherited from [FI_Paged_Device](#)

- void **print_widget** ([FI_Widget](#) *widget, int delta_x=0, int delta_y=0)
Synonym of [draw\(FI_Widget, int, int\)](#)*
- void **print_window** ([FI_Window](#) *win, int x_off=0, int y_off=0)
Synonym of [draw_decorated_window\(FI_Window, int, int\)](#)*
- int **start_job** (int pagecount=0, int *frompage=NULL, int *topage=NULL, char **perr_message=NULL)
*Synonym of [begin_job\(int pagecount, int *frompage, int *topage, char **perr_message\)](#).*
- int **start_page** ()
Synonym of [begin_page\(\)](#).
- virtual **~FI_Paged_Device** ()
The destructor.

Public Member Functions inherited from [FI_Widget_Surface](#)

- void **draw** ([FI_Widget](#) *widget, int delta_x=0, int delta_y=0)
Draws the widget on the drawing surface.
- void **draw_decorated_window** ([FI_Window](#) *win, int x_offset=0, int y_offset=0)
Draws a window with its title bar and frame if any.
- void **print_window_part** ([FI_Window](#) *win, int x, int y, int w, int h, int delta_x=0, int delta_y=0)
Draws a rectangular part of an on-screen window.

Public Member Functions inherited from [FI_Surface_Device](#)

- `FI_Graphics_Driver * driver ()`
Returns the graphics driver of this drawing surface.
- `virtual ~FI_Surface_Device ()`
The destructor.

Static Public Attributes

These attributes are useful for the Linux/Unix platform only.

- `static const char * dialog_title = "Print"`
[this text may be customized at run-time]
- `static const char * dialog_printer = "Printer:"`
[this text may be customized at run-time]
- `static const char * dialog_range = "Print Range"`
[this text may be customized at run-time]
- `static const char * dialog_copies = "Copies"`
[this text may be customized at run-time]
- `static const char * dialog_all = "All"`
[this text may be customized at run-time]
- `static const char * dialog_pages = "Pages"`
[this text may be customized at run-time]
- `static const char * dialog_from = "From:"`
[this text may be customized at run-time]
- `static const char * dialog_to = "To:"`
[this text may be customized at run-time]
- `static const char * dialog_properties = "Properties..."`
[this text may be customized at run-time]
- `static const char * dialog_copyNo = "# Copies:"`
[this text may be customized at run-time]
- `static const char * dialog_print_button = "Print"`
[this text may be customized at run-time]
- `static const char * dialog_cancel_button = "Cancel"`
[this text may be customized at run-time]
- `static const char * dialog_print_to_file = "Print To File"`
[this text may be customized at run-time]
- `static const char * property_title = "Printer Properties"`
[this text may be customized at run-time]
- `static const char * property_pagesize = "Page Size:"`
[this text may be customized at run-time]
- `static const char * property_mode = "Output Mode:"`
[this text may be customized at run-time]
- `static const char * property_use = "Use"`
[this text may be customized at run-time]
- `static const char * property_save = "Save"`
[this text may be customized at run-time]
- `static const char * property_cancel = "Cancel"`
[this text may be customized at run-time]

Static Public Attributes inherited from [FI_Paged_Device](#)

- `static const page\_format page_formats [NO_PAGE_FORMATS]`
width, height and name of all elements of the enum [Page_Format](#).

Additional Inherited Members

Public Types inherited from FI_Paged_Device

- enum [Page_Format](#) {
A0 = 0 , A1 , A2 , A3 ,
A4 , A5 , A6 , A7 ,
A8 , A9 , B0 , B1 ,
B2 , B3 , B4 , B5 ,
B6 , B7 , B8 , B9 ,
B10 , C5E , DLE , EXECUTIVE ,
FOLIO , LEDGER , LEGAL , LETTER ,
TABLOID , ENVELOPE , MEDIA = 0x1000 }
Possible page formats.
- enum [Page_Layout](#) { PORTRAIT = 0 , LANDSCAPE = 0x100 , REVERSED = 0x200 , ORIENTATION = 0x300 }
Possible page layouts.

Static Public Member Functions inherited from FI_Surface_Device

- static [FI_Surface_Device](#) * [pop_current](#) ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void [push_current](#) ([FI_Surface_Device](#) *new_current)
Pushes new_current on top of the stack of current drawing surfaces, and makes it current.
- static [FI_Surface_Device](#) * [surface](#) ()
The current drawing surface.

Protected Member Functions inherited from FI_Paged_Device

- [FI_Paged_Device](#) ()
The constructor.

Protected Member Functions inherited from FI_Widget_Surface

- [FI_Widget_Surface](#) ([FI_Graphics_Driver](#) *d)
The constructor.

Protected Member Functions inherited from FI_Surface_Device

- void [driver](#) ([FI_Graphics_Driver](#) *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void [end_current](#) ()
FLTK calls this each time a surface ceases to be the current drawing surface.
- [FI_Surface_Device](#) ([FI_Graphics_Driver](#) *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

Protected Attributes inherited from FI_Widget_Surface

- int [x_offset](#)
horizontal offset to the origin of graphics coordinates
- int [y_offset](#)
vertical offset to the origin of graphics coordinates

11.107.1 Detailed Description

OS-independent print support.

[FL_Printer](#) allows to use all drawing, color, text, image, and clip FLTK functions, and to have them operate on printed page(s). There are two main, non exclusive, ways to use it.

- Print any widget (standard, custom, [FL_Window](#), [FL_Gl_Window](#)) as it appears on screen, with optional translation, scaling and rotation. This is done by calling [print_widget\(\)](#), [print_window\(\)](#) or [print_window_part\(\)](#).
- Use a series of FLTK graphics commands (e.g., font, text, lines, colors, clip, image) to compose a page appropriately shaped for printing.

In both cases, begin by [begin_job\(\)](#), [begin_page\(\)](#), [printable_rect\(\)](#) and [origin\(\)](#) calls and finish by [end_page\(\)](#) and [end_job\(\)](#) calls.

Example of use: print a widget centered in a page

```
#include <FL/Fl_Printer.H>
#include <FL/fl_draw.H>
int width, height;
Fl_Widget *widget = ... // a widget we want printed
Fl_Printer *printer = new Fl_Printer();
if (printer->begin_job(1) == 0) {
    printer->begin_page();
    printer->printable_rect(&width, &height);
    fl_color(FL_BLACK);
    fl_line_style(FL_SOLID, 2);
    fl_rect(0, 0, width, height);
    fl_font(FL_COURIER, 12);
    time_t now; time(&now); fl_draw(ctime(&now), 0, fl_height());
    printer->origin(width/2, height/2);
    printer->print_widget(widget, -widget->w()/2, -widget->h()/2);
    printer->end_page();
    printer->end_job();
}
delete printer;
```

Recommended method to refresh GUI while printing :

```
printer->begin_job(0);
.....
Fl_Surface_Device::push_current(Fl_Display_Device::display_device());
Fl::check(); // or any operation that draws to display
Fl_Surface_Device::pop_current();
.....
printer->end_job();
```

Platform specifics

- X11 and Wayland platforms:
 - FLTK expresses all graphics data using (Level 2) PostScript and sends that to the selected printer. See class [FL_PostScript_File_Device](#) for a description of how text and transparent images appear in print.
 - If the GTK library is available at run-time, class [FL_Printer](#) runs GTK's printer dialog which allows to set printer, paper size and orientation.
 - If the GTK library is not available, or if `Fl::option(Fl::OPTION_PRINTER_USES_GTK)` has been turned off, class [FL_Printer](#) runs FLTK's print dialog.
 - * Unless it has been previously changed, the default paper size is A4. To change that, press the "Properties" button of the "Print" dialog window opened by an [FL_Printer::begin_job\(\)](#) call. This opens a "Printer Properties" window where it's possible to select the adequate paper size. Finally press the "Save" button therein to assign the chosen paper size to the chosen printer for this and all further print operations.
 - * Use the static public attributes of this class to set the print dialog to other languages than English. For example, the "Printer:" dialog item [FL_Printer::dialog_printer](#) can be set to French with:


```
Fl_Printer::dialog_printer = "Imprimante:";
```

 before creation of the [FL_Printer](#) object.
 - * Use [FL_PostScript_File_Device::file_chooser_title](#) to customize the title of the file chooser dialog that opens when using the "Print To File" option of the print dialog.
- Windows platform: Transparent [FL_RGB_Image](#) 's don't print with exact transparency on most printers (a workaround is to use [print_window_part\(\)](#)). [FL_RGB_Image](#) 's don't [rotate\(\)](#) well.
- Mac OS X platform: all graphics requests print as on display and accept rotation and scaling.

11.107.2 Member Function Documentation

11.107.2.1 begin_job()

```
int Fl_Printer::begin_job (
    int pagecount = 0,
    int * frompage = NULL,
    int * topage = NULL,
    char ** perr_message = NULL) [virtual]
```

Begins a print job.

Parameters

in	<i>pagecount</i>	the total number of pages of the job (or 0 if you don't know the number of pages)
out	<i>frompage</i>	if non-null, *frompage is set to the first page the user wants printed
out	<i>topage</i>	if non-null, *topage is set to the last page the user wants printed
out	<i>perr_message</i>	if non-null and if the returned value is 2, *perr_message is set to a string describing the error. That string can be delete[]'d after use.

Returns

0 if OK, 1 if user cancelled the job, 2 if any error.

Reimplemented from [Fl_Paged_Device](#).

11.107.2.2 begin_page()

```
int Fl_Printer::begin_page (
    void ) [virtual]
```

Begins a new printed page.

The page coordinates are initially in points, i.e., 1/72 inch, and with origin at the top left of the printable page area. This function also makes this surface the current drawing surface with [Fl_Surface_Device::push_current\(\)](#).

Note

[begin_page\(\)](#) calls [Fl_Surface_Device::push_current\(\)](#) and leaves this device as the active surface. If any calls between [begin_page\(\)](#) and [end_page\(\)](#) open dialog boxes or will otherwise draw into FLTK windows, those calls must be put between a call to [Fl_Surface_Device::pop_current\(\)](#) and a call to [Fl_Surface_Device::push_current\(\)](#), or the content of the dialog box will be rendered to the printer instead of the screen.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Paged_Device](#).

11.107.2.3 end_job()

```
void Fl_Printer::end_job (
    void ) [virtual]
```

To be called at the end of a print job.

Reimplemented from [Fl_Paged_Device](#).

11.107.2.4 end_page()

```
int Fl_Printer::end_page (
    void ) [virtual]
```

To be called at the end of each page.

This function also stops this surface from being the current drawing surface with [Fl_Surface_Device::pop_current\(\)](#).

Note

`end_page()` calls `Fl_Surface_Device::pop_current()`. If any calls between `begin_page()` and `end_page()` open dialog boxes or will otherwise draw into FLTK windows, those calls must be put between a call to `Fl_Surface_Device::pop_current()` and a call to `Fl_Surface_Device::push_current()`.

Returns

0 if OK, non-zero if any error.

Reimplemented from [Fl_Paged_Device](#).

11.107.2.5 is_current()

```
bool Fl_Printer::is_current () [virtual]
```

Is this surface the current drawing surface?

Reimplemented from [Fl_Surface_Device](#).

11.107.2.6 margins()

```
void Fl_Printer::margins (
    int * left,
    int * top,
    int * right,
    int * bottom) [virtual]
```

Computes the dimensions of margins that lie between the printable page area and the full page.

Values are in the same unit as that used by FLTK drawing functions. They are changed by [scale\(\)](#) calls.

Parameters

out	<i>left</i>	If non-null, *left is set to the left margin size.
out	<i>top</i>	If non-null, *top is set to the top margin size.
out	<i>right</i>	If non-null, *right is set to the right margin size.
out	<i>bottom</i>	If non-null, *bottom is set to the bottom margin size.

Reimplemented from [Fl_Paged_Device](#).

11.107.2.7 origin() [1/2]

```
void Fl_Printer::origin (
    int * x,
    int * y) [virtual]
```

Computes the coordinates of the current origin of graphics functions.

Parameters

out	<i>x,y</i>	If non-null, *x and *y are set to the horizontal and vertical coordinates of the graphics origin.

Reimplemented from [Fl_Widget_Surface](#).

11.107.2.8 origin() [2/2]

```
void Fl_Printer::origin (
    int x,
    int y) [virtual]
```

Sets the position of the origin of graphics in the drawable part of the drawing surface.

Arguments should be expressed relatively to the result of a previous [printable_rect\(\)](#) call. That is, `printable_rect(&w, &h); origin(w/2, 0);` sets the graphics origin at the top center of the drawable area. Successive [origin\(\)](#) calls don't combine their effects. [Origin\(\)](#) calls are not affected by [rotate\(\)](#) calls (for classes derived from [Fl_Paged_Device](#)).

Parameters

<code>in</code>	<code>x,y</code>	Horizontal and vertical positions in the drawing surface of the desired origin of graphics.

Reimplemented from [Fl_Widget_Surface](#).

11.107.2.9 printable_rect()

```
int Fl_Printer::printable_rect (
    int * w,
    int * h) [virtual]
```

Computes the width and height of the drawable area of the drawing surface.

Values are in the same unit as that used by FLTK drawing functions and are unchanged by calls to [origin\(\)](#). If the object is derived from class [Fl_Paged_Device](#), values account for the user-selected paper type and print orientation and are changed by [scale\(\)](#) calls.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Widget_Surface](#).

11.107.2.10 rotate()

```
void Fl_Printer::rotate (
    float angle) [virtual]
```

Rotates the graphics operations relatively to paper.

The rotation is centered on the current graphics origin. Successive [rotate\(\)](#) calls don't combine their effects.

Parameters

<code>angle</code>	Rotation angle in counter-clockwise degrees.

Reimplemented from [Fl_Paged_Device](#).

11.107.2.11 scale()

```
void Fl_Printer::scale (
    float scale_x,
    float scale_y = 0.) [virtual]
```

Changes the scaling of page coordinates.

This function also resets the origin of graphics functions at top left of printable page area. After a [scale\(\)](#) call, do a [printable_rect\(\)](#) call to get the new dimensions of the printable page area. Successive [scale\(\)](#) calls don't combine their effects.

Parameters

<code>scale_x</code>	Horizontal dimensions of plot are multiplied by this quantity.
<code>scale_y</code>	Same as above, vertically. The value 0. is equivalent to setting <code>scale_y = scale_x</code> . Thus, <code>scale(factor)</code> ; is equivalent to <code>scale(factor, factor)</code> ;

Reimplemented from [Fl_Paged_Device](#).

11.107.2.12 set_current()

```
void Fl_Printer::set_current (
    void ) [virtual]
```

Make this surface the current drawing surface.

This surface will receive all future graphics requests.

Since FLTK 1.4.0 the preferred API to change the current drawing surface is [Fl_Surface_Device::push_current\(\)](#) / [Fl_Surface_Device::pop_current\(\)](#).

Note

It is recommended to use this function only as follows :

- The current drawing surface is the display;
- make current another surface, e.g., an [Fl_Printer](#) or an [Fl_Image_Surface](#) object, calling [set_current\(\)](#) on this object;
- draw to that surface;
- make the display current again with [Fl_Display_Device::display_device\(\)->set_current\(\)](#); don't do any other call to [set_current\(\)](#) before this one.

Other scenarios of drawing surface changes should be performed via [Fl_Surface_Device::push_current\(\)](#) and [Fl_Surface_Device::pop_current\(\)](#).

Reimplemented from [Fl_Surface_Device](#).

11.107.2.13 translate()

```
void Fl_Printer::translate (
    int x,
    int y) [virtual]
```

Translates the current graphics origin accounting for the current rotation.

Each [translate\(\)](#) call must be matched by an [untranslate\(\)](#) call. Successive [translate\(\)](#) calls add up their effects.

Reimplemented from [Fl_Widget_Surface](#).

11.107.2.14 untranslate()

```
void Fl_Printer::untranslate (
    void ) [virtual]
```

Undoes the effect of a previous [translate\(\)](#) call.

Reimplemented from [Fl_Widget_Surface](#).

The documentation for this class was generated from the following files:

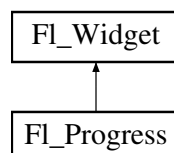
- [Fl_Printer.H](#)
- [Fl_Printer.cxx](#)

11.108 Fl_Progress Class Reference

Displays a progress bar for the user.

```
#include <Fl_Progress.H>
```

Inheritance diagram for [Fl_Progress](#):



Public Member Functions

- [Fl_Progress](#) (int [x](#), int [y](#), int [w](#), int [h](#), const char *[l](#)=0)
The constructor creates the progress bar using the position, size, and label.
- float [maximum](#) () const
Gets the maximum value in the progress widget.

- void **maximum** (float v)
Sets the maximum value in the progress widget.
- float **minimum** () const
Gets the minimum value in the progress widget.
- void **minimum** (float v)
Sets the minimum value in the progress widget.
- float **value** () const
Gets the current value in the progress widget.
- void **value** (float v)
Sets the current value in the progress widget.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- FI_Align **align** () const
Gets the label alignment.
- void **align** (FI_Align alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class FI_Gl_Window * **as_gl_window** ()
Returns an FI_Gl_Window pointer if this widget is an FI_Gl_Window.
- virtual class FI_Gl_Window const * **as_gl_window** () const
- virtual FI_Group * **as_group** ()
Returns an FI_Group pointer if this widget is an FI_Group.
- virtual FI_Group const * **as_group** () const
- virtual FI_Window * **as_window** ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual FI_Window const * **as_window** () const
- void **bind_deimage** (FI_Image *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (FI_Image *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- FI_Boxtype **box** () const
Gets the box type of the widget.
- void **box** (FI_Boxtype new_box)
Sets the box type for the widget.
- FI_Callback_p **callback** () const

- Gets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb)

Sets the current callback function for the widget.

 - void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.

 - void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.

 - void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.

 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.

 - unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.

 - void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.

 - void [clear_changed](#) ()

Marks the value of the widget as unchanged.

 - void [clear_damage](#) ([uchar](#) c=0)

Clears or sets the damage flags.

 - void [clear_output](#) ()

Sets a widget to accept input.

 - void [clear_visible](#) ()

Hides the widget.

 - void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.

 - [FI_Color](#) [color](#) () const

Gets the background color of the widget.

 - void [color](#) ([FI_Color](#) bg)

Sets the background color of the widget.

 - void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)

Sets the background and selection color of the widget.

 - [FI_Color](#) [color2](#) () const

For back compatibility only.

 - void [color2](#) (unsigned a)

For back compatibility only.

 - int [contains](#) (const [FI_Widget](#) *w) const

Checks if w is a child of this widget.

 - void [copy_label](#) (const char *new_label)

Sets the current label.

 - void [copy_tooltip](#) (const char *text)

Sets the current tooltip text.

 - [uchar](#) [damage](#) () const

Returns non-zero if [draw\(\)](#) needs to be called.

 - void [damage](#) ([uchar](#) c)

Sets the damage bits for the widget.

 - void [damage](#) ([uchar](#) c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.

 - int [damage_resize](#) (int, int, int, int)

Internal use only.

 - void [deactivate](#) ()

Deactivates the widget.

- `FL_Image * deimage ()`
Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`
Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`
Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`
Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`
Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FL_Align) const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`
Gets the widget height.
- `virtual int handle (int event)`
Handles the specified event.
- `virtual void hide ()`
Makes a widget invisible.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FL_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FL_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FL_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with copy_label().
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FL_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`

- Set the gap between the label and the image in pixels.*

 - [FI_Color labelcolor](#) () const
 - Gets the label color.*
 - void [labelcolor](#) ([FI_Color](#) c)
 - Sets the label color.*
 - [FI_Font labelfont](#) () const
 - Gets the font to use.*
 - void [labelfont](#) ([FI_Font](#) f)
 - Sets the font to use.*
 - [FI_Fontsize labelsize](#) () const
 - Gets the font size in pixels.*
 - void [labelsiz](#) ([FI_Fontsize](#) pix)
 - Sets the font size in pixels.*
 - [FI_Labeltype labeltype](#) () const
 - Gets the label type.*
 - void [labeltype](#) ([FI_Labeltype](#) a)
 - Sets the label type.*
 - void [measure_label](#) (int &ww, int &hh) const
 - Sets width ww and height hh accordingly with the label size.*
 - bool [needs_keyboard](#) () const
 - Returns whether this widget needs a keyboard.*
 - void [needs_keyboard](#) (bool needs)
 - Sets whether this widget needs a keyboard.*
 - unsigned int [output](#) () const
 - Returns if a widget is used for output only.*
 - [FI_Group * parent](#) () const
 - Returns a pointer to the parent widget.*
 - void [parent](#) ([FI_Group](#) *p)
 - Internal use only - "for hacks only".*
 - void [position](#) (int X, int Y)
 - Repositions the window or widget.*
 - void [redraw](#) ()
 - Schedules the drawing of the widget.*
 - void [redraw_label](#) ()
 - Schedules the drawing of the label.*
 - virtual void [resize](#) (int x, int y, int w, int h)
 - Changes the size or position of the widget.*
 - [FI_Color selection_color](#) () const
 - Gets the selection color.*
 - void [selection_color](#) ([FI_Color](#) a)
 - Sets the selection color.*
 - void [set_active](#) ()
 - Marks the widget as active without sending events or changing focus.*
 - void [set_changed](#) ()
 - Marks the value of the widget as changed.*
 - void [set_output](#) ()
 - Sets a widget to output only.*
 - void [set_visible](#) ()
 - Makes the widget visible.*
 - void [set_visible_focus](#) ()
 - Enables keyboard focus navigation with this widget.*

- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)

- Sets the flags used to decide when a callback is called.*
- `FL_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~FL_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the progress bar.

Protected Member Functions inherited from `FL_Widget`

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const`
Draws a focus box for the widget at the given position and size.
- `void draw_label () const`
Draws the widget's label at the defined label position.
- `void draw_label (int, int, int, int) const`
Draws the label in an arbitrary bounding box.
- `FL_Widget (int x, int y, int w, int h, const char *label=0L)`
Creates a widget at the given position and size.
- `unsigned int flags () const`
Gets the widget flags mask.
- `void h (int v)`
Internal use only.
- `void set_flag (unsigned int c)`
Sets a flag in the flags mask.
- `void w (int v)`
Internal use only.
- `void x (int v)`
Internal use only.
- `void y (int v)`
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.108.1 Detailed Description

Displays a progress bar for the user.

11.108.2 Constructor & Destructor Documentation

11.108.2.1 [Fl_Progress\(\)](#)

```
Fl_Progress::Fl_Progress (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

The constructor creates the progress bar using the position, size, and label.

You can set the background color with [color\(\)](#) and the progress bar color with [selection_color\(\)](#), or you can set both colors together with [color\(unsigned bg, unsigned sel\)](#).

The default colors are [FL_BACKGROUND2_COLOR](#) and [FL_YELLOW](#), resp.

11.108.3 Member Function Documentation

11.108.3.1 [draw\(\)](#)

```
void Fl_Progress::draw (
    void ) [protected], [virtual]
```

Draws the progress bar.

Implements [Fl_Widget](#).

11.108.3.2 maximum() [1/2]

```
float Fl_Progress::maximum () const [inline]
```

Gets the maximum value in the progress widget.

11.108.3.3 maximum() [2/2]

```
void Fl_Progress::maximum (  
    float v) [inline]
```

Sets the maximum value in the progress widget.

11.108.3.4 minimum() [1/2]

```
float Fl_Progress::minimum () const [inline]
```

Gets the minimum value in the progress widget.

11.108.3.5 minimum() [2/2]

```
void Fl_Progress::minimum (  
    float v) [inline]
```

Sets the minimum value in the progress widget.

11.108.3.6 value() [1/2]

```
float Fl_Progress::value () const [inline]
```

Gets the current value in the progress widget.

11.108.3.7 value() [2/2]

```
void Fl_Progress::value (  
    float v) [inline]
```

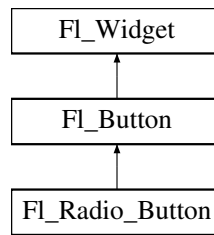
Sets the current value in the progress widget.

The documentation for this class was generated from the following files:

- Fl_Progress.H
- Fl_Progress.cxx

11.109 FI_Radio_Button Class Reference

Inheritance diagram for FI_Radio_Button:



Public Member Functions

- [Fl_Radio_Button](#) (int X, int Y, int W, int H, const char *L=0)
The constructor creates the button using the given position, size, and label.

Public Member Functions inherited from [Fl_Button](#)

- int [clear](#) ()
Same as `value(0)`.
- [uchar compact](#) ()
Return true if buttons are rendered as compact buttons.
- void [compact](#) ([uchar](#) v)
Decide if buttons should be rendered in compact mode.
- [Fl_Boxtype down_box](#) () const
Returns the current down box type, which is drawn when `value()` is non-zero.
- void [down_box](#) ([Fl_Boxtype](#) b)
Sets the down box type.
- [Fl_Color down_color](#) () const
(for backwards compatibility)
- void [down_color](#) (unsigned c)
(for backwards compatibility)
- [Fl_Button](#) (int X, int Y, int W, int H, const char *L=0)
The constructor creates the button using the given position, size, and label.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- int [set](#) ()
Same as `value(1)`.
- void [setonly](#) ()
Turns on this button and turns off all other radio buttons in the group (calling `value(1)` or `set()` does not do this).
- int [shortcut](#) () const
Returns the current shortcut key for the button.
- void [shortcut](#) (const char *s)
(for backwards compatibility)
- void [shortcut](#) (int s)
Sets the shortcut key to `s`.
- char [value](#) () const
Returns the current value of the button (0 or 1).
- int [value](#) (int v)
Sets the current value of the button.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color color` () const

Gets the background color of the widget.
- void `color` (FL_Color bg)

Sets the background color of the widget.
- void `color` (FL_Color bg, FL_Color sel)

Sets the background and selection color of the widget.
- `FL_Color color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const FL_Widget *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([FL_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from FI_Button

- void **draw** () **FL_OVERRIDE**
Draws the widget.
- void **simulate_key_action** ()

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.

- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [Fl_Widget](#) (int [x](#), int [y](#), int [w](#), int [h](#), const char *[label](#)=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void [h](#) (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void [w](#) (int v)
Internal use only.
- void [x](#) (int v)
Internal use only.
- void [y](#) (int v)
Internal use only.

Static Protected Member Functions inherited from [Fl_Button](#)

- static void **key_release_timeout** (void *)

Static Protected Attributes inherited from [Fl_Button](#)

- static [Fl_Widget_Tracker](#) * **key_release_tracker** = 0

11.109.1 Constructor & Destructor Documentation

11.109.1.1 [Fl_Radio_Button](#)()

```
Fl_Radio_Button::Fl_Radio_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

The constructor creates the button using the given position, size, and label. The Button [type\(\)](#) is set to FL_RADIO_BUTTON.

Parameters

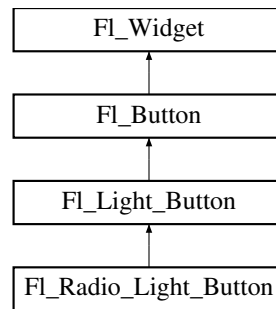
in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

The documentation for this class was generated from the following files:

- [Fl_Radio_Button.H](#)
- [Fl_Button.cxx](#)

11.110 [Fl_Radio_Light_Button](#) Class Reference

Inheritance diagram for [Fl_Radio_Light_Button](#):



Public Member Functions

- **Fl_Radio_Light_Button** (int X, int Y, int W, int H, const char *l=0)

Public Member Functions inherited from **Fl_Light_Button**

- **Fl_Light_Button** (int x, int y, int w, int h, const char *l=0)
*Creates a new **Fl_Light_Button** widget using the given position, size, and label string.*
- int **handle** (int) **FL_OVERRIDE**
Handles the specified event.

Public Member Functions inherited from **Fl_Button**

- int **clear** ()
Same as `value(0)`.
- uchar **compact** ()
Return true if buttons are rendered as compact buttons.
- void **compact** (uchar v)
Decide if buttons should be rendered in compact mode.
- **Fl_Boxtype down_box** () const
Returns the current down box type, which is drawn when `value()` is non-zero.
- void **down_box** (**Fl_Boxtype** b)
Sets the down box type.
- **Fl_Color down_color** () const
(for backwards compatibility)
- void **down_color** (unsigned c)
(for backwards compatibility)
- **Fl_Button** (int X, int Y, int W, int H, const char *L=0)
The constructor creates the button using the given position, size, and label.
- int **set** ()
Same as `value(1)`.
- void **setonly** ()
Turns on this button and turns off all other radio buttons in the group (calling `value(1)` or `set()` does not do this).
- int **shortcut** () const
Returns the current shortcut key for the button.
- void **shortcut** (const char *s)
(for backwards compatibility)
- void **shortcut** (int s)
Sets the shortcut key to s.
- char **value** () const
Returns the current value of the button (0 or 1).
- int **value** (int v)
Sets the current value of the button.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color color` () const

Gets the background color of the widget.
- void `color` (FL_Color bg)

Sets the background color of the widget.
- void `color` (FL_Color bg, FL_Color sel)

Sets the background and selection color of the widget.
- `FL_Color color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const FL_Widget *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([FL_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from FI_Light_Button

- void **draw** () **FL_OVERRIDE**
Draws the widget.

Protected Member Functions inherited from FI_Button

- void **simulate_key_action** ()

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const

- Draws a focus box for the widget at the given position and size.*
 - void `draw_label` () const
- Draws the widget's label at the defined label position.*
 - void `draw_label` (int, int, int, int) const
- Draws the label in an arbitrary bounding box.*
 - `Fl_Widget` (int `x`, int `y`, int `w`, int `h`, const char *`label`=0L)
- Creates a widget at the given position and size.*
 - unsigned int `flags` () const
- Gets the widget flags mask.*
 - void `h` (int `v`)
- Internal use only.*
 - void `set_flag` (unsigned int `c`)
- Sets a flag in the flags mask.*
 - void `w` (int `v`)
- Internal use only.*
 - void `x` (int `v`)
- Internal use only.*
 - void `y` (int `v`)

Static Protected Member Functions inherited from `Fl_Button`

- static void `key_release_timeout` (void *)

Static Protected Attributes inherited from `Fl_Button`

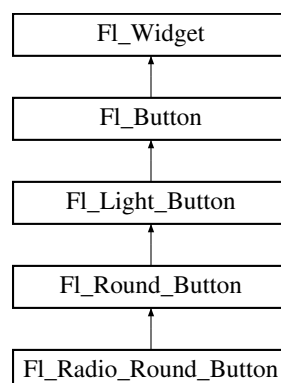
- static `Fl_Widget_Tracker` * `key_release_tracker` = 0

The documentation for this class was generated from the following files:

- `Fl_Radio_Light_Button.H`
- `Fl_Light_Button.cxx`

11.111 `Fl_Radio_Round_Button` Class Reference

Inheritance diagram for `Fl_Radio_Round_Button`:



Public Member Functions

- `Fl_Radio_Round_Button` (int `X`, int `Y`, int `W`, int `H`, const char *`L`=0)
- Creates a new `Fl_Radio_Button` widget using the given position, size, and label string.*

Public Member Functions inherited from FI_Round_Button

- [FI_Round_Button](#) (int *x*, int *y*, int *w*, int *h*, const char **l*=0)
Creates a new FI_Round_Button widget using the given position, size, and label string.

Public Member Functions inherited from FI_Light_Button

- [FI_Light_Button](#) (int *x*, int *y*, int *w*, int *h*, const char **l*=0)
Creates a new FI_Light_Button widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from FI_Button

- int [clear](#) ()
Same as `value(0)`.
- [uchar compact](#) ()
Return true if buttons are rendered as compact buttons.
- void [compact](#) ([uchar](#) *v*)
Decide if buttons should be rendered in compact mode.
- [FI_Boxtype down_box](#) () const
Returns the current down box type, which is drawn when `value()` is non-zero.
- void [down_box](#) ([FI_Boxtype](#) *b*)
Sets the down box type.
- [FI_Color down_color](#) () const
(for backwards compatibility)
- void [down_color](#) (unsigned *c*)
(for backwards compatibility)
- [FI_Button](#) (int *X*, int *Y*, int *W*, int *H*, const char **L*=0)
The constructor creates the button using the given position, size, and label.
- int [set](#) ()
Same as `value(1)`.
- void [setonly](#) ()
Turns on this button and turns off all other radio buttons in the group (calling `value(1)` or `set()` does not do this).
- int [shortcut](#) () const
Returns the current shortcut key for the button.
- void [shortcut](#) (const char **s*)
(for backwards compatibility)
- void [shortcut](#) (int *s*)
*Sets the shortcut key to *s*.*
- char [value](#) () const
Returns the current value of the button (0 or 1).
- int [value](#) (int *v*)
Sets the current value of the button.

Public Member Functions inherited from FI_Widget

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.

- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [Fl_Align](#) align () const
Gets the label alignment.
- void [align](#) ([Fl_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()
Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [Fl_Group](#) * [as_group](#) ()
Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).
- virtual [Fl_Group](#) const * [as_group](#) () const
- virtual [Fl_Window](#) * [as_window](#) ()
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- virtual [Fl_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype](#) box () const
Gets the box type of the widget.
- void [box](#) ([Fl_Boxtype](#) new_box)
Sets the box type for the widget.
- [Fl_Callback_p](#) callback () const
Gets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([Fl_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.

- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FL_Color` bg)
Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)
Sets the background and selection color of the widget.
- `FL_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FL_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FL_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const

- Gets the widget height.*

 - virtual void `hide` ()
- Makes a widget invisible.*

 - int `horizontal_label_margin` ()

Get the spacing between the label and the horizontal edge of the widget.

 - void `horizontal_label_margin` (int px)

Set the spacing between the label and the horizontal edge of the widget.

 - `FL_Image` * `image` ()

Gets the image that is used as part of the widget label when in the active state.

 - const `FL_Image` * `image` () const

Gets the image that is used as part of the widget label when in the active state.

 - void `image` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the active state.

 - void `image` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the active state.

 - int `image_bound` () const

Returns whether the image is managed by the widget.

 - int `inside` (const `FL_Widget` *wgt) const

Checks if this widget is a child of wgt.

 - int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.

 - const char * `label` () const

Gets the current label text.

 - void `label` (const char *text)

Sets the current label pointer.

 - void `label` (`FL_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.

 - int `label_image_spacing` ()

Return the gap size between the label and the image.

 - void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.

 - `FL_Color` `labelcolor` () const

Gets the label color.

 - void `labelcolor` (`FL_Color` c)

Sets the label color.

 - `FL_Font` `labelfont` () const

Gets the font to use.

 - void `labelfont` (`FL_Font` f)

Sets the font to use.

 - `FL_Fonsize` `labelsize` () const

Gets the font size in pixels.

 - void `labelsize` (`FL_Fonsize` pix)

Sets the font size in pixels.

 - `FL_Labeltype` `labeltype` () const

Gets the label type.

 - void `labeltype` (`FL_Labeltype` a)

Sets the label type.

 - void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.

 - bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.

- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [Fl_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([Fl_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)
Changes the size or position of the widget.
- [Fl_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const

- Finds the x/y offset of the current widget relative to the top-level window.*

 - `uchar type () const`
Gets the widget type.
 - `void type (uchar t)`
Sets the widget type.
 - `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
 - `void * user_data () const`
Gets the user data for this widget.
 - `void user_data (FI_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
 - `void user_data (void *v)`
Sets the user data for this widget.
 - `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
 - `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
 - `unsigned int visible () const`
Returns whether a widget is visible.
 - `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
 - `void visible_focus (int v)`
Modifies keyboard focus navigation.
 - `int visible_r () const`
Returns whether a widget and all its parents are visible.
 - `int w () const`
Gets the widget width.
 - `FI_When when () const`
Returns the conditions under which the callback is called.
 - `void when (uchar i)`
Sets the flags used to decide when a callback is called.
 - `FI_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
 - `int x () const`
Gets the widget position in its window.
 - `int y () const`
Gets the widget position in its window.
 - `virtual ~FI_Widget ()`
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- `static void default_callback (FI_Widget *widget, void *data)`
The default callback for all widgets that don't set a callback.
- `static unsigned int label_shortcut (const char *t)`
Returns the Unicode value of the '&x' shortcut in a given text.
- `static int test_shortcut (const char *, const bool require_alt=false)`
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
 ,
OVERRIDE = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
 ,
MENU_WINDOW = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
 ,
GROUP_RELATIVE = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
 = 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from FI_Light_Button

- void **draw** () **FL_OVERRIDE**

Draws the widget.

Protected Member Functions inherited from FI_Button

- void **simulate_key_action** ()

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)

Internal use only.

- void **set_flag** (unsigned int c)

Sets a flag in the flags mask.

- void **w** (int v)

Internal use only.

- void **x** (int v)

Internal use only.

- void **y** (int v)

Internal use only.

Static Protected Member Functions inherited from [Fl_Button](#)

- static void **key_release_timeout** (void *)

Static Protected Attributes inherited from [Fl_Button](#)

- static [Fl_Widget_Tracker](#) * **key_release_tracker** = 0

11.111.1 Constructor & Destructor Documentation

11.111.1.1 [Fl_Radio_Round_Button](#)()

```
Fl_Radio_Round_Button::Fl_Radio_Round_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Radio_Button](#) widget using the given position, size, and label string. The button [type\(\)](#) is set to FL_RADIO_BUTTON.

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

The documentation for this class was generated from the following files:

- [Fl_Radio_Round_Button.H](#)
- [Fl_Round_Button.cxx](#)

11.112 [Fl_Rect](#) Class Reference

Rectangle with standard FLTK coordinates (X, Y, W, H).

```
#include <Fl_Rect.H>
```

Public Member Functions

- int **b** () const

gets the bottom edge (y + h).

- void **b** (int B)

sets the height based on B and y

- [Fl_Rect](#) ()

The default constructor creates an empty rectangle (x = y = w = h = 0).

- [Fl_Rect](#) (const [Fl_Widget](#) &widget)

This constructor creates a rectangle based on a widget's position and size.

- **Fl_Rect** (const [Fl_Widget](#) *const widget)

This constructor creates a rectangle based on a widget's position and size.

- **Fl_Rect** (int W, int H)

This constructor creates a rectangle with $x = y = 0$ and the given width and height.

- **Fl_Rect** (int X, int Y, int W, int H)

This constructor creates a rectangle with the given x,y coordinates and the given width and height.

- **Fl_Rect** (int X, int Y, int W, int H, [Fl_Boxtype](#) bt)

This constructor creates a rectangle with the given x,y coordinates and the given width and height reduced by the box frame size.

- int **h** () const

gets the height

- void **h** (int H)

sets the height

- void **inset** ([Fl_Boxtype](#) bt)

Move all edges in by the frame size of box type bt .

- void **inset** (int d)

Move all edges in by d .

- void **inset** (int left, int top, int right, int bottom)

Move all edges in by $left, top, right, bottom$.

- int **r** () const

gets the right edge ($x + w$).

- void **r** (int R)

sets the width based on R and x

- int **w** () const

gets the width

- void **w** (int W)

sets the width

- int **x** () const

gets the x coordinate (left edge)

- void **x** (int X)

sets the x coordinate (left edge)

- int **y** () const

gets the y coordinate (top edge)

- void **y** (int Y)

sets the y coordinate (top edge)

Friends

- bool **operator!=** (const [Fl_Rect](#) &lhs, const [Fl_Rect](#) &rhs)
- bool **operator==** (const [Fl_Rect](#) &lhs, const [Fl_Rect](#) &rhs)

11.112.1 Detailed Description

Rectangle with standard FLTK coordinates (X, Y, W, H).

This may be used internally, for overloaded widget constructors and other overloaded methods like [fl_measure\(\)](#), [fl_text_extents\(\)](#), [fl_rect\(\)](#), [fl_rectf\(\)](#), and maybe more.

11.112.2 Constructor & Destructor Documentation

11.112.2.1 Fl_Rect()

```
Fl_Rect::Fl_Rect (
    int X,
    int Y,
    int W,
    int H,
    Fl_Boxtype bt) [inline]
```

This constructor creates a rectangle with the given x,y coordinates and the given width and height reduced by the box frame size.

This is the same as using the constructor w/o `bt` and subsequently calling `inset(bt)`.

11.112.3 Member Function Documentation

11.112.3.1 b()

```
int Fl_Rect::b () const [inline]
```

gets the bottom edge (`y + h`).

Note

`r()` and `b()` are coordinates **outside** the area of the rectangle.

11.112.3.2 inset() [1/3]

```
void Fl_Rect::inset (
    Fl_Boxtype bt) [inline]
```

Move all edges in by the frame size of box type `bt`.

Shrinks the rectangle at all sides by the frame width or height of the given box type `bt`.

This method uses the frame sizes given by the box type `bt` using

- `Fl::box_dx(bt)`
- `Fl::box_dy(bt)`
- `Fl::box_dw(bt)`
- `Fl::box_dh(bt)`

If the rectangle is smaller than the frame sizes the result is undefined, i.e. an invalid or empty rectangle.

11.112.3.3 inset() [2/3]

```
void Fl_Rect::inset (
    int d) [inline]
```

Move all edges in by `d`.

Shrinks the rectangle by `d` at all sides keeping the center of the rectangle at the same spot.

If `d` is negative, the rectangle is enlarged.

If `d >= w()` or `h()` the result is undefined, i.e. an invalid or empty rectangle.

11.112.3.4 inset() [3/3]

```
void Fl_Rect::inset (
    int left,
    int top,
    int right,
    int bottom) [inline]
```

Move all edges in by `left`, `top`, `right`, `bottom`.

Shrinks the rectangle on all sides keeping the center of the rectangle at the same spot.

If any value is negative, the rectangle is enlarged.

Values are not range checked; it is possible to create an invalid or empty rectangle.

11.112.3.5 r()

```
int FI_Rect::r () const [inline]
gets the right edge (x + w).
```

Note

r() and **b()** are coordinates **outside** the area of the rectangle.

The documentation for this class was generated from the following file:

- FI_Rect.H

11.113 FI_Scroll::FI_Region_LRTB Struct Reference

A local struct to manage a region defined by left/right/top/bottom.

```
#include <FI_Scroll.H>
```

Public Attributes

- int **b**
(b)ottom "y" position, aka y2
- int **l**
(l)eft "x" position, aka x1
- int **r**
(r)ight "x" position, aka x2
- int **t**
(t)op "y" position, aka y1

11.113.1 Detailed Description

A local struct to manage a region defined by left/right/top/bottom.

The documentation for this struct was generated from the following file:

- FI_Scroll.H

11.114 FI_Scroll::FI_Region_XYWH Struct Reference

A local struct to manage a region defined by xywh.

```
#include <FI_Scroll.H>
```

Public Attributes

- int **h**
- int **w**
- int **x**
- int **y**

11.114.1 Detailed Description

A local struct to manage a region defined by xywh.

The documentation for this struct was generated from the following file:

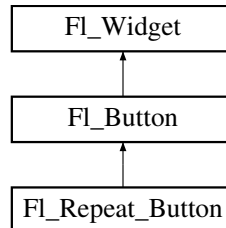
- FI_Scroll.H

11.115 Fl_Repeat_Button Class Reference

The [Fl_Repeat_Button](#) is a subclass of [Fl_Button](#) that generates a callback when it is pressed and then repeatedly generates callbacks as long as it is held down.

```
#include <Fl_Repeat_Button.H>
```

Inheritance diagram for [Fl_Repeat_Button](#):



Public Member Functions

- void **deactivate** ()
- [Fl_Repeat_Button](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [Fl_Repeat_Button](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [Fl_Button](#)

- int [clear](#) ()
Same as `value(0)`.
- [uchar compact](#) ()
Return true if buttons are rendered as compact buttons.
- void [compact](#) ([uchar](#) v)
Decide if buttons should be rendered in compact mode.
- [Fl_Boxtype down_box](#) () const
Returns the current down box type, which is drawn when `value()` is non-zero.
- void [down_box](#) ([Fl_Boxtype](#) b)
Sets the down box type.
- [Fl_Color down_color](#) () const
(for backwards compatibility)
- void [down_color](#) (unsigned c)
(for backwards compatibility)
- [Fl_Button](#) (int X, int Y, int W, int H, const char *L=0)
The constructor creates the button using the given position, size, and label.
- int [set](#) ()
Same as `value(1)`.
- void [setonly](#) ()
Turns on this button and turns off all other radio buttons in the group (calling `value(1)` or `set()` does not do this).
- int [shortcut](#) () const
Returns the current shortcut key for the button.
- void [shortcut](#) (const char *s)
(for backwards compatibility)
- void [shortcut](#) (int s)
Sets the shortcut key to `s`.
- char [value](#) () const
Returns the current value of the button (0 or 1).
- int [value](#) (int v)
Sets the current value of the button.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (FI_Align alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class FI_GL_Window * **as_gl_window** ()
Returns an FI_GL_Window pointer if this widget is an FI_GL_Window.
- virtual class FI_GL_Window const * **as_gl_window** () const
- virtual FI_Group * **as_group** ()
Returns an FI_Group pointer if this widget is an FI_Group.
- virtual FI_Group const * **as_group** () const
- virtual FI_Window * **as_window** ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual FI_Window const * **as_window** () const
- void **bind_deimage** (FI_Image *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (FI_Image *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- **FI_Boxtype box** () const
Gets the box type of the widget.
- void **box** (FI_Boxtype new_box)
Sets the box type for the widget.
- **FI_Callback_p callback** () const
Gets the current callback function for the widget.
- void **callback** (FI_Callback *cb)
Sets the current callback function for the widget.
- void **callback** (FI_Callback *cb, FI_Callback_User_Data *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void **callback** (FI_Callback *cb, void *p)
Sets the current callback function and data for the widget.
- void **callback** (FI_Callback0 *cb)
Sets the current callback function for the widget.
- void **callback** (FI_Callback1 *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int **changed** () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FL_Color` bg)

Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)

Sets the background and selection color of the widget.
- `FL_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.

- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [Fl_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([Fl_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [Fl_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Widget](#)

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from [FI_Button](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void [simulate_key_action](#) ()

Protected Member Functions inherited from [FI_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.

- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [Fl_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Static Protected Member Functions inherited from [Fl_Button](#)

- static void **key_release_timeout** (void *)

Static Protected Attributes inherited from [Fl_Button](#)

- static [Fl_Widget_Tracker](#) * **key_release_tracker** = 0

11.115.1 Detailed Description

The [Fl_Repeat_Button](#) is a subclass of [Fl_Button](#) that generates a callback when it is pressed and then repeatedly generates callbacks as long as it is held down.

The speed of the repeat is fixed and depends on the implementation.

11.115.2 Constructor & Destructor Documentation

11.115.2.1 [Fl_Repeat_Button](#)()

```
Fl_Repeat_Button::Fl_Repeat_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Repeat_Button](#) widget using the given position, size, and label string. The default boxtype is FL_UP_BOX. Deletes the button.

11.115.3 Member Function Documentation

11.115.3.1 [handle](#)()

```
int Fl_Repeat_Button::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

<code>in</code>	<code>event</code>	the kind of event received

Return values

<code>0</code>	if the event was not used or understood
<code>1</code>	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Button](#).

The documentation for this class was generated from the following files:

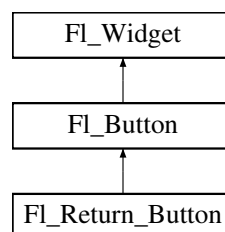
- [Fl_Repeat_Button.H](#)
- [Fl_Repeat_Button.cxx](#)

11.116 Fl_Return_Button Class Reference

The [Fl_Return_Button](#) is a subclass of [Fl_Button](#) that generates a callback when it is pressed or when the user presses the Enter key.

```
#include <Fl_Return_Button.H>
```

Inheritance diagram for [Fl_Return_Button](#):



Public Member Functions

- [Fl_Return_Button](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [Fl_Return_Button](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [Fl_Button](#)

- int [clear](#) ()
Same as `value(0)`.
- [uchar compact](#) ()
Return true if buttons are rendered as compact buttons.
- void [compact](#) ([uchar](#) v)

- Decide if buttons should be rendered in compact mode.*
- [FL_Boxtype](#) **down_box** () const
Returns the current down box type, which is drawn when [value\(\)](#) is non-zero.
- void **down_box** ([FL_Boxtype](#) b)
Sets the down box type.
- [FL_Color](#) **down_color** () const
(for backwards compatibility)
- void **down_color** (unsigned c)
(for backwards compatibility)
- [FL_Button](#) (int X, int Y, int W, int H, const char *L=0)
The constructor creates the button using the given position, size, and label.
- int **set** ()
Same as [value\(1\)](#).
- void **setonly** ()
Turns on this button and turns off all other radio buttons in the group (calling [value\(1\)](#) or [set\(\)](#) does not do this).
- int **shortcut** () const
Returns the current shortcut key for the button.
- void **shortcut** (const char *s)
(for backwards compatibility)
- void **shortcut** (int s)
Sets the shortcut key to s.
- char **value** () const
Returns the current value of the button (0 or 1).
- int **value** (int v)
Sets the current value of the button.

Public Member Functions inherited from [FL_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- [FL_Align](#) **align** () const
Gets the label alignment.
- void **align** ([FL_Align](#) alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FL_Gl_Window](#) * **as_gl_window** ()
Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).
- virtual class [FL_Gl_Window](#) const * **as_gl_window** () const
- virtual [FL_Group](#) * **as_group** ()
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- virtual [FL_Group](#) const * **as_group** () const
- virtual [FL_Window](#) * **as_window** ()

- Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).*

 - virtual [FL_Window](#) const * **as_window** () const
- void [bind_deimage](#) ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FL_Boxtype](#) **box** () const

Gets the box type of the widget.
- void [box](#) ([FL_Boxtype](#) new_box)

Sets the box type for the widget.
- [FL_Callback_p](#) **callback** () const

Gets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FL_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.
- void [callback](#) ([FL_Callback0](#) *cb)

Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()

Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.
- void [clear_output](#) ()

Sets a widget to accept input.
- void [clear_visible](#) ()

Hides the widget.
- void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.
- [FL_Color](#) **color** () const

Gets the background color of the widget.
- void [color](#) ([FL_Color](#) bg)

Sets the background color of the widget.
- void [color](#) ([FL_Color](#) bg, [FL_Color](#) sel)

Sets the background and selection color of the widget.
- [FL_Color](#) **color2** () const

For back compatibility only.
- void [color2](#) (unsigned a)

For back compatibility only.
- int [contains](#) (const [FL_Widget](#) *w) const

- Checks if w is a child of this widget.*

 - void [copy_label](#) (const char *new_label)

Sets the current label.

- void [copy_tooltip](#) (const char *text)

Sets the current tooltip text.

- [uchar damage](#) () const

Returns non-zero if [draw\(\)](#) needs to be called.

- void [damage](#) (uchar c)

Sets the damage bits for the widget.

- void [damage](#) (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.

- int [damage_resize](#) (int, int, int, int)

Internal use only.

- void [deactivate](#) ()

Deactivates the widget.

- [FL_Image * deimage](#) ()

Gets the image that is used as part of the widget label when in the inactive state.

- const [FL_Image * deimage](#) () const

Gets the image that is used as part of the widget label when in the inactive state.

- void [deimage](#) ([FL_Image](#) &img)

Sets the image to use as part of the widget label when in the inactive state.

- void [deimage](#) ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the inactive state.

- int [deimage_bound](#) () const

Returns whether the inactive image is managed by the widget.

- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))

Calls the widget callback function with default arguments.

- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))

Calls the widget callback function with arbitrary arguments.

- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))

Calls the widget callback function with arbitrary arguments.

- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.

- int [h](#) () const

Gets the widget height.

- virtual void [hide](#) ()

Makes a widget invisible.

- int [horizontal_label_margin](#) ()

Get the spacing between the label and the horizontal edge of the widget.

- void [horizontal_label_margin](#) (int px)

Set the spacing between the label and the horizontal edge of the widget.

- [FL_Image * image](#) ()

Gets the image that is used as part of the widget label when in the active state.

- const [FL_Image * image](#) () const

Gets the image that is used as part of the widget label when in the active state.

- void [image](#) ([FL_Image](#) &img)

Sets the image to use as part of the widget label when in the active state.

- void [image](#) ([FL_Image](#) *img)

Sets the image to use as part of the widget label when in the active state.

- int [image_bound](#) () const

Returns whether the image is managed by the widget.

- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const
Gets the label type.
- void `labeltype` (`FL_Labeltype` a)
Sets the label type.
- void `measure_label` (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const
Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int `output` () const
Returns if a widget is used for output only.
- `FL_Group` * `parent` () const
Returns a pointer to the parent widget.
- void `parent` (`FL_Group` *p)
Internal use only - "for hacks only".
- void `position` (int X, int Y)
Repositions the window or widget.
- void `redraw` ()
Schedules the drawing of the widget.
- void `redraw_label` ()
Schedules the drawing of the label.
- virtual void `resize` (int x, int y, int w, int h)
Changes the size or position of the widget.
- `FL_Color` `selection_color` () const

- Gets the selection color.*

 - void **selection_color** (**FI_Color** a)

Sets the selection color.
- void **set_active** ()

Marks the widget as active without sending events or changing focus.
- void **set_changed** ()

Marks the value of the widget as changed.
- void **set_output** ()

Sets a widget to output only.
- void **set_visible** ()

Makes the widget visible.
- void **set_visible_focus** ()

Enables keyboard focus navigation with this widget.
- int **shortcut_label** () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void **shortcut_label** (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void **show** ()

Makes a widget visible.
- void **size** (int W, int H)

Changes the size of the widget.
- int **take_focus** ()

Gives the widget the keyboard focus.
- unsigned int **takeevents** () const

Returns if the widget is able to take events.
- int **test_shortcut** ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * **tooltip** () const

Gets the current tooltip text.
- void **tooltip** (const char *text)

Sets the current tooltip text.
- **FI_Window** * **top_window** () const

Returns a pointer to the top-level window for the widget.
- **FI_Window** * **top_window_offset** (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- **uchar** **type** () const

Gets the widget type.
- void **type** (**uchar** t)

Sets the widget type.
- int **use_accents_menu** ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const

Gets the user data for this widget.
- void **user_data** (**FI_Callback_User_Data** *v, bool auto_free)

Sets the user data for this widget.
- void **user_data** (void *v)

Sets the user data for this widget.
- int **vertical_label_margin** ()

Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)

Set the spacing between the label and the vertical edge of the widget.

- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **FL_When** **when** () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **FL_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~FL_Widget** ()
Destroys the widget.

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
Draws the widget.

Protected Member Functions inherited from **FL_Button**

- void **simulate_key_action** ()

Protected Member Functions inherited from **FL_Widget**

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FL_Boxtype t, FL_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FL_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.

- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `FI_Widget` (int `x`, int `y`, int `w`, int `h`, const char *`label`=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int `v`)
Internal use only.
- void `set_flag` (unsigned int `c`)
Sets a flag in the flags mask.
- void `w` (int `v`)
Internal use only.
- void `x` (int `v`)
Internal use only.
- void `y` (int `v`)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *`widget`, void *`data`)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *`t`)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *`t`, const bool `require_alt`=false)
Returns true if the given text `t` contains the entered '&x' shortcut.

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Static Protected Member Functions inherited from `FI_Button`

- static void `key_release_timeout` (void *)

Static Protected Attributes inherited from `FI_Button`

- static `FI_Widget_Tracker` * `key_release_tracker` = 0

11.116.1 Detailed Description

The `Fl_Return_Button` is a subclass of `Fl_Button` that generates a callback when it is pressed or when the user presses the Enter key.

A carriage-return symbol is drawn next to the button label.

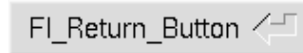


Figure 11.37 `Fl_Return_Button`

11.116.2 Constructor & Destructor Documentation

11.116.2.1 `Fl_Return_Button()`

```
Fl_Return_Button::Fl_Return_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new `Fl_Return_Button` widget using the given position, size, and label string.

The default boxtype is `FL_UP_BOX`.

The inherited destructor deletes the button.

11.116.3 Member Function Documentation

11.116.3.1 `draw()`

```
void Fl_Return_Button::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call `redraw()` instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own `draw()` method*, e.g. for an embedded scrollbar, you can do it (because `draw()` is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Reimplemented from `Fl_Button`.

11.116.3.2 `handle()`

```
int Fl_Return_Button::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited `handle()` method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your `handle()` method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the `handle()` method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Button](#).

The documentation for this class was generated from the following files:

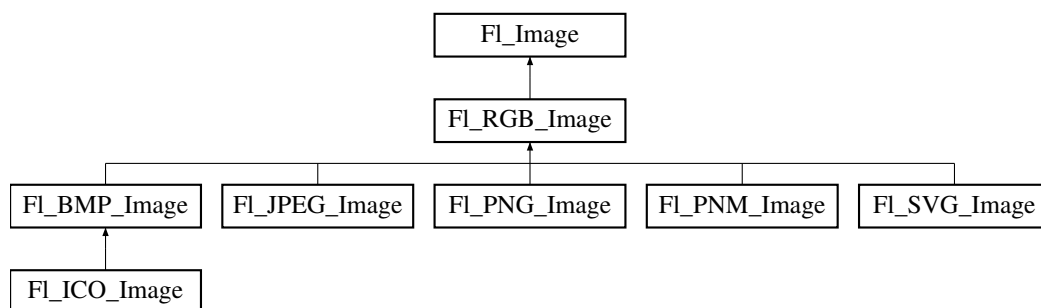
- [Fl_Return_Button.H](#)
- [Fl_Return_Button.cxx](#)

11.117 Fl_RGB_Image Class Reference

The [Fl_RGB_Image](#) class supports caching and drawing of full-color images with 1 to 4 channels of color information.

```
#include <Fl_Image.H>
```

Inheritance diagram for [Fl_RGB_Image](#):



Public Member Functions

- virtual [Fl_SVG_Image](#) * [as_svg_image](#) ()
Returns whether an image is an [Fl_SVG_Image](#) or not.
- int **cache_h** ()
- int **cache_w** ()
- void [color_average](#) ([Fl_Color](#) c, float i) [FL_OVERRIDE](#)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [Fl_Image](#) * **copy** () const
- [Fl_Image](#) * [copy](#) (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void [desaturate](#) () [FL_OVERRIDE](#)
The [desaturate\(\)](#) method converts an image to grayscale.
- void **draw** (int X, int Y)
- void [draw](#) (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [Fl_RGB_Image](#) (const [Fl_Pixmap](#) *pxm, [Fl_Color](#) bg=FL_GRAY)
The constructor creates a new RGBA image from the specified [Fl_Pixmap](#).
- [Fl_RGB_Image](#) (const [uchar](#) *bits, int bits_length, int W, int H, int D, int LD)
The constructor creates a new image from the specified data.
- [Fl_RGB_Image](#) (const [uchar](#) *bits, int W, int H, int D=3, int LD=0)
The constructor creates a new image from the specified data.

- void [label](#) ([Fl_Menu_Item](#) *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.
- void [label](#) ([Fl_Widget](#) *w) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a widget or menu item.
- virtual void [normalize](#) ()
Makes sure the object is fully initialized.
- void [uncache](#) () [FL_OVERRIDE](#)
If the image has been cached for display, delete the cache data.
- [~Fl_RGB_Image](#) () [FL_OVERRIDE](#)
The destructor frees all memory and server resources that are used by the image.

Public Member Functions inherited from [Fl_Image](#)

- virtual class [Fl_Shared_Image](#) * [as_shared_image](#) ()
Returns whether an image is an [Fl_Shared_Image](#) or not.
- [Fl_Image](#) * [copy](#) () const
Creates a copy of the image in the same size.
- int [count](#) () const
Returns the number of data values associated with the image.
- int [d](#) () const
Returns the image depth.
- const char *const * [data](#) () const
Returns a pointer to the current image data array.
- int [data_h](#) () const
Returns the height of the image data.
- int [data_w](#) () const
Returns the width of the image data.
- void [draw](#) (int X, int Y)
Draws the image to the current drawing surface.
- int [fail](#) () const
Returns a value that is not 0 if there is currently no image available.
- [Fl_Image](#) (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int [h](#) () const
Returns the current image drawing height in FLTK units.
- void [inactive](#) ()
The [inactive\(\)](#) method calls [color_average\(FL_BACKGROUND_COLOR, 0.33f\)](#) to produce an image that appears grayed out.
- int [ld](#) () const
Returns the current line data size in bytes.
- virtual void [release](#) ()
Releases an [Fl_Image](#) - the same as 'delete this'.
- virtual void [scale](#) (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- int [w](#) () const
Returns the current image drawing width in FLTK units.
- virtual [~Fl_Image](#) ()
The destructor is a virtual method that frees all memory used by the image.

Static Public Member Functions

- static `size_t max_size ()`
Returns the maximum allowed image size in bytes when creating an [FI_RGB_Image](#) object.
- static void `max_size (size_t size)`
Sets the maximum allowed image size in bytes when creating an [FI_RGB_Image](#) object.

Static Public Member Functions inherited from [FI_Image](#)

- static `FI_Labeltype define_FL_IMAGE_LABEL ()`
- static `FI_RGB_Scaling RGB_scaling ()`
Returns the currently used RGB image scaling method.
- static void `RGB_scaling (FI_RGB_Scaling)`
Sets the RGB image scaling method used for copy(int, int).
- static `FI_RGB_Scaling scaling_algorithm ()`
Gets what algorithm is used when resizing a source image to draw it.
- static void `scaling_algorithm (FI_RGB_Scaling algorithm)`
Sets what algorithm is used when resizing a source image to draw it.

Public Attributes

- int `alloc_array`
If non-zero, the object's data array is delete[]'d when deleting the object.
- const `uchar * array`
Points to the start of the object's data array.

Friends

- class `FI_Graphics_Driver`

Additional Inherited Members

Static Public Attributes inherited from [FI_Image](#)

- static const int `ERR_FILE_ACCESS = -2`
- static const int `ERR_FORMAT = -3`
- static const int `ERR_MEMORY_ACCESS = -4`
- static const int `ERR_NO_IMAGE = -1`
- static bool `register_images_done = false`
True after [fi_register_images\(\)](#) was called, false before.

Protected Member Functions inherited from [FI_Image](#)

- void `d (int D)`
Sets the current image depth.
- void `data (const char *const *p, int c)`
Sets the current data pointer and count of pointers in the array.
- void `draw_empty (int X, int Y)`
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int `draw_scaled (int X, int Y, int W, int H)`
Draw the image to the current drawing surface rescaled to a given width and height.
- void `h (int H)`
Sets the height of the image data.
- void `ld (int LD)`
Sets the current line data size in bytes.
- void `w (int W)`
Sets the width of the image data.

Static Protected Member Functions inherited from [FI_Image](#)

- static void **labeltype** (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void **measure** (const [FI_Label](#) *lo, int &lw, int &lh)

11.117.1 Detailed Description

The [FI_RGB_Image](#) class supports caching and drawing of full-color images with 1 to 4 channels of color information.

Images with an even number of channels are assumed to contain alpha information, which is used to blend the image with the contents of the screen.

[FI_RGB_Image](#) is defined in [<FI/Image.H>](#), however for compatibility reasons [<FI/FI_RGB_Image.H>](#) should be included.

11.117.2 Constructor & Destructor Documentation

11.117.2.1 [FI_RGB_Image](#)() [1/3]

```
FI_RGB_Image::FI_RGB_Image (
    const uchar * bits,
    int W,
    int H,
    int D = 3,
    int LD = 0)
```

The constructor creates a new image from the specified data.

The data array `bits` must contain sufficient data to provide $W * H * D$ image bytes and optional line padding, see `LD`.

`W` and `H` are the width and height of the image in pixels, resp.

`D` is the image depth and can be:

- `D=1`: each uchar in `bits[]` is a grayscale pixel value
- `D=2`: each uchar pair in `bits[]` is a grayscale + alpha pixel value
- `D=3`: each uchar triplet in `bits[]` is an R/G/B pixel value
- `D=4`: each uchar quad in `bits[]` is an R/G/B/A pixel value

`LD` specifies the line data size of the array, see [FI_Image::ld\(int\)](#). If `LD` is zero, then $W * D$ is assumed, otherwise `LD` must be greater than or equal to $W * D$ to account for (unused) extra data per line (padding).

The caller is responsible that the image data array `bits` persists as long as the image is used.

This constructor sets [FI_RGB_Image::alloc_array](#) to 0. To have the image object control the deallocation of the data array `bits`, set `alloc_array` to non-zero after construction.

Parameters

in	<i>bits</i>	The image data array.
in	<i>W</i>	The width of the image in pixels.
in	<i>H</i>	The height of the image in pixels.
in	<i>D</i>	The image depth, or 'number of channels' (default=3).
in	<i>LD</i>	Line data size (default=0).

See also

[FI_Image::data\(\)](#), [FI_Image::w\(\)](#), [FI_Image::h\(\)](#), [FI_Image::d\(\)](#), [FI_Image::ld\(int\)](#)

11.117.2.2 FI_RGB_Image() [2/3]

```
Fl_RGB_Image::Fl_RGB_Image (
    const uchar * bits,
    int bits_length,
    int W,
    int H,
    int D,
    int LD)
```

The constructor creates a new image from the specified data.

If the provided array is too small to contain all the image data, the constructor will not generate the image to avoid illegal memory read access and instead set `data` to `NULL` and `ld` to `ERR_MEMORY_ACCESS`.

Parameters

<i>bits</i>	image data
<i>bits_length</i>	length of the <code>bits</code> array in bytes
<i>W</i>	image width in pixels
<i>H</i>	image height in pixels
<i>D</i>	image depth in bytes, 1 for gray scale, 2 for gray with alpha, 3 for RGB, and 4 for RGB plus alpha
<i>LD</i>	line length in bytes, or 0 to use <code>W*D</code> .

See also

[FI_RGB_Image\(const uchar *bits, int W, int H, int D, int LD\)](#)

11.117.2.3 FI_RGB_Image() [3/3]

```
Fl_RGB_Image::Fl_RGB_Image (
    const Fl_Pixmap * pxm,
    Fl_Color bg = FL_GRAY)
```

The constructor creates a new RGBA image from the specified [Fl_Pixmap](#).

The RGBA image is built fully opaque except for the transparent area of the pixmap that is assigned the `bg` color with full transparency.

This constructor creates a new internal data array and sets [Fl_RGB_Image::alloc_array](#) to 1 so the data array is deleted when the image is destroyed.

11.117.3 Member Function Documentation**11.117.3.1 as_svg_image()**

```
virtual Fl_SVG_Image * Fl_RGB_Image::as_svg_image () [inline], [virtual]
```

Returns whether an image is an [Fl_SVG_Image](#) or not.

This virtual method returns a pointer to the [Fl_SVG_Image](#) if this object is an instance of [Fl_SVG_Image](#) or `NULL` if not.

Reimplemented in [Fl_SVG_Image](#).

11.117.3.2 color_average()

```
void Fl_RGB_Image::color_average (
    Fl_Color c,
    float i) [virtual]
```

The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.

The first argument specifies the FLTK color to be used.

The second argument specifies the amount of the original image to combine with the color, so a value of 1.0 results in no color blend, and a value of 0.0 results in a constant image of the specified color.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented from [Fl_Image](#).

Reimplemented in [Fl_SVG_Image](#).

11.117.3.3 copy()

```
Fl_Image * Fl_RGB_Image::copy (
    int W,
    int H) const [virtual]
```

Creates a resized copy of the image.

It is recommended not to call this member function to reduce the size of an image to the size of the area where this image will be drawn, and to use [Fl_Image::scale\(\)](#) instead.

The new image should be released when you are done with it.

Note: since FLTK 1.4.0 you can use [Fl_Image::release\(\)](#) for all types of images (i.e. all subclasses of [Fl_Image](#)) instead of operator *delete* for [Fl_Image](#)'s and [Fl_Image::release\(\)](#) for [Fl_Shared_Image](#)'s.

The new image data will be converted to the requested size. RGB images are resized using the algorithm set by [Fl_Image::RGB_scaling\(\)](#).

For the new image the following equations are true:

- `w() == data_w() == W`
- `h() == data_h() == H`

Parameters

in	<i>W,H</i>	Requested width and height of the new image

Note

The returned image can be safely cast to the same image type as that of the source image provided this type is one of [Fl_RGB_Image](#), [Fl_SVG_Image](#), [Fl_Pixmap](#), [Fl_Bitmap](#), [Fl_Tiled_Image](#), [Fl_Anim_GIF_Image](#) and [Fl_Shared_Image](#). Returned objects copied from images of other, derived, image classes belong to the parent class appearing in this list. For example, the copy of an [Fl_GIF_Image](#) is an object of class [Fl_Pixmap](#).

Since FLTK 1.4.0 this method is 'const'. If you derive your own class from [Fl_Image](#) or any subclass your overridden methods of '[Fl_Image::copy\(\) const](#)' and '[Fl_Image::copy\(int, int\) const](#)' **must** also be 'const' for inheritance to work properly. This is different than in FLTK 1.3.x and earlier where these methods have not been 'const'.

Reimplemented from [Fl_Image](#).

Reimplemented in [Fl_SVG_Image](#).

11.117.3.4 desaturate()

```
void Fl_RGB_Image::desaturate () [virtual]
```

The [desaturate\(\)](#) method converts an image to grayscale.

If the image contains an alpha channel (depth = 4), the alpha channel is preserved.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented from [Fl_Image](#).

Reimplemented in [Fl_SVG_Image](#).

11.117.3.5 draw()

```
void Fl_RGB_Image::draw (
    int X,
    int Y,
    int W,
    int H,
    int cx = 0,
    int cy = 0) [virtual]
```

Draws the image to the current drawing surface with a bounding box.

Arguments *X*, *Y*, *W*, *H* specify a bounding box for the image, with the origin (upper-left corner) of the image offset by the *cx* and *cy* arguments.

In other words: `fl_push_clip(X,Y,W,H)` is applied, the image is drawn with its upper-left corner at *X-cx*, *Y-cy* and its own width and height, `fl_pop_clip()` is applied.

Reimplemented from [Fl_Image](#).

Reimplemented in [Fl_SVG_Image](#).

11.117.3.6 label() [1/2]

```
void Fl_RGB_Image::label (
    Fl_Menu_Item * m) [virtual]
```

This method is an obsolete way to set the image attribute of a menu item.

Deprecated Please use [Fl_Menu_Item::image\(\)](#) instead.

Reimplemented from [Fl_Image](#).

11.117.3.7 label() [2/2]

```
void Fl_RGB_Image::label (
    Fl_Widget * widget) [virtual]
```

This method is an obsolete way to set the image attribute of a widget or menu item.

Deprecated Please use [Fl_Widget::image\(\)](#) or [Fl_Widget::deimage\(\)](#) instead.

Reimplemented from [Fl_Image](#).

11.117.3.8 max_size() [1/2]

```
static size_t Fl_RGB_Image::max_size () [inline], [static]
```

Returns the maximum allowed image size in bytes when creating an [Fl_RGB_Image](#) object.

See also

```
void Fl_RGB_Image::max_size(size_t)
```

11.117.3.9 max_size() [2/2]

```
static void Fl_RGB_Image::max_size (
    size_t size) [inline], [static]
```

Sets the maximum allowed image size in bytes when creating an [Fl_RGB_Image](#) object.

The image size in bytes of an [Fl_RGB_Image](#) object is the value of the product `w() * h() * d()`. If this product exceeds size, the created object of a derived class of [Fl_RGB_Image](#) won't be loaded with the image data. This does not apply to direct RGB image creation with [Fl_RGB_Image::Fl_RGB_Image\(const uchar *bits, int W, int H, int D, int LD\)](#). The default [max_size\(\)](#) value is essentially infinite.

11.117.3.10 normalize()

```
virtual void Fl_RGB_Image::normalize () [inline], [virtual]
```

Makes sure the object is fully initialized.

In particular, makes sure member variable [array](#) is non-null.

Reimplemented in [Fl_SVG_Image](#).

11.117.3.11 uncache()

```
void Fl_RGB_Image::uncache () [virtual]
```

If the image has been cached for display, delete the cache data.

This allows you to change the data used for the image and then redraw it without recreating an image object.

Reimplemented from [Fl_Image](#).

11.117.4 Member Data Documentation

11.117.4.1 array

`const uchar* Fl_RGB_Image::array`

Points to the start of the object's data array.

See also

class [Fl_SVG_Image](#) which delays initialization of this member variable.

The documentation for this class was generated from the following files:

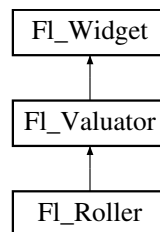
- [Fl_Image.H](#)
- [Fl_Image.cxx](#)

11.118 Fl_Roller Class Reference

The [Fl_Roller](#) widget is a "dolly" control commonly used to move 3D objects.

`#include <Fl_Roller.H>`

Inheritance diagram for [Fl_Roller](#):



Public Member Functions

- [Fl_Roller](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Roller](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [Fl_Valuator](#)

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int [format](#) (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double [increment](#) (double, int)
Adds n times the step value to the passed value.
- double [maximum](#) () const
Gets the maximum value for the valuator.
- void [maximum](#) (double a)
Sets the maximum value for the valuator.
- double [minimum](#) () const
Gets the minimum value for the valuator.
- void [minimum](#) (double a)
Sets the minimum value for the valuator.
- void [precision](#) (int digits)

- Sets the step value to $1.0 / 10^{\text{digits}}$.*
- void **range** (double a, double b)
 - Sets the minimum and maximum values for the valuator.*
- double **round** (double)
 - Round the passed value to the nearest step increment.*
- double **step** () const
 - Gets or sets the step value.*
- void **step** (double a, int b)
 - See double [FI_Valuator::step\(\)](#) const*
- void **step** (double s)
 - See double [FI_Valuator::step\(\)](#) const.*
- void **step** (int a)
 - See double [FI_Valuator::step\(\)](#) const*
- double **value** () const
 - Gets the floating point(double) value.*
- int **value** (double)
 - Sets the current value.*
- ~**FI_Valuator** () **FL_OVERRIDE**
 - Destructor is accessible despite protected constructor.*

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
 - Activates the widget.*
- unsigned int **active** () const
 - Returns whether the widget is active.*
- int **active_r** () const
 - Returns whether the widget and all of its parents are active.*
- [FI_Align](#) **align** () const
 - Gets the label alignment.*
- void **align** ([FI_Align](#) alignment)
 - Sets the label alignment.*
- long **argument** () const
 - Gets the current user data (long) argument that is passed to the callback function.*
- void **argument** (long v)
 - Sets the current user data (long) argument that is passed to the callback function.*
- virtual class [FI_Gl_Window](#) * **as_gl_window** ()
 - Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).*
- virtual class [FI_Gl_Window](#) const * **as_gl_window** () const
- virtual [FI_Group](#) * **as_group** ()
 - Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).*
- virtual [FI_Group](#) const * **as_group** () const
- virtual [FI_Window](#) * **as_window** ()
 - Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).*
- virtual [FI_Window](#) const * **as_window** () const
- void **bind_deimage** ([FI_Image](#) *img)
 - Sets the image to use as part of the widget label when in the inactive state.*
- void **bind_deimage** (int f)

- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*

 - void `bind_image` (`FI_Image` *img)

Sets the image to use as part of the widget label when in the active state.
- void `bind_image` (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- `FI_Boxtype` `box` () const

Gets the box type of the widget.
- void `box` (`FI_Boxtype` new_box)

Sets the box type for the widget.
- `FI_Callback_p` `callback` () const

Gets the current callback function for the widget.
- void `callback` (`FI_Callback` *cb)

Sets the current callback function for the widget.
- void `callback` (`FI_Callback` *cb, `FI_Callback_User_Data` *p, bool auto_free)

Sets the current callback function and managed user data for the widget.
- void `callback` (`FI_Callback` *cb, void *p)

Sets the current callback function and data for the widget.
- void `callback` (`FI_Callback0` *cb)

Sets the current callback function for the widget.
- void `callback` (`FI_Callback1` *cb, long p=0)

Sets the current callback function for the widget.
- unsigned int `changed` () const

Checks if the widget value changed since the last callback.
- void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FI_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FI_Color` bg)

Sets the background color of the widget.
- void `color` (`FI_Color` bg, `FI_Color` sel)

Sets the background and selection color of the widget.
- `FI_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FI_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.

- `uchar damage () const`
Returns non-zero if `draw()` needs to be called.
- `void damage (uchar c)`
Sets the damage bits for the widget.
- `void damage (uchar c, int x, int y, int w, int h)`
Sets the damage bits for an area inside the widget.
- `int damage_resize (int, int, int, int)`
Internal use only.
- `void deactivate ()`
Deactivates the widget.
- `FL_Image * deimage ()`
Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`
Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`
Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`
Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`
Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FL_Align) const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`
Gets the widget height.
- `virtual void hide ()`
Makes a widget invisible.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FL_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FL_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FL_Widget *wgt) const`
Checks if this widget is a child of `wgt`.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`

- Gets the current label text.*

 - void [label](#) (const char *text)
- Sets the current label pointer.*

 - void [label](#) (FI_Labeltype a, const char *b)
- Shortcut to set the label text and type in one call.*

 - int [label_image_spacing](#) ()
- Return the gap size between the label and the image.*

 - void [label_image_spacing](#) (int gap)
- Set the gap between the label and the image in pixels.*

 - [FI_Color](#) [labelcolor](#) () const
- Gets the label color.*

 - void [labelcolor](#) ([FI_Color](#) c)
- Sets the label color.*

 - [FI_Font](#) [labelfont](#) () const
- Gets the font to use.*

 - void [labelfont](#) ([FI_Font](#) f)
- Sets the font to use.*

 - [FI_Fonsize](#) [labelsize](#) () const
- Gets the font size in pixels.*

 - void [labelsize](#) ([FI_Fonsize](#) pix)
- Sets the font size in pixels.*

 - [FI_Labeltype](#) [labeltype](#) () const
- Gets the label type.*

 - void [labeltype](#) ([FI_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const
- Sets width ww and height hh accordingly with the label size.*

 - bool [needs_keyboard](#) () const
- Returns whether this widget needs a keyboard.*

 - void [needs_keyboard](#) (bool needs)
- Sets whether this widget needs a keyboard.*

 - unsigned int [output](#) () const
- Returns if a widget is used for output only.*

 - [FI_Group](#) * [parent](#) () const
- Returns a pointer to the parent widget.*

 - void [parent](#) ([FI_Group](#) *p)
- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
- Repositions the window or widget.*

 - void [redraw](#) ()
- Schedules the drawing of the widget.*

 - void [redraw_label](#) ()
- Schedules the drawing of the label.*

 - virtual void [resize](#) (int x, int y, int w, int h)
- Changes the size or position of the widget.*

 - [FI_Color](#) [selection_color](#) () const
- Gets the selection color.*

 - void [selection_color](#) ([FI_Color](#) a)
- Sets the selection color.*

 - void [set_active](#) ()
- Marks the widget as active without sending events or changing focus.*

- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)

- Modifies keyboard focus navigation.*

 - int **visible_r** () const
Returns whether a widget and all its parents are visible.
 - int **w** () const
Gets the widget width.
 - **Fl_When when** () const
Returns the conditions under which the callback is called.
 - void **when** (uchar i)
Sets the flags used to decide when a callback is called.
 - **Fl_Window * window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
 - int **x** () const
Gets the widget position in its window.
 - int **y** () const
Gets the widget position in its window.
 - virtual **~Fl_Widget** ()
Destroys the widget.

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
Draws the widget.

Protected Member Functions inherited from **Fl_Valuator**

- **Fl_Valuator** (int X, int Y, int W, int H, const char *L)
*Creates a new **Fl_Valuator** widget using the given position, size, and label string.*
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void **value_damage** ()
Asks for partial redraw.

Protected Member Functions inherited from **Fl_Widget**

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.

- void **draw_box** (Fl_Boxtype t, Fl_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **Fl_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from Fl_Widget

- static void **default_callback** (Fl_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from Fl_Widget

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,

```
MAXIMIZED = 1<<24 , POPUP = 1<<25 , USERFLAG3 = 1<<29 , USERFLAG2 = 1<<30 ,
USERFLAG1 = 1<<31 }
```

flags possible values enumeration.

11.118.1 Detailed Description

The [Fl_Roller](#) widget is a "dolly" control commonly used to move 3D objects.

The roller can be controlled by clicking and dragging the mouse, by the corresponding arrow keys when the roller has the keyboard focus, or by the mouse wheels when the mouse pointer is positioned over the roller widget.

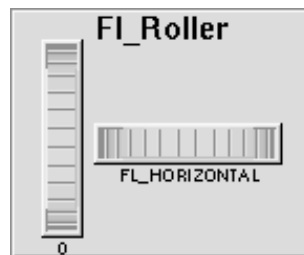


Figure 11.38 Fl_Roller

11.118.2 Constructor & Destructor Documentation

11.118.2.1 Fl_Roller()

```
Fl_Roller::Fl_Roller (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Roller](#) widget using the given position, size, and label string.

The default boxtype is FL_NO_BOX.

Inherited destructor destroys the valuator.

11.118.3 Member Function Documentation

11.118.3.1 draw()

```
void Fl_Roller::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.118.3.2 handle()

```
int Fl_Roller::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

<code>in</code>	<code>event</code>	the kind of event received

Return values

<code>0</code>	if the event was not used or understood
<code>1</code>	if the event was used and can be deleted

See also

[FL_Event](#)

Reimplemented from [FL_Widget](#).

The documentation for this class was generated from the following files:

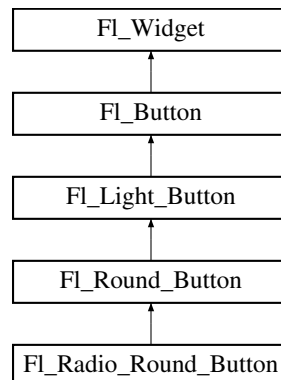
- [FL_Roller.H](#)
- [FL_Roller.cxx](#)

11.119 FL_Round_Button Class Reference

Buttons generate callbacks when they are clicked by the user.

```
#include <FL_Round_Button.H>
```

Inheritance diagram for FL_Round_Button:



Public Member Functions

- [FL_Round_Button](#) (int `x`, int `y`, int `w`, int `h`, const char *l=0)
Creates a new [FL_Round_Button](#) widget using the given position, size, and label string.

Public Member Functions inherited from [FL_Light_Button](#)

- [FL_Light_Button](#) (int `x`, int `y`, int `w`, int `h`, const char *l=0)
Creates a new [FL_Light_Button](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [Fl_Button](#)

- `int clear ()`
Same as `value(0)`.
- `uchar compact ()`
Return true if buttons are rendered as compact buttons.
- `void compact (uchar v)`
Decide if buttons should be rendered in compact mode.
- `Fl_Boxtype down_box () const`
Returns the current down box type, which is drawn when `value()` is non-zero.
- `void down_box (Fl_Boxtype b)`
Sets the down box type.
- `Fl_Color down_color () const`
(for backwards compatibility)
- `void down_color (unsigned c)`
(for backwards compatibility)
- `Fl_Button (int X, int Y, int W, int H, const char *L=0)`
The constructor creates the button using the given position, size, and label.
- `int set ()`
Same as `value(1)`.
- `void setonly ()`
Turns on this button and turns off all other radio buttons in the group (calling `value(1)` or `set()` does not do this).
- `int shortcut () const`
Returns the current shortcut key for the button.
- `void shortcut (const char *s)`
(for backwards compatibility)
- `void shortcut (int s)`
Sets the shortcut key to `s`.
- `char value () const`
Returns the current value of the button (0 or 1).
- `int value (int v)`
Sets the current value of the button.

Public Member Functions inherited from [Fl_Widget](#)

- `void _clear_fullscreen ()`
- `void _set_fullscreen ()`
- `void activate ()`
Activates the widget.
- `unsigned int active () const`
Returns whether the widget is active.
- `int active_r () const`
Returns whether the widget and all of its parents are active.
- `Fl_Align align () const`
Gets the label alignment.
- `void align (Fl_Align alignment)`
Sets the label alignment.
- `long argument () const`
Gets the current user data (long) argument that is passed to the callback function.
- `void argument (long v)`
Sets the current user data (long) argument that is passed to the callback function.
- virtual class `Fl_Gl_Window * as_gl_window ()`

Returns an *FI_GL_Window* pointer if this widget is an *FI_GL_Window*.

- virtual class *FI_GL_Window* const * **as_gl_window** () const
- virtual *FI_Group* * **as_group** ()

Returns an *FI_Group* pointer if this widget is an *FI_Group*.

- virtual *FI_Group* const * **as_group** () const
- virtual *FI_Window* * **as_window** ()

Returns an *FI_Window* pointer if this widget is an *FI_Window*.

- virtual *FI_Window* const * **as_window** () const
- void **bind_deimage** (*FI_Image* *img)

Sets the image to use as part of the widget label when in the inactive state.

- void **bind_deimage** (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.

- void **bind_image** (*FI_Image* *img)

Sets the image to use as part of the widget label when in the active state.

- void **bind_image** (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.

- *FI_Boxtype* **box** () const

Gets the box type of the widget.

- void **box** (*FI_Boxtype* new_box)

Sets the box type for the widget.

- *FI_Callback_p* **callback** () const

Gets the current callback function for the widget.

- void **callback** (*FI_Callback* *cb)

Sets the current callback function for the widget.

- void **callback** (*FI_Callback* *cb, *FI_Callback_User_Data* *p, bool auto_free)

Sets the current callback function and managed user data for the widget.

- void **callback** (*FI_Callback* *cb, void *p)

Sets the current callback function and data for the widget.

- void **callback** (*FI_Callback0* *cb)

Sets the current callback function for the widget.

- void **callback** (*FI_Callback1* *cb, long p=0)

Sets the current callback function for the widget.

- unsigned int **changed** () const

Checks if the widget value changed since the last callback.

- void **clear_active** ()

Marks the widget as inactive without sending events or changing focus.

- void **clear_changed** ()

Marks the value of the widget as unchanged.

- void **clear_damage** (uchar c=0)

Clears or sets the damage flags.

- void **clear_output** ()

Sets a widget to accept input.

- void **clear_visible** ()

Hides the widget.

- void **clear_visible_focus** ()

Disables keyboard focus navigation with this widget.

- *FI_Color* **color** () const

Gets the background color of the widget.

- void **color** (*FI_Color* bg)

Sets the background color of the widget.

- void **color** (*FI_Color* bg, *FI_Color* sel)

- Sets the background and selection color of the widget.*

 - `FL_Color color2 () const`

For back compatibility only.
- `void color2 (unsigned a)`

For back compatibility only.
- `int contains (const FL_Widget *w) const`

Checks if w is a child of this widget.
- `void copy_label (const char *new_label)`

Sets the current label.
- `void copy_tooltip (const char *text)`

Sets the current tooltip text.
- `uchar damage () const`

Returns non-zero if `draw()` needs to be called.
- `void damage (uchar c)`

Sets the damage bits for the widget.
- `void damage (uchar c, int x, int y, int w, int h)`

Sets the damage bits for an area inside the widget.
- `int damage_resize (int, int, int, int)`

Internal use only.
- `void deactivate ()`

Deactivates the widget.
- `FL_Image * deimage ()`

Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`

Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`

Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`

Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`

Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FL_Align) const`

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`

Gets the widget height.
- `virtual void hide ()`

Makes a widget invisible.
- `int horizontal_label_margin ()`

Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`

Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`

Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`

Gets the image that is used as part of the widget label when in the active state.

- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()

- Schedules the drawing of the widget.*
 - void `redraw_label` ()
- Schedules the drawing of the label.*
 - virtual void `resize` (int `x`, int `y`, int `w`, int `h`)
- Changes the size or position of the widget.*
 - `FI_Color` `selection_color` () const
- Gets the selection color.*
 - void `selection_color` (`FI_Color` `a`)
- Sets the selection color.*
 - void `set_active` ()
- Marks the widget as active without sending events or changing focus.*
 - void `set_changed` ()
- Marks the value of the widget as changed.*
 - void `set_output` ()
- Sets a widget to output only.*
 - void `set_visible` ()
- Makes the widget visible.*
 - void `set_visible_focus` ()
- Enables keyboard focus navigation with this widget.*
 - int `shortcut_label` () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*
 - void `shortcut_label` (int `value`)
- Sets whether the widget's label uses '&' to indicate shortcuts.*
 - virtual void `show` ()
- Makes a widget visible.*
 - void `size` (int `W`, int `H`)
- Changes the size of the widget.*
 - int `take_focus` ()
- Gives the widget the keyboard focus.*
 - unsigned int `takeevents` () const
- Returns if the widget is able to take events.*
 - int `test_shortcut` ()
- Returns true if the widget's label contains the entered '&x' shortcut.*
 - const char * `tooltip` () const
- Gets the current tooltip text.*
 - void `tooltip` (const char *`text`)
- Sets the current tooltip text.*
 - `FI_Window` * `top_window` () const
- Returns a pointer to the top-level window for the widget.*
 - `FI_Window` * `top_window_offset` (int &`xoff`, int &`yoff`) const
- Finds the x/y offset of the current widget relative to the top-level window.*
 - `uchar` `type` () const
- Gets the widget type.*
 - void `type` (`uchar` `t`)
- Sets the widget type.*
 - int `use_accents_menu` ()
- Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*
 - void * `user_data` () const
- Gets the user data for this widget.*
 - void `user_data` (`FI_Callback_User_Data` *`v`, bool `auto_free`)
- Sets the user data for this widget.*

- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **Fl_When** when () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~Fl_Widget** ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from **Fl_Widget**

- static void **default_callback** (**Fl_Widget** *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from **Fl_Widget**

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,

`MAXIMIZED = 1<<24 , POPUP = 1<<25 , USERFLAG3 = 1<<29 , USERFLAG2 = 1<<30 ,
USERFLAG1 = 1<<31 }`

flags possible values enumeration.

Protected Member Functions inherited from `Fl_Light_Button`

- void `draw ()` `FL_OVERRIDE`

Draws the widget.

Protected Member Functions inherited from `Fl_Button`

- void `simulate_key_action ()`

Protected Member Functions inherited from `Fl_Widget`

- void `clear_flag` (unsigned int c)
Clears a flag in the flags mask.
- void `draw_backdrop ()` const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void `draw_box ()` const
Draws the widget box according its box style.
- void `draw_box` (Fl_Boxtype t, Fl_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void `draw_box` (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void `draw_focus ()` const
Draws a focus rectangle around the widget.
- void `draw_focus` (Fl_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void `draw_focus` (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label ()` const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `Fl_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags ()` const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Static Protected Member Functions inherited from `Fl_Button`

- static void `key_release_timeout` (void *)

Static Protected Attributes inherited from [Fl_Button](#)

- static [Fl_Widget_Tracker](#) * `key_release_tracker` = 0

11.119.1 Detailed Description

Buttons generate callbacks when they are clicked by the user.

You control exactly when and how by changing the values for [type\(\)](#) and [when\(\)](#).



Figure 11.39 [Fl_Round_Button](#)

The [Fl_Round_Button](#) subclass displays the "on" state by turning on a light, rather than drawing pushed in. The shape of the "light" is initially set to `FL_ROUND_DOWN_BOX`. The color of the light when on is controlled with [selection_color\(\)](#), which defaults to `FL_FOREGROUND_COLOR`.

11.119.2 Constructor & Destructor Documentation

11.119.2.1 [Fl_Round_Button\(\)](#)

```
Fl_Round_Button::Fl_Round_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Round_Button](#) widget using the given position, size, and label string.

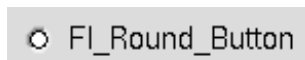


Figure 11.40 [Fl_Round_Button](#)

The [Fl_Round_Button](#) subclass displays the "ON" state by turning on a light, rather than drawing pushed in.

The default box type is `FL_NO_BOX`, which draws the label w/o a box right of the checkmark.

The shape of the "light" is set with [down_box\(\)](#) and its default value is `FL_ROUND_DOWN_BOX`.

The color of the light when on is controlled with [selection_color\(\)](#), which defaults to `FL_FOREGROUND_COLOR` (usually black).

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

The documentation for this class was generated from the following files:

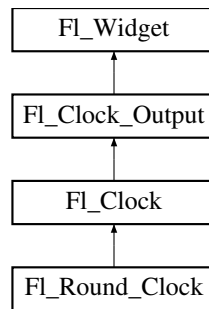
- [Fl_Round_Button.H](#)
- [Fl_Round_Button.cxx](#)

11.120 [Fl_Round_Clock](#) Class Reference

A clock widget of type `FL_ROUND_CLOCK`.

```
#include <Fl_Round_Clock.H>
```

Inheritance diagram for [Fl_Round_Clock](#):



Public Member Functions

- [Fl_Round_Clock](#) (int X, int Y, int W, int H, const char *L=0)
Creates the clock widget, setting his type and box.

Public Member Functions inherited from [Fl_Clock](#)

- [Fl_Clock](#) (int X, int Y, int W, int H, const char *L=0)
Create an [Fl_Clock](#) widget using the given position, size, and label string.
- [Fl_Clock](#) (uchar t, int X, int Y, int W, int H, const char *L)
Create an [Fl_Clock](#) widget using the given clock type t, position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- [~Fl_Clock](#) ()
The destructor removes the clock.

Public Member Functions inherited from [Fl_Clock_Output](#)

- [Fl_Clock_Output](#) (int X, int Y, int W, int H, const char *L=0)
Create a new [Fl_Clock_Output](#) widget with the given position, size and label.
- int [hour](#) () const
Returns the displayed hour (0 to 23).
- int [minute](#) () const
Returns the displayed minute (0 to 59).
- int [second](#) () const
Returns the displayed second (0 to 60, 60=leap second).
- int [shadow](#) () const
Returns the shadow drawing mode of the hands.
- void [shadow](#) (int mode)
Sets the shadow drawing mode of the hands.
- [ulong value](#) () const
Returns the displayed time.
- void [value](#) (int H, int m, int s)
Set the displayed time.
- void [value](#) (ulong v)
Set the displayed time.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (**FI_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FI_Gl_Window** * **as_gl_window** ()
Returns an FI_Gl_Window pointer if this widget is an FI_Gl_Window.
- virtual class **FI_Gl_Window** const * **as_gl_window** () const
- virtual **FI_Group** * **as_group** ()
Returns an FI_Group pointer if this widget is an FI_Group.
- virtual **FI_Group** const * **as_group** () const
- virtual **FI_Window** * **as_window** ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual **FI_Window** const * **as_window** () const
- void **bind_deimage** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- **FI_Boxtype box** () const
Gets the box type of the widget.
- void **box** (**FI_Boxtype** new_box)
Sets the box type for the widget.
- **FI_Callback_p callback** () const
Gets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb, **FI_Callback_User_Data** *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void **callback** (**FI_Callback** *cb, void *p)
Sets the current callback function and data for the widget.
- void **callback** (**FI_Callback0** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback1** *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int **changed** () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FL_Color` bg)

Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)

Sets the background and selection color of the widget.
- `FL_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.

- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const
- Sets width ww and height hh accordingly with the label size.*

 - bool [needs_keyboard](#) () const
- Returns whether this widget needs a keyboard.*

 - void [needs_keyboard](#) (bool needs)
- Sets whether this widget needs a keyboard.*

 - unsigned int [output](#) () const
- Returns if a widget is used for output only.*

 - [Fl_Group](#) * [parent](#) () const
- Returns a pointer to the parent widget.*

 - void [parent](#) ([Fl_Group](#) *p)
- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
- Repositions the window or widget.*

 - void [redraw](#) ()
- Schedules the drawing of the widget.*

 - void [redraw_label](#) ()
- Schedules the drawing of the label.*

 - virtual void [resize](#) (int x, int y, int w, int h)
- Changes the size or position of the widget.*

 - [Fl_Color](#) [selection_color](#) () const
- Gets the selection color.*

 - void [selection_color](#) ([Fl_Color](#) a)
- Sets the selection color.*

 - void [set_active](#) ()
- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

 - void [set_output](#) ()
- Sets a widget to output only.*

 - void [set_visible](#) ()
- Makes the widget visible.*

 - void [set_visible_focus](#) ()
- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void [shortcut_label](#) (int value)
- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - virtual void [show](#) ()
- Makes a widget visible.*

 - void [size](#) (int W, int H)
- Changes the size of the widget.*

 - int [take_focus](#) ()
- Gives the widget the keyboard focus.*

 - unsigned int [takeevents](#) () const
- Returns if the widget is able to take events.*

 - int [test_shortcut](#) ()
- Returns true if the widget's label contains the entered '&x' shortcut.*

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Widget](#)

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from [FI_Clock_Output](#)

- void [draw](#) () [FL_OVERRIDE](#)
Draw clock with current position and size.
- void [draw](#) (int X, int Y, int W, int H)
Draw clock with the given position and size.

Protected Member Functions inherited from [FI_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const

- Draws the widget's label at the defined label position.*
- void [draw_label](#) (int, int, int, int) const
- Draws the label in an arbitrary bounding box.*
- [Fl_Widget](#) (int [x](#), int [y](#), int [w](#), int [h](#), const char *[label](#)=0L)
- Creates a widget at the given position and size.*
- unsigned int **flags** () const
- Gets the widget flags mask.*
- void [h](#) (int v)
- Internal use only.*
- void **set_flag** (unsigned int c)
- Sets a flag in the flags mask.*
- void [w](#) (int v)
- Internal use only.*
- void [x](#) (int v)
- Internal use only.*
- void [y](#) (int v)
- Internal use only.*

11.120.1 Detailed Description

A clock widget of type FL_ROUND_CLOCK.
Has no box.

11.120.2 Constructor & Destructor Documentation

11.120.2.1 Fl_Round_Clock()

```
Fl_Round_Clock::Fl_Round_Clock (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates the clock widget, setting his type and box.

Create an [Fl_Round_Clock](#) widget using the given position, size, and label string.

The clock type is FL_ROUND_CLOCK and the boxtype is FL_NO_BOX.

This constructor is the same as [Fl_Clock](#)(FL_ROUND_CLOCK, X, Y, W, H, L).

See also

[Fl_Clock](#)(uchar, int, int, int, int, const char *)

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

The documentation for this class was generated from the following files:

- [Fl_Round_Clock.H](#)
- [Fl_Clock.cxx](#)

11.121 FL_Scheme Class Reference

Static Public Member Functions

- static int [add_scheme_name](#) (const char *name)
Add a scheme name to the list of known schemes.
- static const char ** [names](#) ()
Return a list of all known scheme names.
- static int [num_schemes](#) ()
Return the number of currently registered schemes.

11.121.1 Member Function Documentation

11.121.1.1 add_scheme_name()

```
int FL_Scheme::add_scheme_name (
    const char * name) [static]
```

Add a scheme name to the list of known schemes.

This method is public in FLTK 1.4.0 because derived classes of [FL_Scheme](#) are not yet implemented. Thus, users implementing their own schemes can use this method to add the scheme name to the list of known schemes which is for instance used in [FL_Scheme::names\(\)](#).

Note

Attention! In a future version, when subclasses of [FL_Scheme](#) will be implemented, this method will either be replaced by another `protected` method or it will no longer do anything (kept only for ABI reasons).

The new scheme name must consist of valid ASCII characters as described below:

- lowercase letters 'a' - 'z'
- numbers '0' - '9'
- any character in "\$+_. " (w/o the quotes).

The name must not be longer than 12 ASCII characters (bytes). The new scheme name is added to the **end** of the **unordered** list.

Note

Call this method only once for each scheme name. If the returned value is ≤ 0 you should check the scheme name.

The given scheme `name` is copied and may be freed directly after the call to [add_scheme_name\(\)](#).

Parameters

in	<i>name</i>	New scheme name

Returns

The new number of schemes if the name was successfully added. This is the same as the index of the scheme + 1.

Return values

0	Scheme name already exists
-1	Invalid character(s) in <i>name</i>
-2	The <i>name</i> is too long

Since

1.4.0

11.121.1.2 names()

```
const char ** Fl_Scheme::names () [static]
```

Return a list of all known scheme names.

This list is only valid until a new scheme is added or one is removed. It is possible that scheme names are appended to the list during the runtime of the program but schemes can't be removed.

Getting the list of known schemes can be useful to populate a menu of scheme choices to let the user select a scheme. You should process the names immediately and you should never store a pointer to the list or any individual name for later reference because the location of the list can be changed (reallocated) when schemes are added.

The list of scheme names is nul-terminated.

Note

Currently (in FLTK 1.4.0) schemes can only be added to the list and not removed from the list. This may change in a later version.

Returns

List of currently known scheme names.

11.121.1.3 num_schemes()

```
static int Fl_Scheme::num_schemes () [inline], [static]
```

Return the number of currently registered schemes.

Returns

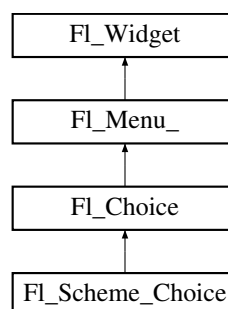
Number of registered schemes.

The documentation for this class was generated from the following files:

- Fl_Scheme.H
- Fl_Scheme.cxx

11.122 Fl_Scheme_Choice Class Reference

Inheritance diagram for Fl_Scheme_Choice:



Public Member Functions

- [Fl_Scheme_Choice](#) (int X, int Y, int W, int H, const char *L=0)
The constructor initializes the [Fl_Scheme_Choice](#) object with all known schemes.
- int [handle](#) (int event) [FL_OVERRIDE](#)
Handle FLTK events.
- virtual void [init_value](#) ()
Public method to initialize the value of the [Fl_Scheme_Choice](#) widget.

Public Member Functions inherited from [FI_Choice](#)

- [FI_Choice](#) (int X, int Y, int W, int H, const char *L=0)
Create a new [FI_Choice](#) widget using the given position, size and label string.
- int [value](#) () const
Gets the index of the last item chosen by the user.
- int [value](#) (const [FI_Menu_Item](#) *v)
Sets the currently selected value using a pointer to menu item.
- int [value](#) (int v)
Sets the currently selected value using the index into the menu item array.

Public Member Functions inherited from [FI_Menu_](#)

- int [add](#) (const char *)
This is a Forms (and SGI GL library) compatible add function, it adds many menu items, with '|' separating the menu items, and tab separating the menu item names from an optional shortcut string.
- int [add](#) (const char *, int [shortcut](#), [FI_Callback](#) *, void *=0, int=0)
Adds a new menu item.
- int [add](#) (const char *a, const char *b, [FI_Callback](#) *c, void *d=0, int e=0)
See int [FI_Menu_::add](#)(const char label, int shortcut, [FI_Callback](#)*, void *user_data=0, int flags=0)*
- void [clear](#) ()
Same as menu(NULL), set the array pointer to null, indicating a zero-length menu.
- int [clear_submenu](#) (int index)
*Clears the specified submenu pointed to by *index* of all menu items.*
- void [copy](#) (const [FI_Menu_Item](#) *m, void *user_data=0)
Sets the menu array pointer with a copy of m that will be automatically deleted.
- [FI_Boxtype](#) [down_box](#) () const
This box type is used to surround the currently-selected items in the menus.
- void [down_box](#) ([FI_Boxtype](#) b)
Sets the box type used to surround the currently-selected items in the menus.
- [FI_Color](#) [down_color](#) () const
For back compatibility, same as [selection_color\(\)](#)
- void [down_color](#) (unsigned c)
For back compatibility, same as [selection_color\(\)](#)
- int [find_index](#) (const char *name) const
*Find the menu item index for a given menu *pathname*, such as "Edit/Copy".*
- int [find_index](#) (const [FI_Menu_Item](#) *item) const
*Find the index into the menu array for a given *item*.*
- int [find_index](#) ([FI_Callback](#) *cb) const
*Find the index into the menu array for a given callback *cb*.*
- const [FI_Menu_Item](#) * [find_item](#) (const char *name)
*Find the menu item for a given menu *pathname*, such as "Edit/Copy".*
- const [FI_Menu_Item](#) * [find_item](#) ([FI_Callback](#) *)
*Find the menu item for the given callback *cb*.*
- const [FI_Menu_Item](#) * [find_item_with_argument](#) (long)
*Find the menu item for the given user argument *v*.*
- const [FI_Menu_Item](#) * [find_item_with_user_data](#) (void *)
*Find the menu item for the given user data *v*.*
- [FI_Menu_](#) (int, int, int, int, const char *=0)
Creates a new [FI_Menu_](#) widget using the given position, size, and label string.
- void [global](#) ()
Make the shortcuts for this menu work no matter what window has the focus when you type it.

- int **insert** (int index, const char *, int **shortcut**, **Fl_Callback** *, void *=0, int=0)
Inserts a new menu item at the specified `index` position.
- int **insert** (int index, const char *a, const char *b, **Fl_Callback** *c, void *d=0, int e=0)
See int `Fl_Menu_::insert(const char* label, int shortcut, Fl_Callback*, void *user_data=0, int flags=0)`
- int **item_pathname** (char *name, int namelen, const **Fl_Menu_Item** *finditem=0) const
Get the menu 'pathname' for the specified menuitem.
- const **Fl_Menu_Item** * **menu** () const
Returns a pointer to the array of `Fl_Menu_Items`.
- void **menu** (const **Fl_Menu_Item** *m)
Sets the menu array pointer directly.
- **Fl_Boxtype** **menu_box** () const
Get the box type for the menu popup windows.
- void **menu_box** (**Fl_Boxtype** b)
Set the box type for the menu popup windows.
- const **Fl_Menu_Item** * **menu_end** ()
Finishes menu modifications and returns `menu()`.
- int **mode** (int i) const
Get the flags of item `i`.
- void **mode** (int i, int fl)
Set the flags of item `i`.
- const **Fl_Menu_Item** * **mvalue** () const
Return a pointer to the last menu item that was picked.
- const **Fl_Menu_Item** * **picked** (const **Fl_Menu_Item** *)
When user picks a menu item, call this.
- const **Fl_Menu_Item** * **prev_mvalue** () const
Return a pointer to the menu item that was picked before the current one was picked.
- void **remove** (int)
Deletes item `i` from the menu.
- void **replace** (int, const char *)
Changes the text of item `i`.
- void **setonly** (**Fl_Menu_Item** *item)
Turns the radio item "on" for the menu item and turns "off" adjacent radio items of the same group.
- void **shortcut** (int i, int s)
Change the shortcut of item `i` to `s`.
- int **size** () const
This returns the number of `Fl_Menu_Item` structures that make up the menu, correctly counting submenus.
- void **size** (int W, int H)
- const **Fl_Menu_Item** * **test_shortcut** ()
Returns the menu item with the entered shortcut (key value).
- const char * **text** () const
Returns the title of the last item chosen.
- const char * **text** (int i) const
Returns the title of item `i`.
- **Fl_Color** **textcolor** () const
Get the current color of menu item labels.
- void **textcolor** (**Fl_Color** c)
Sets the current color of menu item labels.
- **Fl_Font** **textfont** () const
Gets the current font of menu item labels.
- void **textfont** (**Fl_Font** c)
Sets the current font of menu item labels.

- [FI_Fontsize](#) [textsize](#) () const
Gets the font size of menu item labels.
- void [textsize](#) ([FI_Fontsize](#) c)
Sets the font size of menu item labels.
- int [value](#) () const
Return the index into the [menu\(\)](#) of the last item chosen by the user.
- int [value](#) (const [FI_Menu_Item](#) *)
Set the value of a menu to the menu item m.
- int [value](#) (int i)
Set the value of the menu to index i.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const

- Gets the current callback function for the widget.*

 - void [callback](#) ([Fl_Callback](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*

 - void [callback](#) ([Fl_Callback](#) *cb, void *p)
- Sets the current callback function and data for the widget.*

 - void [callback](#) ([Fl_Callback0](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
- Sets the current callback function for the widget.*

 - unsigned int [changed](#) () const
- Checks if the widget value changed since the last callback.*

 - void [clear_active](#) ()
- Marks the widget as inactive without sending events or changing focus.*

 - void [clear_changed](#) ()
- Marks the value of the widget as unchanged.*

 - void [clear_damage](#) ([uchar](#) c=0)
- Clears or sets the damage flags.*

 - void [clear_output](#) ()
- Sets a widget to accept input.*

 - void [clear_visible](#) ()
- Hides the widget.*

 - void [clear_visible_focus](#) ()
- Disables keyboard focus navigation with this widget.*

 - [Fl_Color](#) [color](#) () const
- Gets the background color of the widget.*

 - void [color](#) ([Fl_Color](#) bg)
- Sets the background color of the widget.*

 - void [color](#) ([Fl_Color](#) bg, [Fl_Color](#) sel)
- Sets the background and selection color of the widget.*

 - [Fl_Color](#) [color2](#) () const
- For back compatibility only.*

 - void [color2](#) (unsigned a)
- For back compatibility only.*

 - int [contains](#) (const [Fl_Widget](#) *w) const
- Checks if w is a child of this widget.*

 - void [copy_label](#) (const char *new_label)
- Sets the current label.*

 - void [copy_tooltip](#) (const char *text)
- Sets the current tooltip text.*

 - [uchar](#) [damage](#) () const
- Returns non-zero if [draw\(\)](#) needs to be called.*

 - void [damage](#) ([uchar](#) c)
- Sets the damage bits for the widget.*

 - void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int [damage_resize](#) (int, int, int, int)
- Internal use only.*

 - void [deactivate](#) ()
- Deactivates the widget.*

- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const

- Gets the label color.*

 - void [labelcolor](#) ([Fl_Color](#) c)
- Sets the label color.*

 - [Fl_Font](#) [labelfont](#) () const
- Gets the font to use.*

 - void [labelfont](#) ([Fl_Font](#) f)
- Sets the font to use.*

 - [Fl_Fontsize](#) [labelsize](#) () const
- Gets the font size in pixels.*

 - void [labelsize](#) ([Fl_Fontsize](#) pix)
- Sets the font size in pixels.*

 - [Fl_Labeltype](#) [labeltype](#) () const
- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const
- Sets width ww and height hh accordingly with the label size.*

 - bool [needs_keyboard](#) () const
- Returns whether this widget needs a keyboard.*

 - void [needs_keyboard](#) (bool needs)
- Sets whether this widget needs a keyboard.*

 - unsigned int [output](#) () const
- Returns if a widget is used for output only.*

 - [Fl_Group](#) * [parent](#) () const
- Returns a pointer to the parent widget.*

 - void [parent](#) ([Fl_Group](#) *p)
- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
- Repositions the window or widget.*

 - void [redraw](#) ()
- Schedules the drawing of the widget.*

 - void [redraw_label](#) ()
- Schedules the drawing of the label.*

 - virtual void [resize](#) (int x, int y, int w, int h)
- Changes the size or position of the widget.*

 - [Fl_Color](#) [selection_color](#) () const
- Gets the selection color.*

 - void [selection_color](#) ([Fl_Color](#) a)
- Sets the selection color.*

 - void [set_active](#) ()
- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

 - void [set_output](#) ()
- Sets a widget to output only.*

 - void [set_visible](#) ()
- Makes the widget visible.*

 - void [set_visible_focus](#) ()
- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const

- Returns a pointer to the nearest parent window up the widget hierarchy.*
- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual `~FI_Widget` ()
Destroys the widget.

Static Protected Member Functions

- static void `scheme_cb_` (`FI_Widget *w`, void *)
Internal `FI_Scheme_Choice` callback function (protected).

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget *widget`, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from `FI_Choice`

- void `draw` () `FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from `FI_Menu_`

- int `item_pathname_` (char *name, int namelen, const `FI_Menu_Item` *finditem, const `FI_Menu_Item` *menu=0) const

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Protected Attributes inherited from [FI_Menu_](#)

- [uchar](#) **alloc**
- [uchar](#) **down_box_**
- [FI_Boxtype](#) **menu_box_**
- [FI_Color](#) **textcolor_**
- [FI_Font](#) **textfont_**
- [FI_Fonsize](#) **textsize_**

11.122.1 Constructor & Destructor Documentation

11.122.1.1 Fl_Scheme_Choice()

```
Fl_Scheme_Choice::Fl_Scheme_Choice (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

The constructor initializes the [Fl_Scheme_Choice](#) object with all known schemes.

Parameters

in	<i>X,Y</i>	Widget coordinates
in	<i>W,H</i>	Widget size (width, height)
in	<i>L</i>	Widget label (default: NULL, no label)

11.122.2 Member Function Documentation

11.122.2.1 handle()

```
int Fl_Scheme_Choice::handle (
    int event) [virtual]
```

Handle FLTK events.

This widget uses FL_SHOW and some other events to initialize its [value\(\)](#) according to the current scheme.

All events are also handled by the base class [Fl_Choice](#).

Parameters

in	<i>event</i>	

Returns

1 if the event was used, 0 otherwise

Reimplemented from [Fl_Choice](#).

11.122.2.2 init_value()

```
void Fl_Scheme_Choice::init_value () [virtual]
```

Public method to initialize the value of the [Fl_Scheme_Choice](#) widget.

Normally you don't need to call this unless you change the current scheme by calling [Fl::scheme\(const char *\)](#).

The [Fl_Scheme_Choice](#) widget does this automatically when the widget is shown (when receiving the FL_SHOW event) which should always be after [Fl_Window::show\(argc, argv\)](#) which may set the current scheme by interpreting the commandline.

Since

1.4.0

11.122.2.3 scheme_cb_()

```
void Fl_Scheme_Choice::scheme_cb_ (
    Fl_Widget * w,
    void * ) [static], [protected]
```

Internal [Fl_Scheme_Choice](#) callback function (protected).

You don't need to set a callback for this widget. The default callback changes the scheme ([Fl::scheme\(\)](#)) and redraws all open windows.

You may override the callback if changing the scheme shall redraw other windows or don't redraw the window at all.

Parameters

in	w	The Fl_Scheme_Choice widget

The documentation for this class was generated from the following files:

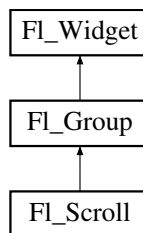
- [Fl_Scheme_Choice.H](#)
- [Fl_Scheme_Choice.cxx](#)

11.123 Fl_Scroll Class Reference

This container widget lets you maneuver around a set of widgets much larger than your window.

```
#include <Fl_Scroll.H>
```

Inheritance diagram for [Fl_Scroll](#):



Classes

- struct [Fl_Region_LRTB](#)
A local struct to manage a region defined by left/right/top/bottom.
- struct [Fl_Region_XYWH](#)
A local struct to manage a region defined by xywh.
- struct [Fl_Scrollbar_Data](#)
A local struct to manage a scrollbar's xywh region and tab values.
- struct [ScrollInfo](#)
Structure to manage scrollbar and widget interior sizes.

Public Types

- enum {
HORIZONTAL = 1 , **VERTICAL** = 2 , **BOTH** = 3 , **ALWAYS_ON** = 4 ,
HORIZONTAL_ALWAYS = 5 , **VERTICAL_ALWAYS** = 6 , **BOTH_ALWAYS** = 7 }

Public Member Functions

- void **clear** ()
Clear all but the scrollbars...
- int **delete_child** (int n) [FL_OVERRIDE](#)
Removes the widget at *index* from the group and deletes it.
- [Fl_Scroll](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Scroll](#) widget using the given position, size, and label string.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- void **resize** (int X, int Y, int W, int H) [FL_OVERRIDE](#)
Resizes the [Fl_Scroll](#) widget and moves its children if necessary.
- void **scroll_to** (int, int)
Moves the contents of the scroll group to a new position.

- int `scrollbar_size` () const
Gets the current size of the scrollbars' troughs, in pixels.
- void `scrollbar_size` (int newSize)
Sets the pixel size of the scrollbars' troughs to newSize, in pixels.
- int `xposition` () const
Gets the current horizontal scrolling position.
- int `yposition` () const
Gets the current vertical scrolling position.
- virtual `~FL_Scroll` ()
The destructor also deletes all the children.

Public Member Functions inherited from FL_Group

- `FL_Widget` *& `_ddfdesign_kludge` ()
This is for forms compatibility only.
- void `add` (`FL_Widget` &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void `add` (`FL_Widget` *o)
See void FL_Group::add(FL_Widget &w)
- void `add_resizable` (`FL_Widget` &o)
Adds a widget to the group and makes it the resizable widget.
- `FL_Widget` *const * `array` () const
Returns a pointer to the internal array of children.
- `FL_Group` const * `as_group` () const `FL_OVERRIDE`
- `FL_Group` * `as_group` () `FL_OVERRIDE`
Returns an FL_Group pointer if this widget is an FL_Group.
- void `begin` ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- `FL_Widget` * `child` (int n) const
Returns the n'th child.
- int `children` () const
Returns how many child widgets the group has.
- void `clear` ()
Deletes all child widgets from memory recursively.
- unsigned int `clip_children` ()
Returns the current clipping mode.
- void `clip_children` (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- void `end` ()
Exactly the same as current(this->parent()).
- int `find` (const `FL_Widget` &o) const
*See int FL_Group::find(const FL_Widget *w) const.*
- int `find` (const `FL_Widget` *) const
Searches the children for the widget and returns the index.
- `FL_Group` (int, int, int, int, const char *s)
Creates a new FL_Group widget using the given position, size, and label string.
- void `focus` (`FL_Widget` *W)
- void `forms_end` ()
This is for forms compatibility only.
- void `init_sizes` ()
Resets the internal array of widget sizes and positions.

- void `insert` (`FL_Widget` &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void `insert` (`FL_Widget` &o, `FL_Widget` *before)
This does `insert(w, find(before))`.
- void `remove` (`FL_Widget` &)
Removes a widget from the group but does not delete it.
- void `remove` (`FL_Widget` *o)
Removes the widget o from the group.
- void `remove` (int index)
Removes the widget at index from the group but does not delete it.
- `FL_Widget` * `resizable` () const
Returns the group's resizable widget.
- void `resizable` (`FL_Widget` &o)
Sets the group's resizable widget.
- void `resizable` (`FL_Widget` *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual `~FL_Group` ()
The destructor also deletes all the children.

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
Activates the widget.
- unsigned int `active` () const
Returns whether the widget is active.
- int `active_r` () const
Returns whether the widget and all of its parents are active.
- `FL_Align` `align` () const
Gets the label alignment.
- void `align` (`FL_Align` alignment)
Sets the label alignment.
- long `argument` () const
Gets the current user data (long) argument that is passed to the callback function.
- void `argument` (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class `FL_Gl_Window` * `as_gl_window` ()
Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.
- virtual class `FL_Gl_Window` const * `as_gl_window` () const
- virtual `FL_Window` * `as_window` ()
Returns an `FL_Window` pointer if this widget is an `FL_Window`.
- virtual `FL_Window` const * `as_window` () const
- void `bind_deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- void `bind_deimage` (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void `bind_image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- void `bind_image` (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.

- [FL_Boxtype box](#) () const
Gets the box type of the widget.
- void [box](#) ([FL_Boxtype](#) new_box)
Sets the box type for the widget.
- [FL_Callback_p callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FL_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FL_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) ([uchar](#) c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FL_Color color](#) () const
Gets the background color of the widget.
- void [color](#) ([FL_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FL_Color](#) bg, [FL_Color](#) sel)
Sets the background and selection color of the widget.
- [FL_Color color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FL_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)

- Sets the damage bits for an area inside the widget.*

 - int **damage_resize** (int, int, int, int)

Internal use only.
- void **deactivate** ()

Deactivates the widget.
- **Fl_Image** * **deimage** ()

Gets the image that is used as part of the widget label when in the inactive state.
- const **Fl_Image** * **deimage** () const

Gets the image that is used as part of the widget label when in the inactive state.
- void **deimage** (**Fl_Image** &img)

Sets the image to use as part of the widget label when in the inactive state.
- void **deimage** (**Fl_Image** *img)

Sets the image to use as part of the widget label when in the inactive state.
- int **deimage_bound** () const

Returns whether the inactive image is managed by the widget.
- void **do_callback** (**Fl_Callback_Reason** reason=**FL_REASON_UNKNOWN**)

Calls the widget callback function with default arguments.
- void **do_callback** (**Fl_Widget** *widget, long arg, **Fl_Callback_Reason** reason=**FL_REASON_UNKNOWN**)

Calls the widget callback function with arbitrary arguments.
- void **do_callback** (**Fl_Widget** *widget, void *arg=0, **Fl_Callback_Reason** reason=**FL_REASON_UNKNOWN**)

Calls the widget callback function with arbitrary arguments.
- void **draw_label** (int, int, int, int, **Fl_Align**) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int **h** () const

Gets the widget height.
- virtual void **hide** ()

Makes a widget invisible.
- int **horizontal_label_margin** ()

Get the spacing between the label and the horizontal edge of the widget.
- void **horizontal_label_margin** (int px)

Set the spacing between the label and the horizontal edge of the widget.
- **Fl_Image** * **image** ()

Gets the image that is used as part of the widget label when in the active state.
- const **Fl_Image** * **image** () const

Gets the image that is used as part of the widget label when in the active state.
- void **image** (**Fl_Image** &img)

Sets the image to use as part of the widget label when in the active state.
- void **image** (**Fl_Image** *img)

Sets the image to use as part of the widget label when in the active state.
- int **image_bound** () const

Returns whether the image is managed by the widget.
- int **inside** (const **Fl_Widget** *wgt) const

Checks if this widget is a child of wgt.
- int **is_label_copied** () const

Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * **label** () const

Gets the current label text.
- void **label** (const char *text)

Sets the current label pointer.
- void **label** (**Fl_Labeltype** a, const char *b)

Shortcut to set the label text and type in one call.

- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [Fl_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([Fl_Color](#) c)
Sets the label color.
- [Fl_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([Fl_Font](#) f)
Sets the font to use.
- [Fl_Fontsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([Fl_Fontsize](#) pix)
Sets the font size in pixels.
- [Fl_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([Fl_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [Fl_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([Fl_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [Fl_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()

- Enables keyboard focus navigation with this widget.*

 - int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()

Makes a widget visible.
- void `size` (int W, int H)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const

Gets the current tooltip text.
- void `tooltip` (const char *text)

Sets the current tooltip text.
- `Fl_Window` * `top_window` () const

Returns a pointer to the top-level window for the widget.
- `Fl_Window` * `top_window_offset` (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- `uchar` `type` () const

Gets the widget type.
- void `type` (`uchar` t)

Sets the widget type.
- int `use_accents_menu` ()

Returns non zero if `MAC_USE_ACCENTS_MENU` flag is set, 0 otherwise.
- void * `user_data` () const

Gets the user data for this widget.
- void `user_data` (`Fl_Callback_User_Data` *v, bool auto_free)

Sets the user data for this widget.
- void `user_data` (void *v)

Sets the user data for this widget.
- int `vertical_label_margin` ()

Get the spacing between the label and the vertical edge of the widget.
- void `vertical_label_margin` (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int `visible` () const

Returns whether a widget is visible.
- unsigned int `visible_focus` () const

Checks whether this widget has a visible focus.
- void `visible_focus` (int v)

Modifies keyboard focus navigation.
- int `visible_r` () const

Returns whether a widget and all its parents are visible.
- int `w` () const

Gets the widget width.
- `Fl_When` `when` () const

Returns the conditions under which the callback is called.

- void [when](#) ([uchar](#) i)
Sets the flags used to decide when a callback is called.
- [FL_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual [~FL_Widget](#) ()
Destroys the widget.

Public Attributes

- [FL_Scrollbar](#) **hscrollbar**
- [FL_Scrollbar](#) **scrollbar**

Protected Member Functions

- void [bbox](#) (int &, int &, int &, int &) const
Returns the bounding box for the interior of the scrolling area, inside the scrollbars.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void [fix_scrollbar_order](#) ()
Ensure the scrollbars are the last children.
- int [on_insert](#) ([FL_Widget](#) *, int) [FL_OVERRIDE](#)
Change insert position of a child before it is added.
- int [on_move](#) (int, int) [FL_OVERRIDE](#)
Change new position of a child before it is moved.
- void [recalc_scrollbars](#) ([ScrollInfo](#) &si) const
Calculate visibility/size/position of scrollbars, find children's bounding box.

Protected Member Functions inherited from [FL_Group](#)

- [FL_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw_child](#) ([FL_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [FL_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual void [on_remove](#) (int)
Allow derived groups to act when a child widget is removed from the group.
- int * [sizes](#) ()
Returns the internal array of widget sizes and positions.
- void [update_child](#) ([FL_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * **current** ()
Returns the currently active group.
- static void **current** ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from Fl_Widget

```
enum {
    INACTIVE = 1<<0, INVISIBLE = 1<<1, OUTPUT = 1<<2, NOBORDER = 1<<3,
    FORCE_POSITION = 1<<4, NON_MODAL = 1<<5, SHORTCUT_LABEL = 1<<6, CHANGED = 1<<7,
    OVERRIDE = 1<<8, VISIBLE_FOCUS = 1<<9, COPIED_LABEL = 1<<10, CLIP_CHILDREN = 1<<11,
    MENU_WINDOW = 1<<12, TOOLTIP_WINDOW = 1<<13, MODAL = 1<<14, NO_OVERLAY = 1<<15,
    GROUP_RELATIVE = 1<<16, COPIED_TOOLTIP = 1<<17, FULLSCREEN = 1<<18, MAC_USE_ACCENTS_MENU = 1<<19,
    NEEDS_KEYBOARD = 1<<20, IMAGE_BOUND = 1<<21, DEIMAGE_BOUND = 1<<22,
    AUTO_DELETE_USER_DATA = 1<<23,
    MAXIMIZED = 1<<24, POPUP = 1<<25, USERFLAG3 = 1<<29, USERFLAG2 = 1<<30,
    USERFLAG1 = 1<<31 }
```

flags possible values enumeration.

11.123.1 Detailed Description

This container widget lets you maneuver around a set of widgets much larger than your window.

If the child widgets are larger than the size of this object then scrollbars will appear so that you can scroll over to them:

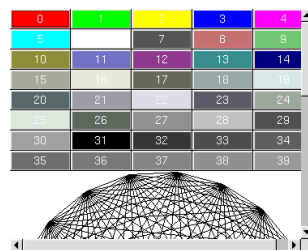


Figure 11.41 Fl_Scroll

If all of the child widgets are packed together into a solid rectangle then you want to set `box()` to `FL_NO_BOX` or one of the `_FRAME` types. This will result in the best output. However, if the child widgets are a sparse arrangement you must set `box()` to a real `_BOX` type. This can result in some blinking during redrawing, but that can be solved by using a `Fl_Double_Window`.

The `Fl_Scroll` widget calculates the bounding box of all its children by using their widget positions and sizes (x, y, w, h). Outside labels are not considered. If you need outside labels of any widgets or free space outside of this bounding box you can add a tiny invisible `Fl_Box` at the relevant corner(s) of the `Fl_Scroll` widget, for instance:

```
Fl_Scroll scroll(100, 100, 200, 200); // Fl_Scroll at (100, 100)
Fl_Box(100, 100, 1, 1); // Fl_Box in top left corner
Fl_Input(150, 120, 60, 30, "Input:"); // left most widget with label
// ... more widgets ...
scroll.end();
```

By default you can scroll in both directions, and the scrollbars disappear if the data will fit in the area of the scroll.

Use `Fl_Scroll::type()` to change this as follows :

- 0 - No scrollbars
- `Fl_Scroll::HORIZONTAL` - Only a horizontal scrollbar.
- `Fl_Scroll::VERTICAL` - Only a vertical scrollbar.
- `Fl_Scroll::BOTH` - The default is both scrollbars.
- `Fl_Scroll::HORIZONTAL_ALWAYS` - Horizontal scrollbar always on, vertical always off.
- `Fl_Scroll::VERTICAL_ALWAYS` - Vertical scrollbar always on, horizontal always off.
- `Fl_Scroll::BOTH_ALWAYS` - Both always on.

Use `scrollbar.align(int)` (see void [Fl_Widget::align\(Fl_Align\)](#)) : to change what side the scrollbars are drawn on.

If the `FL_ALIGN_LEFT` bit is on, the vertical scrollbar is on the left. If the `FL_ALIGN_TOP` bit is on, the horizontal scrollbar is on the top. Note that only the alignment flags in scrollbar are considered. The flags in `hscrollbar` however are ignored.

This widget can also be used to pan around a single child widget "canvas". This child widget should be of your own class, with a `draw()` method that draws the contents. The scrolling is done by changing the `x()` and `y()` of the widget, so this child must use the `x()` and `y()` to position its drawing. To speed up drawing it should test `fl_not_clipped(int x,int y,int w,int h)` to find out if a particular area of the widget must be drawn.

Another very useful child is a single [Fl_Pack](#), which is itself a group that packs its children together and changes size to surround them. Filling the [Fl_Pack](#) with [Fl_Tabs](#) groups (and then putting normal widgets inside those) gives you a very powerful scrolling list of individually-openable panels.

Fluid lets you create these, but you can only lay out objects that fit inside the [Fl_Scroll](#) without scrolling. Be sure to leave space for the scrollbars, as Fluid won't show these either.

You cannot use [Fl_Window](#) as a child of this since the clipping is not conveyed to it when drawn, and it will draw over the scrollbars and neighboring objects.

11.123.2 Constructor & Destructor Documentation

11.123.2.1 Fl_Scroll()

```
Fl_Scroll::Fl_Scroll (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Scroll](#) widget using the given position, size, and label string.

The default boxtype is `FL_NO_BOX`.

The destructor *also deletes all the children*. This allows a whole tree to be deleted at once, without having to keep a pointer to all the children in the user code. A kludge has been done so the [Fl_Scroll](#) and all of its children can be automatic (local) variables, but you must declare the [Fl_Scroll](#) *first*, so that it is destroyed last.

11.123.2.2 ~Fl_Scroll()

```
Fl_Scroll::~~Fl_Scroll () [virtual]
```

The destructor also deletes all the children.

See also

[Fl_Group::~~Fl_Group\(\)](#)

11.123.3 Member Function Documentation

11.123.3.1 bbox()

```
void Fl_Scroll::bbox (
    int & X,
    int & Y,
    int & W,
    int & H) const [protected]
```

Returns the bounding box for the interior of the scrolling area, inside the scrollbars.

This method does not change the scrollbars or their visibility. First the scrollbar positions and visibility are calculated as they **should** be, according to the positions and sizes of the children. Then the bounding box is calculated.

You may need to call [redraw\(\)](#) to make sure the widget gets updated.

See also

[recalc_scrollbars\(\)](#)

11.123.3.2 delete_child()

```
int Fl_Scroll::delete_child (
    int index) [virtual]
```

Removes the widget at `index` from the group and deletes it.

This method does nothing if `index` is out of bounds or if `Fl_Group::child(index)` is one of the scrollbars.

Parameters

<code>in</code>	<code>index</code>	index of child to be removed

Returns

success (0) or error code

Return values

0	success
1	index out of range
2	widget not allowed to be removed (see note)

See also

[Fl_Group::delete_child\(int index\)](#)

Since

FLTK 1.4.0

Reimplemented from [Fl_Group](#).

11.123.3.3 draw()

```
void Fl_Scroll::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Reimplemented from [Fl_Group](#).

11.123.3.4 fix_scrollbar_order()

```
void Fl_Scroll::fix_scrollbar_order () [protected]
```

Ensure the scrollbars are the last children.

When [Fl_Scroll](#) is instantiated the first child of the [Fl_Group](#) is the vertical scrollbar `scrollbar` and the second child is the horizontal scrollbar `hscrollbar`.

These two widgets must always be the last two widgets and in this order to guarantee the correct drawing order and event delivery.

Since FLTK 1.4.0 the new method [on_insert\(\)](#) modifies the insert position of other children if it would be after the scrollbars.

11.123.3.5 handle()

```
int Fl_Scroll::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Group](#).

11.123.3.6 on_insert()

```
int Fl_Scroll::on_insert (
    Fl_Widget * candidate,
    int index) [protected], [virtual]
```

Change insert position of a child before it is added.

Fix insert position if the new child is planned to be inserted after the scrollbars. We can assume that the scrollbars are always the last two children!

[Fl_Group](#) calls this when a new widget is added. We return the new index if the new widget would be added after the scrollbars.

Parameters

in	<i>candidate</i>	the candidate will be added to the child array_ after this method returns.
in	<i>index</i>	add the child at this position in the array_

Returns

index to position the child as planned

a new index to force the child to a different position

-1 to keep the group from adding the candidate

Version

1.4.0

See also

[Fl_Group::on_insert\(Fl_Widget *candidate, int index\)](#)

Reimplemented from [Fl_Group](#).

11.123.3.7 on_move()

```
int Fl_Scroll::on_move (
    int old_index,
    int new_index) [protected], [virtual]
```

Change new position of a child before it is moved.

Fix new position if the new child is planned to be moved after the scrollbars. We can assume that the scrollbars are always the last two children!

[Fl_Group](#) calls this when a widget is moved within the list of children. We return a new index if the widget would be moved after the scrollbars.

Parameters

<i>old_index</i>	the current index of the child that will be moved
<i>new_index</i>	the new index of the child

Returns

new index, possibly corrected to avoid last two scrollbar entries

Reimplemented from [Fl_Group](#).

11.123.3.8 recalc_scrollbars()

```
void Fl_Scroll::recalc_scrollbars (
    ScrollInfo & si) const [protected]
```

Calculate visibility/size/position of scrollbars, find children's bounding box.

The *si* parameter will be filled with data from the calculations. Derived classes can make use of this call to figure out the scrolling area eg. during [resize\(\)](#) handling.

This method does not change the scrollbars or their visibility. It calculates the scrollbar positions and visibility as they **should** be, according to the positions and sizes of the children.

You may need to call [redraw\(\)](#) to make sure the widget gets updated.

Parameters

<i>in, out</i>	<i>si</i>	– ScrollInfo structure, filled with data

See also

[bbox\(\)](#)

11.123.3.9 resize()

```
void Fl_Scroll::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Resizes the [Fl_Scroll](#) widget and moves its children if necessary.

The [Fl_Scroll](#) widget first resizes itself, and then it moves all its children if (and only if) the [Fl_Scroll](#) widget has been moved. The children are moved by the same amount as the [Fl_Scroll](#) widget has been moved, hence all children keep their relative positions.

Note

[Fl_Scroll::resize\(\)](#) does **not** call [Fl_Group::resize\(\)](#), and child widgets are **not** resized.

Since children of an [Fl_Scroll](#) are not resized, the [resizable\(\)](#) widget is ignored (if it is set).

The scrollbars are moved to their proper positions, as given by [Fl_Scroll::scrollbar.align\(\)](#), and switched on or off as necessary.

Note

Due to current (FLTK 1.3.x) implementation constraints some of this may effectively be postponed until the [Fl_Scroll](#) is drawn the next time. This may change in a future release.

See also

[Fl_Group::resizable\(\)](#)

[Fl_Widget::resize\(int,int,int,int\)](#)

Reimplemented from [Fl_Group](#).

11.123.3.10 scroll_to()

```
void Fl_Scroll::scroll_to (
    int X,
    int Y)
```

Moves the contents of the scroll group to a new position.

This is like moving the scrollbars of the [Fl_Scroll](#) around. For instance:

```
Fl_Scroll scroll (10,10,200,200);
Fl_Box b1 ( 10, 10,50,50,"b1"); // relative (x,y) = (0,0)
Fl_Box b2 ( 60, 60,50,50,"b2"); // relative (x,y) = (50,50)
Fl_Box b3 ( 60,110,50,50,"b3"); // relative (x,y) = (50,100)
// populate scroll with more children ...
scroll.end();
scroll.scroll_to(50,100);
```

will move the logical origin of the internal scroll area to (-50,-100) relative to the origin of the [Fl_Scroll](#) (10,10), i.e. [Fl_Box](#) b3 will be visible in the top left corner of the scroll area.

11.123.3.11 scrollbar_size() [1/2]

```
int Fl_Scroll::scrollbar_size () const [inline]
```

Gets the current size of the scrollbars' troughs, in pixels.

If this value is zero (default), this widget will use the [Fl::scrollbar_size\(\)](#) value as the scrollbar's width.

Returns

Scrollbar size in pixels, or 0 if the global [Fl::scrollbar_size\(\)](#) is being used.

See also

[Fl::scrollbar_size\(int\)](#)

11.123.3.12 scrollbar_size() [2/2]

```
void Fl_Scroll::scrollbar_size (
    int newSize) [inline]
```

Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.

Normally you should not need this method, and should use [Fl::scrollbar_size\(int\)](#) instead to manage the size of ALL your widgets' scrollbars. This ensures your application has a consistent UI, is the default behavior, and is normally what you want.

Only use THIS method if you really need to override the global scrollbar size. The need for this should be rare.

Setting `newSize` to the special value of 0 causes the widget to track the global [Fl::scrollbar_size\(\)](#), which is the default.

Parameters

in	newSize	Sets the scrollbar size in pixels. If 0 (default), scrollbar size tracks the global Fl::scrollbar_size()

See also

[Fl::scrollbar_size\(\)](#)

11.123.3.13 xposition()

```
int Fl_Scroll::xposition () const [inline]
```

Gets the current horizontal scrolling position.

11.123.3.14 yposition()

```
int Fl_Scroll::yposition () const [inline]
```

Gets the current vertical scrolling position.

The documentation for this class was generated from the following files:

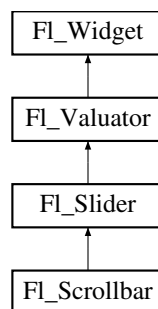
- Fl_Scroll.H
- Fl_Scroll.cxx

11.124 Fl_Scrollbar Class Reference

The [Fl_Scrollbar](#) widget displays a slider with arrow buttons at the ends of the scrollbar.

```
#include <Fl_Scrollbar.H>
```

Inheritance diagram for Fl_Scrollbar:



Public Member Functions

- [Fl_Scrollbar](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Scrollbar](#) widget with given position, size, and label.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- int **linesize** () const
Get the size of step, in lines, that the arrow keys move.
- void [linesize](#) (int i)
This number controls how big the steps are that the arrow keys do.
- int [value](#) () const

- *Gets the integer value (position) of the slider in the scrollbar.*
- `int value (int p)`
Sets the value (position) of the slider in the scrollbar.
- `int value (int pos, int windowSize, int first_line, int total_lines)`
Sets the position, size and range of the slider in the scrollbar.
- `~FI_Scrollbar ()`
Destroys the Scrollbar.

Public Member Functions inherited from `FI_Slider`

- `void bounds (double a, double b)`
Sets the minimum (a) and maximum (b) values for the valuator widget.
- `FI_Slider (int X, int Y, int W, int H, const char *L=0)`
Creates a new `FI_Slider` widget using the given position, size, and label string.
- `FI_Slider (uchar t, int X, int Y, int W, int H, const char *L)`
Creates a new `FI_Slider` widget using the given type, position, size, and label string.
- `int handle (int) FL_OVERRIDE`
Handles the specified event.
- `int scrollvalue (int pos, int size, int first, int total)`
Sets the size and position of the sliding knob in the box.
- `FI_Boxtype slider () const`
Gets the slider box type.
- `void slider (FI_Boxtype c)`
Sets the slider box type.
- `float slider_size () const`
Get the dimensions of the moving piece of slider.
- `void slider_size (double v)`
Set the dimensions of the moving piece of slider.

Public Member Functions inherited from `FI_Valuator`

- `void bounds (double a, double b)`
Sets the minimum (a) and maximum (b) values for the valuator widget.
- `double clamp (double)`
Clamps the passed value to the valuator range.
- `virtual int format (char *)`
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- `double increment (double, int)`
Adds n times the step value to the passed value.
- `double maximum () const`
Gets the maximum value for the valuator.
- `void maximum (double a)`
Sets the maximum value for the valuator.
- `double minimum () const`
Gets the minimum value for the valuator.
- `void minimum (double a)`
Sets the minimum value for the valuator.
- `void precision (int digits)`
Sets the step value to $1.0 / 10^{\text{digits}}$.
- `void range (double a, double b)`
Sets the minimum and maximum values for the valuator.
- `double round (double)`

- Round the passed value to the nearest step increment.*

 - double **step** () const
 - Gets or sets the step value.*
 - void **step** (double a, int b)
 - See double [FL_Valuator::step\(\)](#) const*
 - void **step** (double s)
 - See double [FL_Valuator::step\(\)](#) const.*
 - void **step** (int a)
 - See double [FL_Valuator::step\(\)](#) const*
- double **value** () const
- Gets the floating point(double) value.*
- int **value** (double)
- Sets the current value.*
- ~**FL_Valuator** () **FL_OVERRIDE**
- Destructor is accessible despite protected constructor.*

Public Member Functions inherited from [FL_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
- Activates the widget.*
- unsigned int **active** () const
- Returns whether the widget is active.*
- int **active_r** () const
- Returns whether the widget and all of its parents are active.*
- [FL_Align](#) **align** () const
- Gets the label alignment.*
- void **align** ([FL_Align](#) alignment)
- Sets the label alignment.*
- long **argument** () const
- Gets the current user data (long) argument that is passed to the callback function.*
- void **argument** (long v)
- Sets the current user data (long) argument that is passed to the callback function.*
- virtual class [FL_Gl_Window](#) * **as_gl_window** ()
- Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).*
- virtual class [FL_Gl_Window](#) const * **as_gl_window** () const
- virtual [FL_Group](#) * **as_group** ()
- Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).*
- virtual [FL_Group](#) const * **as_group** () const
- virtual [FL_Window](#) * **as_window** ()
- Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).*
- virtual [FL_Window](#) const * **as_window** () const
- void **bind_deimage** ([FL_Image](#) *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void **bind_deimage** (int f)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- void **bind_image** ([FL_Image](#) *img)
- Sets the image to use as part of the widget label when in the active state.*
- void **bind_image** (int f)

- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*

 - [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
 - void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
 - [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
 - void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
 - void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
 - void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
 - void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
 - unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
 - void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
 - void [clear_changed](#) ()
Marks the value of the widget as unchanged.
 - void [clear_damage](#) ([uchar](#) c=0)
Clears or sets the damage flags.
 - void [clear_output](#) ()
Sets a widget to accept input.
 - void [clear_visible](#) ()
Hides the widget.
 - void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
 - [FI_Color](#) [color](#) () const
Gets the background color of the widget.
 - void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
 - void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
 - [FI_Color](#) [color2](#) () const
For back compatibility only.
 - void [color2](#) (unsigned a)
For back compatibility only.
 - int [contains](#) (const [FI_Widget](#) *w) const
Checks if w is a child of this widget.
 - void [copy_label](#) (const char *new_label)
Sets the current label.
 - void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
 - [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
 - void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.

- void **damage** (uchar c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int **damage_resize** (int, int, int, int)
Internal use only.
- void **deactivate** ()
Deactivates the widget.
- **FL_Image** * **deimage** ()
Gets the image that is used as part of the widget label when in the inactive state.
- const **FL_Image** * **deimage** () const
Gets the image that is used as part of the widget label when in the inactive state.
- void **deimage** (**FL_Image** &img)
Sets the image to use as part of the widget label when in the inactive state.
- void **deimage** (**FL_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- int **deimage_bound** () const
Returns whether the inactive image is managed by the widget.
- void **do_callback** (**FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)
Calls the widget callback function with default arguments.
- void **do_callback** (**FL_Widget** *widget, long arg, **FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)
Calls the widget callback function with arbitrary arguments.
- void **do_callback** (**FL_Widget** *widget, void *arg=0, **FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)
Calls the widget callback function with arbitrary arguments.
- void **draw_label** (int, int, int, int, **FL_Align**) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int **h** () const
Gets the widget height.
- virtual void **hide** ()
Makes a widget invisible.
- int **horizontal_label_margin** ()
Get the spacing between the label and the horizontal edge of the widget.
- void **horizontal_label_margin** (int px)
Set the spacing between the label and the horizontal edge of the widget.
- **FL_Image** * **image** ()
Gets the image that is used as part of the widget label when in the active state.
- const **FL_Image** * **image** () const
Gets the image that is used as part of the widget label when in the active state.
- void **image** (**FL_Image** &img)
Sets the image to use as part of the widget label when in the active state.
- void **image** (**FL_Image** *img)
Sets the image to use as part of the widget label when in the active state.
- int **image_bound** () const
Returns whether the image is managed by the widget.
- int **inside** (const **FL_Widget** *wgt) const
Checks if this widget is a child of wgt.
- int **is_label_copied** () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * **label** () const
Gets the current label text.
- void **label** (const char *text)
Sets the current label pointer.
- void **label** (**FL_Labeltype** a, const char *b)

- Shortcut to set the label text and type in one call.*

 - int [label_image_spacing](#) ()

Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)

Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const

Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)

Sets the label color.
- [FL_Font](#) [labelfont](#) () const

Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)

Sets the font to use.
- [FL_Fontsize](#) [labelsize](#) () const

Gets the font size in pixels.
- void [labelsize](#) ([FL_Fontsize](#) pix)

Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const

Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)

Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.

- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const

- Gets the widget width.*
- [FL_When](#) [when](#) () const
 - Returns the conditions under which the callback is called.*
- void [when](#) (uchar i)
 - Sets the flags used to decide when a callback is called.*
- [FL_Window](#) * [window](#) () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
- int [x](#) () const
 - Gets the widget position in its window.*
- int [y](#) () const
 - Gets the widget position in its window.*
- virtual [~FL_Widget](#) ()
 - Destroys the widget.*

Protected Member Functions

- void [draw](#) () [FL_OVERRIDE](#)
 - Draws the widget.*

Protected Member Functions inherited from [FL_Slider](#)

- void [draw](#) () [FL_OVERRIDE](#)
 - Draws the widget.*
- void [draw](#) (int, int, int, int)
- int [handle](#) (int, int, int, int, int)

Protected Member Functions inherited from [FL_Valuator](#)

- [FL_Valuator](#) (int X, int Y, int W, int H, const char *L)
 - Creates a new [FL_Valuator](#) widget using the given position, size, and label string.*
- void [handle_drag](#) (double newvalue)
 - Called during a drag operation, after an [FL_WHEN_CHANGED](#) event is received and before the callback.*
- void [handle_push](#) ()
 - Stores the current value in the previous value.*
- void [handle_release](#) ()
 - Called after an [FL_WHEN_RELEASE](#) event is received and before the callback.*
- int [horizontal](#) () const
 - Tells if the valuator is an [FL_HORIZONTAL](#) one.*
- double [previous_value](#) () const
 - Gets the previous floating point value before an event changed it.*
- void [set_value](#) (double v)
 - Sets the current floating point value.*
- double [softclamp](#) (double)
 - Clamps the value, but accepts v if the previous value is not already out of range.*
- virtual void [value_damage](#) ()
 - Asks for partial redraw.*

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.124.1 Detailed Description

The [Fl_Scrollbar](#) widget displays a slider with arrow buttons at the ends of the scrollbar.

Clicking on the arrows move up/left and down/right by [linesize\(\)](#). Scrollbars also accept FL_SHORTCUT events: the arrows move by [linesize\(\)](#), and vertical scrollbars take Page Up/Down (they move by the page size minus [linesize\(\)](#)) and Home/End (they jump to the top or bottom).

Scrollbars have [step\(1\)](#) preset (they always return integers). If desired you can set the [step\(\)](#) to non-integer values. You will then have to use casts to get at the floating-point versions of [value\(\)](#) from [Fl_Slider](#).

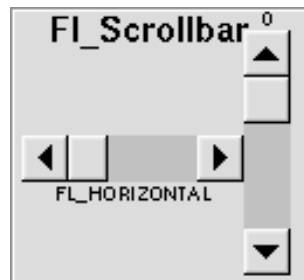


Figure 11.42 [Fl_Scrollbar](#)

11.124.2 Constructor & Destructor Documentation

11.124.2.1 [Fl_Scrollbar\(\)](#)

```
Fl_Scrollbar::Fl_Scrollbar (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Scrollbar](#) widget with given position, size, and label.

You need to do `type(FL_HORIZONTAL)` if you want a horizontal scrollbar.

11.124.3 Member Function Documentation

11.124.3.1 [draw\(\)](#)

```
void Fl_Scrollbar::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.124.3.2 [handle\(\)](#)

```
int Fl_Scrollbar::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

<code>in</code>	<code>event</code>	the kind of event received

Return values

<code>0</code>	if the event was not used or understood
<code>1</code>	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

11.124.3.3 [linesize\(\)](#)

```
void Fl_Scrollbar::linesize (
    int i) [inline]
```

This number controls how big the steps are that the arrow keys do.

In addition page up/down move by the size last sent to [value\(\)](#) minus one [linesize\(\)](#). The default is 16.

11.124.3.4 [value\(\)](#) [1/3]

```
int Fl_Scrollbar::value () const [inline]
```

Gets the integer value (position) of the slider in the scrollbar.

You can get the floating point value with [Fl_Slider::value\(\)](#).

See also

[Fl_Scrollbar::value\(int p\)](#)

[Fl_Scrollbar::value\(int pos, int size, int first, int total\)](#)

11.124.3.5 value() [2/3]

```
int Fl_Scrollbar::value (
    int p) [inline]
```

Sets the value (position) of the slider in the scrollbar.

See also

[Fl_Scrollbar::value\(\)](#)

[Fl_Scrollbar::value\(int pos, int size, int first, int total\)](#)

11.124.3.6 value() [3/3]

```
int Fl_Scrollbar::value (
    int pos,
    int windowSize,
    int first_line,
    int total_lines) [inline]
```

Sets the position, size and range of the slider in the scrollbar.

Parameters

in	<i>pos</i>	position, first line displayed
in	<i>windowSize</i>	number of lines displayed
in	<i>first_line</i>	number of first line
in	<i>total_lines</i>	total number of lines

You should call this every time your window changes size, your data changes size, or your scroll position changes (even if in response to a callback from this scrollbar). All necessary calls to [redraw\(\)](#) are done.

Calls [Fl_Slider::scrollvalue\(int pos, int size, int first, int total\)](#).

The documentation for this class was generated from the following files:

- [Fl_Scrollbar.H](#)
- [Fl_Scrollbar.cxx](#)

11.125 Fl_Scroll::Fl_Scrollbar_Data Struct Reference

A local struct to manage a scrollbar's xywh region and tab values.

```
#include <Fl_Scroll.H>
```

Public Attributes

- int **first**
scrollbar tab's "number of first line"
- int **h**
- int **pos**
scrollbar tab's "position of first line displayed"
- int **size**
scrollbar tab's "size of window in lines"
- int **total**
scrollbar tab's "total number of lines"
- int **w**
- int **x**
- int **y**

11.125.1 Detailed Description

A local struct to manage a scrollbar's xywh region and tab values.

The documentation for this struct was generated from the following file:

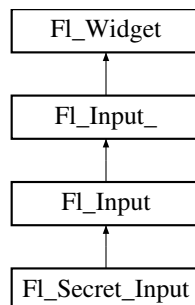
- Fl_Scroll.H

11.126 Fl_Secret_Input Class Reference

The [Fl_Secret_Input](#) class is a subclass of [Fl_Input](#) that displays its input as a string of placeholders.

```
#include <Fl_Secret_Input.H>
```

Inheritance diagram for [Fl_Secret_Input](#):



Public Member Functions

- [Fl_Secret_Input](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [Fl_Secret_Input](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.

Public Member Functions inherited from [Fl_Input](#)

- [Fl_Input](#) (int, int, int, int, const char *l=0)
Creates a new [Fl_Input](#) widget using the given position, size, and label string.

Public Member Functions inherited from [Fl_Input_](#)

- int [append](#) (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool [can_redo](#) () const
Check if there is a redo action available.
- bool [can_undo](#) () const
Check if the last operation can be undone.
- int [copy](#) (int clipboard)
Put the current selection into the clipboard.
- int [copy_cuts](#) ()
Copies the yank buffer to the clipboard.
- [Fl_Color](#) [cursor_color](#) () const
Gets the color of the cursor.
- void [cursor_color](#) ([Fl_Color](#) n)
Sets the color of the cursor.
- int [cut](#) ()
Deletes the current selection.
- int [cut](#) (int a, int b)

- Deletes all characters between index *a* and *b*.*

 - int `cut` (int *n*)

*Deletes the next *n* bytes rounded to characters before or after the cursor.*
- double `dvalue` () const

Returns the widget text interpreted as a floating point number.
- `FL_Input_` (int, int, int, int, const char **=0*)

Creates a new `FL_Input_` widget.
- unsigned int `index` (int *i*) const

*Returns the character at index *i*.*
- int `input_type` () const

Gets the input field type.
- void `input_type` (int *t*)

Sets the input field type.
- int `insert` (const char **t*, int *l=0*)

Inserts text at the cursor position.
- int `insert_position` () const

Gets the position of the text cursor.
- int `insert_position` (int *p*)

Sets the cursor position and mark.
- int `insert_position` (int *p*, int *m*)

Sets the index for the cursor and mark.
- int `ivalue` () const

Returns the widget text interpreted as a signed integer.
- int `mark` () const

Gets the current selection mark.
- int `mark` (int *m*)

Sets the current selection mark.
- int `maximum_size` () const

Gets the maximum length of the input field in characters.
- void `maximum_size` (int *m*)

Sets the maximum length of the input field in characters.
- int `position` () const
- int `position` (int *p*)
- int `position` (int *p*, int *m*)
- int `readonly` () const

Gets the read-only state of the input field.
- void `readonly` (int *b*)

Sets the read-only state of the input field.
- int `redo` ()

Redo previous undo operation.
- int `replace` (int *b*, int *e*, const char **text*, int *ilen=0*)

*Deletes text from *b* to *e* and inserts the new string *text*.*
- void `resize` (int, int, int, int) `FL_OVERRIDE`

Changes the size of the widget.
- int `shortcut` () const

Return the shortcut key associated with this widget.
- void `shortcut` (int *s*)

Sets the shortcut key associated with this widget.
- int `size` () const

*Returns the number of bytes in *value()*.*
- void `size` (int *W*, int *H*)

- Sets the width and height of this widget.*
- int [static_value](#) (const char *)
 - Changes the widget text.*
- int [static_value](#) (const char *, int)
 - Changes the widget text.*
- int [tab_nav](#) () const
 - Gets whether the Tab key causes focus navigation in multiline input fields or not.*
- void [tab_nav](#) (int val)
 - Sets whether the Tab key does focus navigation, or inserts tab characters into [FI_Multiline_Input](#).*
- [FI_Color](#) [textcolor](#) () const
 - Gets the color of the text in the input field.*
- void [textcolor](#) ([FI_Color](#) n)
 - Sets the color of the text in the input field.*
- [FI_Font](#) [textfont](#) () const
 - Gets the font of the text in the input field.*
- void [textfont](#) ([FI_Font](#) s)
 - Sets the font of the text in the input field.*
- [FI_Fonsize](#) [textsize](#) () const
 - Gets the size of the text in the input field.*
- void [textsize](#) ([FI_Fonsize](#) s)
 - Sets the size of the text in the input field.*
- int [undo](#) ()
 - Undoes previous changes to the text buffer.*
- const char * [value](#) () const
 - Returns the text displayed in the widget.*
- int [value](#) (const char *)
 - Changes the widget text.*
- int [value](#) (const char *, int)
 - Changes the widget text.*
- int [value](#) (double value)
 - Changes the widget text to a floating point number ("%g").*
- int [value](#) (int value)
 - Changes the widget text to a signed integer number.*
- int [wrap](#) () const
 - Gets the word wrapping state of the input field.*
- void [wrap](#) (int b)
 - Sets the word wrapping state of the input field.*
- [~FI_Input_](#) ()
 - Destroys the widget.*

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
 - Activates the widget.*
- unsigned int [active](#) () const
 - Returns whether the widget is active.*
- int [active_r](#) () const
 - Returns whether the widget and all of its parents are active.*
- [FI_Align](#) [align](#) () const

- Gets the label alignment.*

 - void [align](#) ([Fl_Align](#) alignment)
- Sets the label alignment.*

 - long [argument](#) () const
- Gets the current user data (long) argument that is passed to the callback function.*

 - void [argument](#) (long v)
- Sets the current user data (long) argument that is passed to the callback function.*

 - virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()
- Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).*

 - virtual class [Fl_Gl_Window](#) const * [as_gl_window](#) () const
- Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).*

 - virtual [Fl_Group](#) * [as_group](#) ()
- Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).*

 - virtual [Fl_Window](#) const * [as_window](#) () const
- Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).*

 - virtual [Fl_Window](#) const * [as_window](#) () const
- Sets the image to use as part of the widget label when in the inactive state.*

 - void [bind_deimage](#) ([Fl_Image](#) *img)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*

 - void [bind_image](#) ([Fl_Image](#) *img)
- Sets the image to use as part of the widget label when in the active state.*

 - void [bind_image](#) (int f)
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*

 - [Fl_Boxtype](#) [box](#) () const
- Gets the box type of the widget.*

 - void [box](#) ([Fl_Boxtype](#) new_box)
- Sets the box type for the widget.*

 - [Fl_Callback_p](#) [callback](#) () const
- Gets the current callback function for the widget.*

 - void [callback](#) ([Fl_Callback](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*

 - void [callback](#) ([Fl_Callback](#) *cb, void *p)
- Sets the current callback function and data for the widget.*

 - void [callback](#) ([Fl_Callback0](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
- Sets the current callback function for the widget.*

 - unsigned int [changed](#) () const
- Checks if the widget value changed since the last callback.*

 - void [clear_active](#) ()
- Marks the widget as inactive without sending events or changing focus.*

 - void [clear_changed](#) ()
- Marks the value of the widget as unchanged.*

 - void [clear_damage](#) (uchar c=0)
- Clears or sets the damage flags.*

 - void [clear_output](#) ()
- Sets a widget to accept input.*

 - void [clear_visible](#) ()

- Hides the widget.*
- void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*
- `FL_Color` `color` () const
- Gets the background color of the widget.*
- void `color` (`FL_Color` bg)
- Sets the background color of the widget.*
- void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*
- `FL_Color` `color2` () const
- For back compatibility only.*
- void `color2` (unsigned a)
- For back compatibility only.*
- int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*
- void `copy_label` (const char *new_label)
- Sets the current label.*
- void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*
- `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*
- void `damage` (`uchar` c)
- Sets the damage bits for the widget.*
- void `damage` (`uchar` c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int `damage_resize` (int, int, int, int)
- Internal use only.*
- void `deactivate` ()
- Deactivates the widget.*
- `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*
- const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void `deimage` (`FL_Image` *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with default arguments.*
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*
- void `draw_label` (int, int, int, int, `FL_Align`) const
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int `h` () const
- Gets the widget height.*
- virtual void `hide` ()
- Makes a widget invisible.*

- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FI_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FI_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FI_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FI_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FI_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FI_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FI_Color c)`
Sets the label color.
- `FI_Font labelfont () const`
Gets the font to use.
- `void labelfont (FI_Font f)`
Sets the font to use.
- `FI_Fonsize labelsiz () const`
Gets the font size in pixels.
- `void labelsiz (FI_Fonsize pix)`
Sets the font size in pixels.
- `FI_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (FI_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`
Sets width ww and height hh accordingly with the label size.
- `bool needs_keyboard () const`
Returns whether this widget needs a keyboard.
- `void needs_keyboard (bool needs)`
Sets whether this widget needs a keyboard.
- `unsigned int output () const`

- Returns if a widget is used for output only.*

 - `Fl_Group * parent () const`

Returns a pointer to the parent widget.
- `void parent (Fl_Group *p)`

Internal use only - "for hacks only".
- `void position (int X, int Y)`

Repositions the window or widget.
- `void redraw ()`

Schedules the drawing of the widget.
- `void redraw_label ()`

Schedules the drawing of the label.
- `Fl_Color selection_color () const`

Gets the selection color.
- `void selection_color (Fl_Color a)`

Sets the selection color.
- `void set_active ()`

Marks the widget as active without sending events or changing focus.
- `void set_changed ()`

Marks the value of the widget as changed.
- `void set_output ()`

Sets a widget to output only.
- `void set_visible ()`

Makes the widget visible.
- `void set_visible_focus ()`

Enables keyboard focus navigation with this widget.
- `int shortcut_label () const`

Returns whether the widget's label uses '&' to indicate shortcuts.
- `void shortcut_label (int value)`

Sets whether the widget's label uses '&' to indicate shortcuts.
- `virtual void show ()`

Makes a widget visible.
- `void size (int W, int H)`

Changes the size of the widget.
- `int take_focus ()`

Gives the widget the keyboard focus.
- `unsigned int takeevents () const`

Returns if the widget is able to take events.
- `int test_shortcut ()`

Returns true if the widget's label contains the entered '&x' shortcut.
- `const char * tooltip () const`

Gets the current tooltip text.
- `void tooltip (const char *text)`

Sets the current tooltip text.
- `Fl_Window * top_window () const`

Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`

Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`

Gets the widget type.
- `void type (uchar t)`

Sets the widget type.

- int **use_accents_menu** ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const
Gets the user data for this widget.
- void **user_data** (FI_Callback_User_Data *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- FI_When **when** () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- FI_Window * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual ~FI_Widget ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Static Public Attributes inherited from FI_Input

- static const char * **copy_menu_text** = "Copy"
[this text may be customized at run-time]
- static const char * **cut_menu_text** = "Cut"
[this text may be customized at run-time]
- static const char * **paste_menu_text** = "Paste"
[this text may be customized at run-time]

Protected Types inherited from FI_Widget

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from FI_Input

- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- int [handle_key](#) ()
Handles a keystroke.
- int [handle_rmb](#) ()
Handle right mouse button down events.

Protected Member Functions inherited from FI_Input_

- int [apply_undo](#) ()
Apply the current undo/redo operation.
- void [drawtext](#) (int, int, int, int)
Draws the text in the passed bounding box.
- void [drawtext](#) (int, int, int, int, bool draw_active)
Draws the text in the passed bounding box.
- void [handle_mouse](#) (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int [handletext](#) (int e, int, int, int, int)
Handles all kinds of text field related events.
- int [line_end](#) (int i) const
Finds the end of a line.
- int [line_start](#) (int i) const
Finds the start of a line.
- int [linesPerPage](#) ()
- void [maybe_do_callback](#) ([FI_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
- int [up_down_position](#) (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int [word_end](#) (int i) const
Finds the end of a word.
- int [word_start](#) (int i) const
Finds the start of a word.
- int [xscroll](#) () const
- int [yscroll](#) () const
- void [yscroll](#) (int yOffset)

Protected Member Functions inherited from [Fl_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([Fl_Boxtype](#) t, [Fl_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([Fl_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [Fl_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.126.1 Detailed Description

The [Fl_Secret_Input](#) class is a subclass of [Fl_Input](#) that displays its input as a string of placeholders. Depending on the platform this placeholder is either the asterisk ('*') or the Unicode bullet character (U+2022). This subclass is usually used to receive passwords and other "secret" information.

11.126.2 Constructor & Destructor Documentation

11.126.2.1 [Fl_Secret_Input](#)()

```
Fl_Secret_Input::Fl_Secret_Input (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Secret_Input](#) widget using the given position, size, and label string. The default boxttype is FL_DOWN_BOX. Inherited destructor destroys the widget and any value associated with it.

11.126.3 Member Function Documentation

11.126.3.1 handle()

```
int Fl_Secret_Input::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Input](#).

The documentation for this class was generated from the following files:

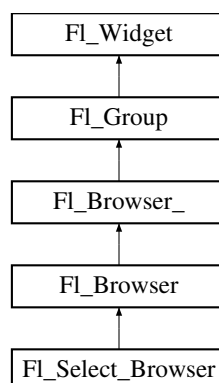
- Fl_Secret_Input.H
- Fl_Input.cxx

11.127 Fl_Select_Browser Class Reference

The class is a subclass of [Fl_Browser](#) which lets the user select a single item, or no items by clicking on the empty space.

```
#include <Fl_Select_Browser.H>
```

Inheritance diagram for Fl_Select_Browser:



Public Member Functions

- [FL_Select_Browser](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [FL_Select_Browser](#) widget using the given position, size, and label string.

Public Member Functions inherited from [FL_Browser](#)

- void [add](#) (const char *newtext, void *d=0)
Adds a new line to the end of the browser.
- void [bottomline](#) (int line)
*Scrolls the browser so the bottom item in the browser is showing the specified *line*.*
- void [clear](#) ()
Removes all the lines in the browser.
- char [column_char](#) () const
Gets the current column separator character.
- void [column_char](#) (char c)
*Sets the column separator to *c*.*
- const int * [column_widths](#) () const
Gets the current column width array.
- void [column_widths](#) (const int *arr)
*Sets the current array to *arr*.*
- void * [data](#) (int line) const
*Returns the user [data\(\)](#) for specified *line*.*
- void [data](#) (int line, void *d)
*Sets the user data for specified *line* to *d*.*
- void [display](#) (int line, int val=1)
For back compatibility.
- int [displayed](#) (int line) const
*Returns non-zero if *line* has been scrolled to a position where it is being displayed.*
- [FL_Browser](#) (int X, int Y, int W, int H, const char *L=0)
The constructor makes an empty browser.
- char [format_char](#) () const
Gets the current format code prefix character, which by default is '@'.
- void [format_char](#) (char c)
*Sets the current format code prefix character to *c*.*
- void [hide](#) () [FL_OVERRIDE](#)
Hides the entire [FL_Browser](#) widget – opposite of [show\(\)](#).
- void [hide](#) (int line)
*Makes *line* invisible, preventing selection by the user.*
- [FL_Image](#) * [icon](#) (int line) const
*Returns the icon currently defined for *line*.*
- void [icon](#) (int line, [FL_Image](#) *icon)
*Set the image icon for *line* to the value *icon*.*
- void [insert](#) (int line, const char *newtext, void *d=0)
*Insert a new entry whose label is *newtext* above given *line*, optional data *d*.*
- void [lineposition](#) (int line, [FL_Line_Position](#) pos)
*Updates the browser so that *line* is shown at position *pos*.*
- int [load](#) (const char *filename)
Clears the browser and reads the file, adding each line from the file to the browser.
- void [make_visible](#) (int line)
*Make the item at the specified *line* [visible\(\)](#).*
- void [middleline](#) (int line)

- Scrolls the browser so the middle item in the browser is showing the specified line.*

 - void **move** (int to, int from)

Line from is removed and reinserted at to.
- void **remove** (int line)

Remove entry for given line number, making the browser one line shorter.
- void **remove_icon** (int line)

Removes the icon for line.
- void **replace** (int a, const char *b)

For back compatibility only.
- int **select** (int line, int val=1)

Sets the selection state of the item at line to the value val.
- int **selected** (int line) const

Returns 1 if specified line is selected, 0 if not.
- void **show** () **FL_OVERRIDE**

Shows the entire Fl_Browser widget – opposite of hide().
- void **show** (int line)

Makes line visible, and available for selection by user.
- int **size** () const

Returns how many lines are in the browser.
- void **size** (int W, int H)
- void **swap** (int a, int b)

Swaps two browser lines a and b.
- const char * **text** (int line) const

Returns the label text for the specified line.
- void **text** (int line, const char *newtext)

Sets the text for the specified line to newtext.
- **Fl_Fontsize** **textsize** () const

Gets the default text size (in pixels) for the lines in the browser.
- void **textsize** (**Fl_Fontsize** newSize)

Sets the default text size (in pixels) for the lines in the browser to newSize.
- int **topline** () const

Returns the line that is currently visible at the top of the browser.
- void **topline** (int line)

Scrolls the browser so the top item in the browser is showing the specified line.
- int **value** () const

Returns the line number of the currently selected line, or 0 if none selected.
- void **value** (int line)

Sets the browser's value(), which selects the specified line.
- int **visible** (int line) const

Returns non-zero if the specified line is visible, 0 if hidden.
- **~Fl_Browser** ()

The destructor deletes all list items and destroys the browser.

Public Member Functions inherited from Fl_Browser_

- int **deselect** (int docallbacks=0)

Deselects all items in the list and returns 1 if the state changed or 0 if it did not.
- void **display** (void *item)

Displays the item, scrolling the list as necessary.
- int **handle** (int event) **FL_OVERRIDE**

Handles the event within the normal widget bounding box.

- **uchar has_scrollbar** () const
Returns the current scrollbar mode, see [FI_Browser_::has_scrollbar\(uchar\)](#)
- void **has_scrollbar** (uchar mode)
Sets whether the widget should have scrollbars or not (default [FI_Browser_::BOTH](#)).
- int **hposition** () const
*Gets the horizontal scroll position of the list as a pixel position *pos*.*
- void **hposition** (int)
*Sets the horizontal scroll position of the list to pixel position *pos*.*
- int **linespacing** () const
Return the height of additional spacing between browser lines.
- void **linespacing** (int pixels)
Add some space between browser lines.
- int **position** () const
- void **position** (int pos)
- void **position** (int *x*, int *y*)
- void **resize** (int *X*, int *Y*, int *W*, int *H*) [FL_OVERRIDE](#)
Repositions and/or resizes the browser.
- void **scrollbar_left** ()
Moves the vertical scrollbar to the lefthand side of the list.
- void **scrollbar_right** ()
Moves the vertical scrollbar to the righthand side of the list.
- int **scrollbar_size** () const
Gets the current size of the scrollbars' troughs, in pixels.
- void **scrollbar_size** (int newSize)
*Sets the pixel size of the scrollbars' troughs to *newSize*, in pixels.*
- int **scrollbar_width** () const
Returns the global value [Fl::scrollbar_size\(\)](#).
- void **scrollbar_width** (int width)
Sets the global [Fl::scrollbar_size\(\)](#), and forces this instance of the widget to use it.
- int **select** (void *item, int val=1, int docallbacks=0)
*Sets the selection state of *item* to *val*, and returns 1 if the state changed or 0 if it did not.*
- int **select_only** (void *item, int docallbacks=0)
*Selects *item* and returns 1 if the state changed or 0 if it did not.*
- void **sort** (int flags=0)
*Sort the items in the browser based on *flags*.*
- [Fl_Color](#) **textcolor** () const
Gets the default text color for the lines in the browser.
- void **textcolor** ([Fl_Color](#) col)
*Sets the default text color for the lines in the browser to color *col*.*
- [Fl_Font](#) **textfont** () const
Gets the default text font for the lines in the browser.
- void **textfont** ([Fl_Font](#) font)
*Sets the default text font for the lines in the browser to *font*.*
- [Fl_Fontsize](#) **textsize** () const
Gets the default text size (in pixels) for the lines in the browser.
- void **textsize** ([Fl_Fontsize](#) newSize)
*Sets the default text size (in pixels) for the lines in the browser to *size*.*
- int **vposition** () const
*Gets the vertical scroll position of the list as a pixel position *pos*.*
- void **vposition** (int pos)
*Sets the vertical scroll position of the list to pixel position *pos*.*

Public Member Functions inherited from FL_Group

- [FL_Widget](#) * & [_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void **add** ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void **add_resizable** ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * **array** () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * **as_group** () const [FL_OVERRIDE](#)
- [FL_Group](#) * **as_group** () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
*Removes the widget at *index* from the group and deletes it.*
- void **end** ()
*Exactly the same as *current(this->parent())*.*
- int **find** (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int **find** (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char * = 0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void **focus** ([FL_Widget](#) *W)
- void **forms_end** ()
This is for forms compatibility only.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** ([FL_Widget](#) &o, [FL_Widget](#) *before)
*This does *insert(w, find(before))*.*
- void **remove** ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void **remove** ([FL_Widget](#) *o)

- Removes the widget `o` from the group.*

 - void `remove` (int index)

Removes the widget at `index` from the group but does not delete it.
- `FL_Widget * resizable` () const

Returns the group's resizable widget.
- void `resizable` (`FL_Widget &o`)

Sets the group's resizable widget.
- void `resizable` (`FL_Widget *o`)

The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- void `resize` (int, int, int, int) `FL_OVERRIDE`

Resizes the `FL_Group` widget and all of its children.
- virtual `~FL_Group` ()

The destructor also deletes all the children.

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
- Activates the widget.*

 - unsigned int `active` () const

Returns whether the widget is active.
- int `active_r` () const

Returns whether the widget and all of its parents are active.
- `FL_Align align` () const

Gets the label alignment.
- void `align` (`FL_Align alignment`)

Sets the label alignment.
- long `argument` () const

Gets the current user data (long) argument that is passed to the callback function.
- void `argument` (long v)

Sets the current user data (long) argument that is passed to the callback function.
- virtual class `FL_Gl_Window * as_gl_window` ()

Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.
- virtual class `FL_Gl_Window` const * `as_gl_window` () const
- virtual `FL_Window * as_window` ()

Returns an `FL_Window` pointer if this widget is an `FL_Window`.
- virtual `FL_Window` const * `as_window` () const
- void `bind_deimage` (`FL_Image *img`)

Sets the image to use as part of the widget label when in the inactive state.
- void `bind_deimage` (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void `bind_image` (`FL_Image *img`)

Sets the image to use as part of the widget label when in the active state.
- void `bind_image` (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- `FL_Boxtype box` () const

Gets the box type of the widget.
- void `box` (`FL_Boxtype new_box`)

Sets the box type for the widget.
- `FL_Callback_p callback` () const

- Gets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, void *p)
- Sets the current callback function and data for the widget.*

 - void [callback](#) ([FI_Callback0](#) *cb)
- Sets the current callback function for the widget.*

 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)
- Sets the current callback function for the widget.*

 - unsigned int [changed](#) () const
- Checks if the widget value changed since the last callback.*

 - void [clear_active](#) ()
- Marks the widget as inactive without sending events or changing focus.*

 - void [clear_changed](#) ()
- Marks the value of the widget as unchanged.*

 - void [clear_damage](#) ([uchar](#) c=0)
- Clears or sets the damage flags.*

 - void [clear_output](#) ()
- Sets a widget to accept input.*

 - void [clear_visible](#) ()
- Hides the widget.*

 - void [clear_visible_focus](#) ()
- Disables keyboard focus navigation with this widget.*

 - [FI_Color](#) [color](#) () const
- Gets the background color of the widget.*

 - void [color](#) ([FI_Color](#) bg)
- Sets the background color of the widget.*

 - void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
- Sets the background and selection color of the widget.*

 - [FI_Color](#) [color2](#) () const
- For back compatibility only.*

 - void [color2](#) (unsigned a)
- For back compatibility only.*

 - int [contains](#) (const [FI_Widget](#) *w) const
- Checks if w is a child of this widget.*

 - void [copy_label](#) (const char *new_label)
- Sets the current label.*

 - void [copy_tooltip](#) (const char *text)
- Sets the current tooltip text.*

 - [uchar](#) [damage](#) () const
- Returns non-zero if [draw\(\)](#) needs to be called.*

 - void [damage](#) ([uchar](#) c)
- Sets the damage bits for the widget.*

 - void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int [damage_resize](#) (int, int, int, int)
- Internal use only.*

 - void [deactivate](#) ()
- Deactivates the widget.*

- `FI_Image * deimage ()`
Gets the image that is used as part of the widget label when in the inactive state.
- `const FI_Image * deimage () const`
Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FI_Image &img)`
Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FI_Image *img)`
Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`
Returns whether the inactive image is managed by the widget.
- `void do_callback (FI_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with default arguments.
- `void do_callback (FI_Widget *widget, long arg, FI_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void do_callback (FI_Widget *widget, void *arg=0, FI_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FI_Align) const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`
Gets the widget height.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FI_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FI_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FI_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FI_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FI_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FI_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FI_Color c)`

- Sets the label color.*
- [Fl_Font](#) [labelfont](#) () const
 - Gets the font to use.*
- void [labelfont](#) ([Fl_Font](#) f)
 - Sets the font to use.*
- [Fl_Fontsize](#) [labelsize](#) () const
 - Gets the font size in pixels.*
- void [labelsize](#) ([Fl_Fontsize](#) pix)
 - Sets the font size in pixels.*
- [Fl_Labeltype](#) [labeltype](#) () const
 - Gets the label type.*
- void [labeltype](#) ([Fl_Labeltype](#) a)
 - Sets the label type.*
- void [measure_label](#) (int &ww, int &hh) const
 - Sets width ww and height hh accordingly with the label size.*
- bool [needs_keyboard](#) () const
 - Returns whether this widget needs a keyboard.*
- void [needs_keyboard](#) (bool needs)
 - Sets whether this widget needs a keyboard.*
- unsigned int [output](#) () const
 - Returns if a widget is used for output only.*
- [Fl_Group](#) * [parent](#) () const
 - Returns a pointer to the parent widget.*
- void [parent](#) ([Fl_Group](#) *p)
 - Internal use only - "for hacks only".*
- void [position](#) (int X, int Y)
 - Repositions the window or widget.*
- void [redraw](#) ()
 - Schedules the drawing of the widget.*
- void [redraw_label](#) ()
 - Schedules the drawing of the label.*
- [Fl_Color](#) [selection_color](#) () const
 - Gets the selection color.*
- void [selection_color](#) ([Fl_Color](#) a)
 - Sets the selection color.*
- void [set_active](#) ()
 - Marks the widget as active without sending events or changing focus.*
- void [set_changed](#) ()
 - Marks the value of the widget as changed.*
- void [set_output](#) ()
 - Sets a widget to output only.*
- void [set_visible](#) ()
 - Makes the widget visible.*
- void [set_visible_focus](#) ()
 - Enables keyboard focus navigation with this widget.*
- int [shortcut_label](#) () const
 - Returns whether the widget's label uses '&' to indicate shortcuts.*
- void [shortcut_label](#) (int value)
 - Sets whether the widget's label uses '&' to indicate shortcuts.*
- void [size](#) (int W, int H)
 - Changes the size of the widget.*

- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- uchar [type](#) () const
Gets the widget type.
- void [type](#) (uchar t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) (uchar i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual ~[Fl_Widget](#) ()
Destroys the widget.

Additional Inherited Members

Public Types inherited from FI_Browser

- enum [FI_Line_Position](#) { TOP , BOTTOM , MIDDLE }

For internal use only?

Public Types inherited from FI_Browser_

- enum {
[HORIZONTAL](#) = 1 , [VERTICAL](#) = 2 , [BOTH](#) = 3 , [ALWAYS_ON](#) = 4 ,
[HORIZONTAL_ALWAYS](#) = 5 , [VERTICAL_ALWAYS](#) = 6 , [BOTH_ALWAYS](#) = 7 }

Values for [has_scrollbar\(\)](#).

Static Public Member Functions inherited from FI_Group

- static [FI_Group](#) * [current](#) ()
Returns the currently active group.
- static void [current](#) ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from FI_Widget

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Public Attributes inherited from FI_Browser_

- [FI_Scrollbar](#) [hscrollbar](#)
Horizontal scrollbar.
- [FI_Scrollbar](#) [scrollbar](#)
Vertical scrollbar.

Protected Types inherited from FI_Widget

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

Protected Member Functions inherited from [FL_Browser](#)

- [FL_BLINE](#) * [_remove](#) (int line)
Removes the item at the specified line.
- [FL_BLINE](#) * [find_line](#) (int line) const
Returns the item for specified line.
- int [full_height](#) () const [FL_OVERRIDE](#)
The height of the entire list of all [visible\(\)](#) items in pixels.
- int [incr_height](#) () const [FL_OVERRIDE](#)
The default 'average' item height (including inter-item spacing) in pixels.
- void [insert](#) (int line, [FL_BLINE](#) *item)
Insert specified item above line.
- void * [item_at](#) (int line) const [FL_OVERRIDE](#)
Return the item at specified line.
- void [item_draw](#) (void *item, int X, int Y, int W, int H) const [FL_OVERRIDE](#)
Draws item at the position specified by X Y W H.
- void * [item_first](#) () const [FL_OVERRIDE](#)
Returns the very first item in the list.
- int [item_height](#) (void *item) const [FL_OVERRIDE](#)
Returns height of item in pixels.
- void * [item_last](#) () const [FL_OVERRIDE](#)
Returns the very last item in the list.
- void * [item_next](#) (void *item) const [FL_OVERRIDE](#)
Returns the next item after item.
- void * [item_prev](#) (void *item) const [FL_OVERRIDE](#)
Returns the previous item before item.
- void [item_select](#) (void *item, int val) [FL_OVERRIDE](#)
Change the selection state of item to the value val.
- int [item_selected](#) (void *item) const [FL_OVERRIDE](#)
See if item is selected.
- void [item_swap](#) (void *a, void *b) [FL_OVERRIDE](#)
Swap the items a and b.
- const char * [item_text](#) (void *item) const [FL_OVERRIDE](#)
Returns the label text for item.
- int [item_width](#) (void *item) const [FL_OVERRIDE](#)
Returns width of item in pixels.
- int [lineno](#) (void *item) const
Returns line number corresponding to item, or zero if not found.
- void [swap](#) ([FL_BLINE](#) *a, [FL_BLINE](#) *b)
Swap the two items a and b.

Protected Member Functions inherited from [FL_Browser_](#)

- void [bbox](#) (int &X, int &Y, int &W, int &H) const
Returns the bounding box for the interior of the list's display window, inside the scrollbars.
- void [deleting](#) (void *item)
This method should be used when item is being deleted from the list.
- int [displayed](#) (void *item) const
Returns non-zero if item has been scrolled to a position where it is being displayed.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the list within the normal widget bounding box.
- void * [find_item](#) (int ypos)

- This method returns the item under mouse y position `ypos`.*
- `Fl_Browser_` (int X, int Y, int W, int H, const char *L=0)
 - The constructor makes an empty browser.*
- virtual int `full_width` () const
 - This method may be provided by the subclass to indicate the full width of the item list, in pixels.*
- void `inserting` (void *a, void *b)
 - This method should be used when an item is in the process of being inserted into the list.*
- virtual int `item_quick_height` (void *item) const
 - This method may be provided by the subclass to return the height of the `item`, in pixels.*
- int `leftedge` () const
 - This method returns the X position of the left edge of the list area after adjusting for the scrollbar and border, if any.*
- void `new_list` ()
 - This method should be called when the list data is completely replaced or cleared.*
- void `redraw_line` (void *item)
 - This method should be called when the contents of `item` has changed, but not its height.*
- void `redraw_lines` ()
 - This method will cause the entire list to be redrawn.*
- void `replacing` (void *a, void *b)
 - This method should be used when item `a` is being replaced by item `b`.*
- void * `selection` () const
 - Returns the item currently selected, or NULL if there is no selection.*
- void `swapping` (void *a, void *b)
 - This method should be used when two items `a` and `b` are being swapped.*
- void * `top` () const
 - Returns the item that appears at the top of the list.*

Protected Member Functions inherited from `Fl_Group`

- `Fl_Rect *` `bounds` ()
 - Returns the internal array of widget sizes and positions.*
- void `draw` () `FL_OVERRIDE`
 - Draws the widget.*
- void `draw_child` (`Fl_Widget` &widget) const
 - Forces a child to redraw.*
- void `draw_children` ()
 - Draws all children of the group.*
- void `draw_outside_label` (const `Fl_Widget` &widget) const
 - Parents normally call this to draw outside labels of child widgets.*
- virtual int `on_insert` (`Fl_Widget` *, int)
 - Allow derived groups to act when a widget is added as a child.*
- virtual int `on_move` (int, int)
 - Allow derived groups to act when a widget is moved within the group.*
- virtual void `on_remove` (int)
 - Allow derived groups to act when a child widget is removed from the group.*
- int * `sizes` ()
 - Returns the internal array of widget sizes and positions.*
- void `update_child` (`Fl_Widget` &widget) const
 - Draws a child only if it needs it.*

Protected Member Functions inherited from [Fl_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([Fl_Boxtype](#) t, [Fl_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([Fl_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [Fl_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.127.1 Detailed Description

The class is a subclass of [Fl_Browser](#) which lets the user select a single item, or no items by clicking on the empty space.

As long as the mouse button is held down on an unselected item it is highlighted. Normally the callback is done when the user presses the mouse, but you can change this with [when\(\)](#).

See [Fl_Browser](#) for methods to add and remove lines from the browser.

11.127.2 Constructor & Destructor Documentation

11.127.2.1 [Fl_Select_Browser\(\)](#)

```
Fl_Select_Browser::Fl_Select_Browser (
    int X,
    int Y,
    int W,
```

```
int H,
const char * L = 0)
```

Creates a new [Fl_Select_Browser](#) widget using the given position, size, and label string.

The default boxtype is FL_DOWN_BOX. The constructor specializes [Fl_Browser\(\)](#) by setting the type to FL_SELECT_BROWSER. The destructor destroys the widget and frees all memory that has been allocated.

The documentation for this class was generated from the following files:

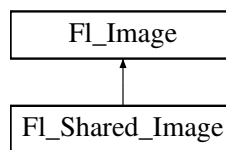
- [Fl_Select_Browser.H](#)
- [Fl_Browser.cxx](#)

11.128 Fl_Shared_Image Class Reference

This class supports caching, loading, and drawing of image files.

```
#include <Fl_Shared_Image.H>
```

Inheritance diagram for Fl_Shared_Image:



Public Member Functions

- [Fl_Shared_Image * as_shared_image \(\)](#) [FL_OVERRIDE](#)
Returns whether an image is an [Fl_Shared_Image](#) or not.
- void [color_average](#) ([Fl_Color](#) c, float i) [FL_OVERRIDE](#)
Averages the colors in the image with the provided FLTK color value.
- [Fl_Image * copy \(\)](#)
Increments the reference counter and returns a pointer to itself.
- [Fl_Image * copy \(\)](#) const
- [Fl_Image * copy](#) (int W, int H) const [FL_OVERRIDE](#)
Return a shared image of this image with the requested size.
- void [desaturate](#) () [FL_OVERRIDE](#)
Convert the image to gray scale.
- void **draw** (int X, int Y)
- void [draw](#) (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draw this image to the current graphics context.
- const [Fl_Image * image](#) () const
Returns a pointer to the internal [Fl_Image](#) object.
- const char * **name** ()
Returns the filename of the shared image.
- int [original](#) ()
Returns whether this is an original image.
- int [refcount](#) ()
Returns the number of references of this shared image.
- void [release](#) () [FL_OVERRIDE](#)
Releases and possibly destroys (if refcount <= 0) a shared image.
- virtual void **reload** ()
Reloads the shared image from disk.
- void [uncache](#) () [FL_OVERRIDE](#)
Remove the cached device specific image data.

Public Member Functions inherited from [Fl_Image](#)

- [Fl_Image](#) * [copy](#) () const
Creates a copy of the image in the same size.
- int [count](#) () const
Returns the number of data values associated with the image.
- int [d](#) () const
Returns the image depth.
- const char *const * [data](#) () const
Returns a pointer to the current image data array.
- int [data_h](#) () const
Returns the height of the image data.
- int [data_w](#) () const
Returns the width of the image data.
- void [draw](#) (int X, int Y)
Draws the image to the current drawing surface.
- int [fail](#) () const
Returns a value that is not 0 if there is currently no image available.
- [Fl_Image](#) (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int [h](#) () const
Returns the current image drawing height in FLTK units.
- void [inactive](#) ()
The [inactive\(\)](#) method calls [color_average\(FL_BACKGROUND_COLOR, 0.33f\)](#) to produce an image that appears grayed out.
- virtual void [label](#) ([Fl_Menu_Item](#) *m)
This method is an obsolete way to set the image attribute of a menu item.
- virtual void [label](#) ([Fl_Widget](#) *w)
This method is an obsolete way to set the image attribute of a widget or menu item.
- int [ld](#) () const
Returns the current line data size in bytes.
- virtual void [scale](#) (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- int [w](#) () const
Returns the current image drawing width in FLTK units.
- virtual ~[Fl_Image](#) ()
The destructor is a virtual method that frees all memory used by the image.

Static Public Member Functions

- static void [add_handler](#) ([Fl_Shared_Handler](#) f)
Adds a shared image handler, which is basically a test function for adding new image formats.
- static [Fl_Shared_Image](#) * [find](#) (const char *name, int W=0, int H=0)
Finds a shared image from its name and size specifications.
- static [Fl_Shared_Image](#) * [get](#) (const char *name, int W=0, int H=0)
Find or load an image that can be shared by multiple widgets.
- static [Fl_Shared_Image](#) * [get](#) ([Fl_RGB_Image](#) *rgb, int own_it=1)
Builds a shared image from a pre-existing [Fl_RGB_Image](#).
- static [Fl_Shared_Image](#) ** [images](#) ()
Returns the [Fl_Shared_Image](#) array.*
- static int [num_images](#) ()
Number of shared images in their various cached sizes.
- static void [remove_handler](#) ([Fl_Shared_Handler](#) f)
Removes a shared image handler.

Static Public Member Functions inherited from FI_Image

- static [FI_Labeltype](#) [define_FL_IMAGE_LABEL](#) ()
- static [FI_RGB_Scaling](#) [RGB_scaling](#) ()
Returns the currently used RGB image scaling method.
- static void [RGB_scaling](#) ([FI_RGB_Scaling](#))
Sets the RGB image scaling method used for copy(int, int).
- static [FI_RGB_Scaling](#) [scaling_algorithm](#) ()
Gets what algorithm is used when resizing a source image to draw it.
- static void [scaling_algorithm](#) ([FI_RGB_Scaling](#) algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Protected Member Functions

- void [add](#) ()
Adds a shared image to the image pool.
- [FI_Shared_Image](#) * [copy](#) (int W, int H) const
Create a resized copy of the image and wrap it into the share image class.
- [FI_Shared_Image](#) ()
Creates an empty shared image.
- [FI_Shared_Image](#) (const char *n, [FI_Image](#) *img=0)
Creates a shared image from its filename and its corresponding FI_Image img.*
- void [update](#) ()
Update the dimensions of the shared images.
- virtual [~FI_Shared_Image](#) ()
The destructor frees all memory and server resources that are used by the image.

Protected Member Functions inherited from FI_Image

- void [d](#) (int D)
Sets the current image depth.
- void [data](#) (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void [draw_empty](#) (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int [draw_scaled](#) (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void [h](#) (int H)
Sets the height of the image data.
- void [ld](#) (int LD)
Sets the current line data size in bytes.
- void [w](#) (int W)
Sets the width of the image data.

Static Protected Member Functions

- static int [compare](#) ([FI_Shared_Image](#) **i0, [FI_Shared_Image](#) **i1)
Compares two shared images.

Static Protected Member Functions inherited from FI_Image

- static void [labeltype](#) (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void [measure](#) (const [FI_Label](#) *lo, int &lw, int &lh)

Protected Attributes

- int **alloc_image_**
- [Fl_Image](#) * **image_**
- const char * **name_**
- int **original_**
- int **refcount_**

Static Protected Attributes

- static int **alloc_handlers_** = 0
- static int **alloc_images_** = 0
- static [Fl_Shared_Handler](#) * **handlers_** = 0
- static [Fl_Shared_Image](#) ** **images_** = 0
- static int **num_handlers_** = 0
- static int **num_images_** = 0

Friends

- class [Fl_Graphics_Driver](#)
- class [Fl_JPEG_Image](#)
- class [Fl_PNG_Image](#)
- class [Fl_SVG_Image](#)

Additional Inherited Members

Static Public Attributes inherited from [Fl_Image](#)

- static const int **ERR_FILE_ACCESS** = -2
- static const int **ERR_FORMAT** = -3
- static const int **ERR_MEMORY_ACCESS** = -4
- static const int **ERR_NO_IMAGE** = -1
- static bool **register_images_done** = false

True after [fl_register_images\(\)](#) was called, false before.

11.128.1 Detailed Description

This class supports caching, loading, and drawing of image files.

Most applications will also want to link against the fltk_images library and call the [fl_register_images\(\)](#) function to support standard image formats such as BMP, GIF, JPEG, PNG, and SVG (unless the library was built with the option removing SVG support).

Images can be requested (loaded) with [Fl_Shared_Image::get\(\)](#), [find\(\)](#), and some other methods. All images are cached in an internal list of shared images and should be released when they are no longer needed. A refcount is used to determine if a released image is to be destroyed with delete.

See also

[fl_register_image\(\)](#)
[Fl_Shared_Image::get\(\)](#)
[Fl_Shared_Image::find\(\)](#)
[Fl_Shared_Image::release\(\)](#)

11.128.2 Constructor & Destructor Documentation

11.128.2.1 [Fl_Shared_Image\(\)](#) [1/2]

```
Fl_Shared_Image::Fl_Shared_Image () [protected]
```

Creates an empty shared image.

The constructors create a new shared image record in the image cache.

The constructors are protected and cannot be used directly from a program. Use the [get\(\)](#) method instead.

11.128.2.2 FI_Shared_Image() [2/2]

```
Fl_Shared_Image::Fl_Shared_Image (
    const char * n,
    Fl_Image * img = 0) [protected]
```

Creates a shared image from its filename and its corresponding `Fl_Image*` `img`.

The constructors create a new shared image record in the image cache.

The constructors are protected and cannot be used directly from a program. Use the [get\(\)](#) method instead.

Parameters

in	<i>n</i>	filename or pool name of the image, must be unique among shared images
in	<i>img</i>	the image that is made available using the name

11.128.2.3 ~FI_Shared_Image()

```
Fl_Shared_Image::~~Fl_Shared_Image () [protected], [virtual]
```

The destructor frees all memory and server resources that are used by the image.

The destructor is protected and cannot be used directly from a program. Use the [Fl_Shared_Image::release\(\)](#) method instead.

11.128.3 Member Function Documentation

11.128.3.1 add()

```
void Fl_Shared_Image::add () [protected]
```

Adds a shared image to the image pool.

This **protected** method adds an image to the pool, an ordered list of shared images. The pool is searched for a matching image whenever one is requested, for instance with [Fl_Shared_Image::get\(\)](#) or [Fl_Shared_Image::find\(\)](#). This method does not increase or decrease reference counts!

11.128.3.2 add_handler()

```
void Fl_Shared_Image::add_handler (
    Fl_Shared_Handler f) [static]
```

Adds a shared image handler, which is basically a test function for adding new image formats.

This function will be called when an [Fl_Shared_Image](#) is to be loaded (for instance with [Fl_Shared_Image::get\(\)](#)) and the image type is not known to FLTK.

All registered image handlers will be called in the order of registration. You should always call [fl_register_images\(\)](#) before adding your own handlers - unless you need to override a known image file type which should be rare.

See also

[Fl_Shared_Handler](#) for more information of the function you need to define.

11.128.3.3 as_shared_image()

```
Fl_Shared_Image * Fl_Shared_Image::as_shared_image () [inline], [virtual]
```

Returns whether an image is an [Fl_Shared_Image](#) or not.

This virtual method returns a pointer to an [Fl_Shared_Image](#) if this object is an instance of [Fl_Shared_Image](#) or NULL if not. This can be used to detect if a given [Fl_Image](#) object is a shared image, i.e. derived from [Fl_Shared_Image](#).

Since

1.4.0

Reimplemented from [Fl_Image](#).

11.128.3.4 color_average()

```
void Fl_Shared_Image::color_average (
    Fl_Color c,
    float i) [virtual]
```

Averages the colors in the image with the provided FLTK color value. This method changes the pixel data of this specific image.

Note

It does not change any of the resized copies of this image, nor does it necessarily apply the color changes if this image is resized later.

Parameters

in	<i>c</i>	blend with this color
in	<i>i</i>	blend fraction

See also

[Fl_Image::color_average\(Fl_Color c, float i\)](#)

Reimplemented from [Fl_Image](#).

11.128.3.5 compare()

```
int Fl_Shared_Image::compare (
    Fl_Shared_Image ** i0,
    Fl_Shared_Image ** i1) [static], [protected]
```

Compares two shared images.

The order of comparison is:

1. Image name, usually the filename used to load it
2. Image width
3. Image height

Binary search in a sorted array works only if we search for the same parameters that were also used for sorting. No special cases are possible here.

[Fl_Shared_Image::find\(\)](#) requires a search for an element with a matching name and the original_ flags set. This is not implemented via binary search, but by a simple run of the array inside [Fl_Shared_Image::find\(\)](#).

Parameters

in	<i>i0,i1</i>	image pointer pointer for sorting

Returns

Whether the images match or their relative sort order (see text).

Return values

0	the images match
<0	Image <i>i0</i> is <i>less</i> than image <i>i1</i>
>0	Image <i>i0</i> is <i>greater</i> than image <i>i1</i>

11.128.3.6 copy() [1/2]

```
Fl_Image * Fl_Shared_Image::copy ()
```

Increments the reference counter and returns a pointer to itself.

When the image is no longer used, call `Fl_Shared_Image::release()`.

To get a copy of the image data, call `this->image()->copy()` instead.

Returns

pointer to an `Fl_Shared_Image` that can be safely cast

11.128.3.7 copy() [2/2]

```
Fl_Image * Fl_Shared_Image::copy (
    int W,
    int H) const [virtual]
```

Return a shared image of this image with the requested size.

This is the same as calling `Fl_Shared_Image::get(this->name(), W, H)`.

If a shared image of the desired size already exists in the shared image pool, the existing image is returned and no copy is made. But the reference counter is incremented. When the image is no longer used, call `Fl_Shared_Image::release()`.

To get a copy of the image data, call `this->image()->copy(W, H)` instead.

Parameters

in	<i>W,H</i>	size of requested image

Returns

pointer to an `Fl_Shared_Image` that can be safely cast, or NULL if the image can't be found and can't be created.

Reimplemented from `Fl_Image`.

11.128.3.8 copy_()

```
Fl_Shared_Image * Fl_Shared_Image::copy_ (
    int W,
    int H) const [protected]
```

Create a resized copy of the image and wrap it into the share image class.

This function is usually followed by a call to `returned_image->add()` to add the image to the pool, and `this->refcounter_++` to make sure that the original shared image keeps a reference to the copy. Don't call this function if an image of the given size is already in the pool.

Parameters

in	<i>W,H</i>	new image size

Returns

a new shared image pointer that is not yet in the pool

11.128.3.9 desaturate()

```
void Fl_Shared_Image::desaturate () [virtual]
```

Convert the image to gray scale.

This method changes the pixel data of this specific image.

Note

It does not change any of the resized copies of this image, nor does it necessarily apply the color changes if this image is resized later.

See also

[Fl_Image::desaturate\(\)](#)

Reimplemented from [Fl_Image](#).

11.128.3.10 draw()

```
void Fl_Shared_Image::draw (
    int X,
    int Y,
    int W,
    int H,
    int cx = 0,
    int cy = 0) [virtual]
```

Draw this image to the current graphics context.

Parameters

in	<i>X,Y,W,H</i>	draw at this position and size
in	<i>cx,cy</i>	image origin

Reimplemented from [Fl_Image](#).

11.128.3.11 find()

```
Fl_Shared_Image * Fl_Shared_Image::find (
    const char * name,
    int W = 0,
    int H = 0) [static]
```

Finds a shared image from its name and size specifications.

This uses a binary search in the image cache.

If the image `name` exists with the exact width `W` and height `H`, then it is returned.

If `W == 0` and the image `name` exists with another size, then the **original** image with that `name` is returned.

In either case the refcount of the returned image is increased. The found image should be released with [Fl_Shared_Image::release\(\)](#) when no longer needed.

An image is marked `original` if it was directly loaded from a file or from memory as opposed to copied and resized images.

This comparison is used in [Fl_Shared_Image::find\(\)](#) to find an image that matches the requested one or to find the position where a new image should be entered into the sorted list of shared images.

It is used in two steps by [Fl_Shared_Image::add\(\)](#):

1. search with exact width and height
2. if not found, search again with width = 0 (and height = 0)

The first step will only return a match if the image exists with the same width and height. The second step will match if there is an image marked `original` with the same name, regardless of width and height.

11.128.3.12 get() [1/2]

```
Fl_Shared_Image * Fl_Shared_Image::get (
    const char * name,
    int W = 0,
    int H = 0) [static]
```

Find or load an image that can be shared by multiple widgets.

If the image exists with the requested size, this image will be returned.

If the image exists, but only with another size, then a new copy with the requested size (width *W* and height *H*) will be created as a resized copy of the original image. The new image is added to the internal list of shared images.

If the image does not yet exist, then a new image of the proper dimension is created from the filename *name*. The original image from filename *name* is always added to the list of shared images in its original size. If the requested size differs, then the resized copy with width *W* and height *H* is also added to the list of shared images.

Note

If the sizes differ, then *two* images are created as mentioned above. This is intentional so the original image is cached and preserved. If you request the same image with another size later, then the **original** image will be found, copied, resized, and returned.

Shared JPEG and PNG images can also be created from memory by using their named memory access constructor. You should [release\(\)](#) the image when you're done with it.

Parameters

<i>name</i>	name of the image
<i>W,H</i>	desired size

Returns

the image at the requested size, or NULL if the image could not be found or generated

See also

[FI_Shared_Image::find\(const char *name, int W, int H\)](#)

[FI_Shared_Image::release\(\)](#)

[FI_JPEG_Image::FI_JPEG_Image\(const char *name, const unsigned char *data\)](#)

[FI_PNG_Image::FI_PNG_Image](#) (const char *name_png, const unsigned char *buffer, int maxsize)

11.128.3.13 get() [2/2]

```
FI_Shared_Image * FI_Shared_Image::get (
    FI_RGB_Image * rgb,
    int own_it = 1) [static]
```

Builds a shared image from a pre-existing [FI_RGB_Image](#).

Parameters

in	<i>rgb</i>	an FI_RGB_Image used to build a new shared image.
in	<i>own_it</i>	1 if the shared image should delete <i>rgb</i> when it is itself deleted, 0 otherwise

Version

1.3.4

11.128.3.14 image()

```
const FI_Image * FI_Shared_Image::image () const [inline]
```

Returns a pointer to the internal [FI_Image](#) object.

The output is a pointer to the *internal* image ('[FI_Image](#)' or subclass) which can be used to inspect or copy the image.

Do not try to modify the image! You can copy the image though if you want or need to change any attributes, size etc. If all you need to do is to resize the image you should use [FI_Shared_Image::copy\(int, int\)](#) instead.

Note

The internal image (pointer) is protected for good reasons, e.g. to prevent access to the image so it can't be modified by user code. **DO NOT** cast away the 'const' attribute to modify the image.

User code should rarely need this method. Use with caution.

Returns

const `Fl_Image*` image, the internal [Fl_Image](#)

Since

1.4.0

11.128.3.15 images()

```
Fl_Shared_Image** Fl_Shared_Image::images () [static]
```

Returns the `Fl_Shared_Image*` array.

Returns

a pointer to an array of shared image pointers, sorted by name and size

See also

[Fl_Shared_Image::num_images\(\)](#)

11.128.3.16 num_images()

```
int Fl_Shared_Image::num_images () [static]
```

Number of shared images in their various cached sizes.

Returns

number of entries in the array

See also

[Fl_Shared_Image::images\(\)](#)

11.128.3.17 original()

```
int Fl_Shared_Image::original () [inline]
```

Returns whether this is an original image.

Images loaded from a file or from memory are marked `original` as opposed to images created as a copy of another image with different size (width or height).

Note

This is useful for debugging (rarely used in user code).

Since

FLTK 1.4.0

11.128.3.18 refcount()

```
int Fl_Shared_Image::refcount () [inline]
```

Returns the number of references of this shared image.

When reference is below 1, the image is deleted.

11.128.3.19 `release()`

```
void Fl_Shared_Image::release () [virtual]
```

Releases and possibly destroys (if refcount ≤ 0) a shared image.

In the latter case, it will reorganize the shared image array so that no hole will occur.

Reimplemented from [Fl_Image](#).

11.128.3.20 `uncache()`

```
void Fl_Shared_Image::uncache () [virtual]
```

Remove the cached device specific image data.

See also

[Fl_Image::uncache\(\)](#)

Reimplemented from [Fl_Image](#).

11.128.3.21 `update()`

```
void Fl_Shared_Image::update () [protected]
```

Update the dimensions of the shared images.

Internal method to synchronize shared image data with the actual image data.

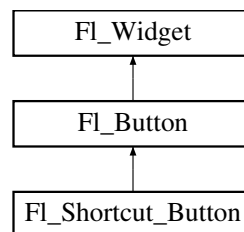
The documentation for this class was generated from the following files:

- [Fl_Shared_Image.H](#)
- [Fl_Shared_Image.cxx](#)

11.129 `Fl_Shortcut_Button` Class Reference

A button that allows the user to type a key combination to create shortcuts.

Inheritance diagram for `Fl_Shortcut_Button`:

**Public Member Functions**

- [Fl_Shortcut_Button](#) (int X, int Y, int W, int H, const char *l=0)
Construct a shortcut button.
- [Fl_Shortcut value](#) ()
Return the user selected shortcut.
- void [value](#) ([Fl_Shortcut shortcut](#))
Set the displayed shortcut.

Public Member Functions inherited from [Fl_Button](#)

- int [clear](#) ()
Same as `value(0)`.
- [uchar compact](#) ()
Return true if buttons are rendered as compact buttons.
- void [compact](#) ([uchar v](#))

- Decide if buttons should be rendered in compact mode.*
- [Fl_Boxtype](#) **down_box** () const
Returns the current down box type, which is drawn when [value\(\)](#) is non-zero.
- void **down_box** ([Fl_Boxtype](#) b)
Sets the down box type.
- [Fl_Color](#) **down_color** () const
(for backwards compatibility)
- void **down_color** (unsigned c)
(for backwards compatibility)
- [Fl_Button](#) (int X, int Y, int W, int H, const char *L=0)
The constructor creates the button using the given position, size, and label.
- int **set** ()
Same as [value\(1\)](#).
- void **setonly** ()
Turns on this button and turns off all other radio buttons in the group (calling [value\(1\)](#) or [set\(\)](#) does not do this).
- int **shortcut** () const
Returns the current shortcut key for the button.
- void **shortcut** (const char *s)
(for backwards compatibility)
- void **shortcut** (int s)
Sets the shortcut key to s.
- char **value** () const
Returns the current value of the button (0 or 1).
- int **value** (int v)
Sets the current value of the button.

Public Member Functions inherited from [Fl_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- [Fl_Align](#) **align** () const
Gets the label alignment.
- void **align** ([Fl_Align](#) alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * **as_gl_window** ()
Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * **as_gl_window** () const
- virtual [Fl_Group](#) * **as_group** ()
Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).
- virtual [Fl_Group](#) const * **as_group** () const
- virtual [Fl_Window](#) * **as_window** ()

- Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).*
- virtual [FL_Window](#) const * **as_window** () const
- void [bind_deimage](#) ([FL_Image](#) *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void [bind_deimage](#) (int f)
- Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- void [bind_image](#) ([FL_Image](#) *img)
- Sets the image to use as part of the widget label when in the active state.*
- void [bind_image](#) (int f)
- Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- [FL_Boxtype](#) **box** () const
- Gets the box type of the widget.*
- void [box](#) ([FL_Boxtype](#) new_box)
- Sets the box type for the widget.*
- [FL_Callback_p](#) **callback** () const
- Gets the current callback function for the widget.*
- void [callback](#) ([FL_Callback](#) *cb)
- Sets the current callback function for the widget.*
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*
- void [callback](#) ([FL_Callback](#) *cb, void *p)
- Sets the current callback function and data for the widget.*
- void [callback](#) ([FL_Callback0](#) *cb)
- Sets the current callback function for the widget.*
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)
- Sets the current callback function for the widget.*
- unsigned int [changed](#) () const
- Checks if the widget value changed since the last callback.*
- void [clear_active](#) ()
- Marks the widget as inactive without sending events or changing focus.*
- void [clear_changed](#) ()
- Marks the value of the widget as unchanged.*
- void [clear_damage](#) (uchar c=0)
- Clears or sets the damage flags.*
- void [clear_output](#) ()
- Sets a widget to accept input.*
- void [clear_visible](#) ()
- Hides the widget.*
- void [clear_visible_focus](#) ()
- Disables keyboard focus navigation with this widget.*
- [FL_Color](#) **color** () const
- Gets the background color of the widget.*
- void [color](#) ([FL_Color](#) bg)
- Sets the background color of the widget.*
- void [color](#) ([FL_Color](#) bg, [FL_Color](#) sel)
- Sets the background and selection color of the widget.*
- [FL_Color](#) **color2** () const
- For back compatibility only.*
- void [color2](#) (unsigned a)
- For back compatibility only.*
- int [contains](#) (const [FL_Widget](#) *w) const

- Checks if w is a child of this widget.*

 - void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.
- void `do_callback` (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with arbitrary arguments.
- void `do_callback` (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, FL_Align) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const

Gets the widget height.
- virtual void `hide` ()

Makes a widget invisible.
- int `horizontal_label_margin` ()

Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)

Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image` ()

Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image * image` () const

Gets the image that is used as part of the widget label when in the active state.
- void `image` (FL_Image &img)

Sets the image to use as part of the widget label when in the active state.
- void `image` (FL_Image *img)

Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const

Returns whether the image is managed by the widget.

- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)
Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

- Gets the selection color.*

 - void **selection_color** (**Fl_Color** a)

Sets the selection color.
- void **set_active** ()

Marks the widget as active without sending events or changing focus.
- void **set_changed** ()

Marks the value of the widget as changed.
- void **set_output** ()

Sets a widget to output only.
- void **set_visible** ()

Makes the widget visible.
- void **set_visible_focus** ()

Enables keyboard focus navigation with this widget.
- int **shortcut_label** () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void **shortcut_label** (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void **show** ()

Makes a widget visible.
- void **size** (int W, int H)

Changes the size of the widget.
- int **take_focus** ()

Gives the widget the keyboard focus.
- unsigned int **takeevents** () const

Returns if the widget is able to take events.
- int **test_shortcut** ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * **tooltip** () const

Gets the current tooltip text.
- void **tooltip** (const char *text)

Sets the current tooltip text.
- **Fl_Window** * **top_window** () const

Returns a pointer to the top-level window for the widget.
- **Fl_Window** * **top_window_offset** (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- **uchar** **type** () const

Gets the widget type.
- void **type** (**uchar** t)

Sets the widget type.
- int **use_accents_menu** ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const

Gets the user data for this widget.
- void **user_data** (**Fl_Callback_User_Data** *v, bool auto_free)

Sets the user data for this widget.
- void **user_data** (void *v)

Sets the user data for this widget.
- int **vertical_label_margin** ()

Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)

Set the spacing between the label and the vertical edge of the widget.

- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **FL_When** **when** () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **FL_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~FL_Widget** ()
Destroys the widget.

Protected Member Functions

- void **do_end_hot_callback** ()
Call the callback if the user is interested.
- void **draw** () **FL_OVERRIDE**
Draw the textual representation of the shortcut button.
- int **handle** (int) **FL_OVERRIDE**
Handle keystrokes to catch the user's shortcut.

Protected Member Functions inherited from **FL_Button**

- void **simulate_key_action** ()

Protected Member Functions inherited from **FL_Widget**

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (**FL_Boxtype** t, **FL_Color** c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (**FL_Boxtype** t, int x, int y, int w, int h, **FL_Color** c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (**FL_Boxtype** t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.

- void `draw_focus` (`FI_Boxtype` t, int x, int y, int w, int h, `FI_Color` bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- `FI_Widget` (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Protected Attributes

- `FI_Shortcut` shortcut_value

Additional Inherited Members

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `FI_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Static Protected Member Functions inherited from [Fl_Button](#)

- static void `key_release_timeout` (void *)

Static Protected Attributes inherited from [Fl_Button](#)

- static [Fl_Widget_Tracker](#) * `key_release_tracker` = 0

11.129.1 Detailed Description

A button that allows the user to type a key combination to create shortcuts.

After clicked once, the button catches the following keyboard events and records the pressed keys and all modifiers. It draws a text representation of the shortcut.

The backspace key deletes the current shortcut. A second click on the button or moving focus makes the last shortcut permanent.

The Shortcut button calls the user callback after every change if `FL_WHEN_CHANGED` is set, and when the button is no longer recording shortcuts if `FL_WHEN_RELEASE` is set.

11.129.2 Constructor & Destructor Documentation

11.129.2.1 `Fl_Shortcut_Button()`

```
Fl_Shortcut_Button::Fl_Shortcut_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Construct a shortcut button.

Parameters

<i>X,Y,W,H</i>	position and size of the button
<i>l</i>	label text when no shortcut is set

11.129.3 Member Function Documentation

11.129.3.1 `draw()`

```
void Fl_Shortcut_Button::draw (
    void ) [protected], [virtual]
```

Draw the textual representation of the shortcut button.

When the button can receive shortcut key events, it's "hot". A hot button is drawn in selection color. A cold button is drawn as a regular text box containing a human readable version of the shortcut key.

Reimplemented from [Fl_Button](#).

11.129.3.2 `handle()`

```
int Fl_Shortcut_Button::handle (
    int e) [protected], [virtual]
```

Handle keystrokes to catch the user's shortcut.

Reimplemented from [Fl_Button](#).

11.129.3.3 `value()` [1/2]

```
Fl\_Shortcut Fl_Shortcut_Button::value ()
```

Return the user selected shortcut.

Returns

shortcut encoded as key and modifier

11.129.3.4 value() [2/2]

```
void Fl_Shortcut_Button::value (
    Fl_Shortcut shortcut)
```

Set the displayed shortcut.

Parameters

in	shortcut	encoded as key and modifier

The documentation for this class was generated from the following files:

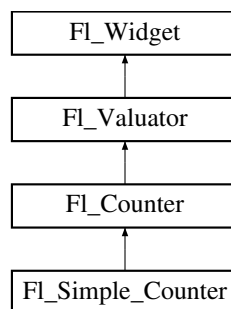
- Fl_Shortcut_Button.H
- Fl_Shortcut_Button.cxx

11.130 Fl_Simple_Counter Class Reference

This widget creates a counter with only 2 arrow buttons.

```
#include <Fl_Simple_Counter.H>
```

Inheritance diagram for Fl_Simple_Counter:

**Public Member Functions**

- **Fl_Simple_Counter** (int X, int Y, int W, int H, const char *L=0)

Public Member Functions inherited from Fl_Counter

- **Fl_Counter** (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Counter](#) widget using the given position, size, and label string.
- int **handle** (int) **FL_OVERRIDE**
Handles the specified event.
- void **lstep** (double a)
Sets the increment for the large step buttons.
- double **step** () const
Returns the increment for normal step buttons.
- void **step** (double a)
Sets the increment for the normal step buttons.
- void **step** (double a, double b)
Sets the increments for the normal and large step buttons.
- **Fl_Color** **textcolor** () const

- Gets the font color.*
- void **textcolor** (FI_Color s)
Sets the font color to s.
- FI_Font **textfont** () const
Gets the text font.
- void **textfont** (FI_Font s)
Sets the text font to s.
- FI_Fontsize **textsize** () const
Gets the font size.
- void **textsize** (FI_Fontsize s)
Sets the font size to s.
- ~FI_Counter ()
Destroys the valuator.

Public Member Functions inherited from FI_Valuator

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)
Sets the maximum value for the valuator.
- double **minimum** () const
Gets the minimum value for the valuator.
- void **minimum** (double a)
Sets the minimum value for the valuator.
- void **precision** (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void **range** (double a, double b)
Sets the minimum and maximum values for the valuator.
- double **round** (double)
Round the passed value to the nearest step increment.
- double **step** () const
Gets or sets the step value.
- void **step** (double a, int b)
See double FI_Valuator::step() const
- void **step** (double s)
See double FI_Valuator::step() const.
- void **step** (int a)
See double FI_Valuator::step() const
- double **value** () const
Gets the floating point(double) value.
- int **value** (double)
Sets the current value.
- ~FI_Valuator () **FL_OVERRIDE**
Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color color` () const

Gets the background color of the widget.
- void `color` (FL_Color bg)

Sets the background color of the widget.
- void `color` (FL_Color bg, FL_Color sel)

Sets the background and selection color of the widget.
- `FL_Color color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const FL_Widget *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([FL_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void `default_callback` (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from FI_Counter

- void `arrow_widths` (int &w1, int &w2)
Compute sizes (widths) of arrow boxes.
- void `draw` () `FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from FI_Valuator

- `FI_Valuator` (int X, int Y, int W, int H, const char *L)
Creates a new FI_Valuator widget using the given position, size, and label string.
- void `handle_drag` (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void `handle_push` ()
Stores the current value in the previous value.
- void `handle_release` ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int `horizontal` () const
Tells if the valuator is an FL_HORIZONTAL one.
- double `previous_value` () const
Gets the previous floating point value before an event changed it.
- void `set_value` (double v)
Sets the current floating point value.
- double `softclamp` (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void `value_damage` ()
Asks for partial redraw.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.130.1 Detailed Description

This widget creates a counter with only 2 arrow buttons.

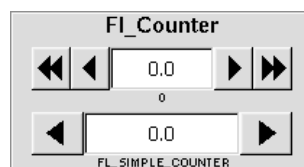


Figure 11.43 [FI_Simple_Counter](#)

The documentation for this class was generated from the following files:

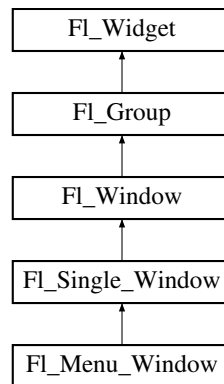
- [FI_Simple_Counter.H](#)
- [FI_Counter.cxx](#)

11.131 Fl_Single_Window Class Reference

This is the same as [Fl_Window](#).

```
#include <Fl_Single_Window.H>
```

Inheritance diagram for Fl_Single_Window:



Public Member Functions

- **Fl_Single_Window** (int W, int H, const char *l=0)
Creates a new [Fl_Single_Window](#) widget using the given size, and label (title) string.
- **Fl_Single_Window** (int X, int Y, int W, int H, const char *l=0)
Creates a new [Fl_Single_Window](#) widget using the given position, size, and label (title) string.
- void [flush](#) () **FL_OVERRIDE**
Same as [Fl_Window::flush\(\)](#)
- void **make_current** ()
Same as [Fl_Window::make_current\(\)](#)
- void [show](#) () **FL_OVERRIDE**
Makes a widget visible.
- void **show** (int argc, char **argv)
*Same as [Fl_Window::show\(int argc, char **argv\)](#)*

Public Member Functions inherited from [Fl_Window](#)

- void [allow_expand_outside_parent](#) ()
Allow this subwindow to expand outside the area of its parent window.
- virtual class [Fl_Double_Window](#) * [as_double_window](#) ()
Return non-null if this is an [Fl_Double_Window](#) object.
- virtual class [Fl_Overlay_Window](#) * [as_overlay_window](#) ()
Return non-null if this is an [Fl_Overlay_Window](#) object.
- [Fl_Window](#) const * [as_window](#) () const **FL_OVERRIDE**
- [Fl_Window](#) * [as_window](#) () **FL_OVERRIDE**
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- unsigned int **border** () const
Returns whether the window possesses a border.
- void [border](#) (int b)
Sets whether or not the window manager border is around the window.
- void [clear_border](#) ()
Fast inline function to turn the window manager border off.
- void [clear_modal_states](#) ()
Clears the "modal" flags and converts a "modal" or "non-modal" window back into a "normal" window.

- void **copy_label** (const char *a)
Sets the window titlebar label to a copy of a character string.
- void **cursor** (const [FL_RGB_Image](#) *, int, int)
Changes the cursor for this window using the provided image as cursor's shape.
- void **cursor** ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void **cursor** ([FL_Cursor](#))
Changes the cursor for this window.
- int **decorated_h** () const
Returns the window height including any window title bar and any frame added by the window manager.
- int **decorated_w** () const
Returns the window width including any frame added by the window manager.
- void **default_cursor** ([FL_Cursor](#) c, [FL_Color](#), [FL_Color](#)=[FL_WHITE](#))
For back compatibility only.
- void **default_cursor** ([FL_Cursor](#))
Sets the default window cursor.
- void **draw_backdrop** ()
Draw the background image if one is set and is aligned inside.
- [FL_Window](#) (int w, int h, const char *title=0)
Creates a window from the given width w, height h, and title.
- [FL_Window](#) (int x, int y, int w, int h, const char *title=0)
Creates a window from the given position (x, y), size (w, h) and title.
- void **free_position** ()
Undoes the effect of a previous [resize\(\)](#) or [show\(\)](#) so that the next time [show\(\)](#) is called the window manager is free to position the window.
- void **fullscreen** ()
Makes the window completely fill one or more screens, without any window manager border visible.
- unsigned int **fullscreen_active** () const
Returns non zero if [FULLSCREEN](#) flag is set, 0 otherwise.
- void **fullscreen_off** ()
Turns off any side effects of [fullscreen\(\)](#)
- void **fullscreen_off** (int X, int Y, int W, int H)
Turns off any side effects of [fullscreen\(\)](#) and does [resize\(x,y,w,h\)](#).
- void **fullscreen_screens** (int top, int bottom, int left, int right)
Sets which screens should be used when this window is in fullscreen mode.
- [uchar](#) **get_size_range** (int *minw, int *minh, int *maxw=NULL, int *maxh=NULL, int *dw=NULL, int *dh=NULL, int *aspect=NULL)
Gets the allowable range to which the user can resize this window.
- int **handle** (int) [FL_OVERRIDE](#)
Handles the specified event.
- void **hide** () [FL_OVERRIDE](#)
Removes the window from the screen.
- void **hotspot** (const [FL_Widget](#) &p, int offscreen=0)
See void [FL_Window::hotspot](#)(int x, int y, int offscreen = 0)
- void **hotspot** (const [FL_Widget](#) *, int offscreen=0)
See void [FL_Window::hotspot](#)(int x, int y, int offscreen = 0)
- void **hotspot** (int x, int y, int offscreen=0)
Positions the window so that the mouse is pointing at the given position, or at the center of the given widget, which may be the window itself.
- const void * **icon** () const
Gets the current icon window target dependent data.

- void [icon](#) (const [FL_RGB_Image](#) *)
Sets or resets a single window icon.
- void [icon](#) (const void *ic)
Platform-specific method to set the window icon usable on Windows and X11 only.
- void [iconize](#) ()
Iconifies the window.
- const char * [iconlabel](#) () const
See void [FL_Window::iconlabel\(const char\)](#)*
- void [iconlabel](#) (const char *)
Sets the icon label.
- void [icons](#) (const [FL_RGB_Image](#) *[], int)
Sets the window icons.
- void [icons](#) (HICON big_icon, HICON small_icon)
Sets the window icons using HICON handles (Windows platform only).
- const char * [label](#) () const
See void [FL_Window::label\(const char\)](#)*
- void [label](#) (const char *)
Sets the window title bar label.
- void [label](#) (const char *label, const char *[iconlabel](#))
Sets the icon label.
- void [make_current](#) ()
Sets things up so that the drawing functions in <[FL/fl_draw.H](#)> will go into this window.
- void [maximize](#) ()
Maximizes a top-level window to its current screen.
- unsigned int [maximize_active](#) () const
Returns whether the window is currently maximized.
- unsigned int [menu_window](#) () const
Returns true if this window is a menu window.
- unsigned int [modal](#) () const
Returns true if this window is modal.
- unsigned int [non_modal](#) () const
Returns true if this window is modal or non-modal.
- [fl_uintptr_t](#) [os_id](#) ()
Returns a platform-specific identification of a shown window, or 0 if not shown.
- unsigned int [override](#) () const
Returns non zero if `OVERRIDE` flag is set, 0 otherwise.
- void [resize](#) (int X, int Y, int W, int H) [FL_OVERRIDE](#)
Changes the size and position of the window.
- int [screen_num](#) ()
The number of the screen containing the mapped window.
- void [screen_num](#) (int screen_num)
Set the number of the screen where to map the window.
- void [set_menu_window](#) ()
Marks the window as a menu window.
- void [set_modal](#) ()
A "modal" window, when [shown\(\)](#), will prevent any events from being delivered to other windows in the same program, and will also remain on top of the other windows (if the X window manager supports the "transient for" property).
- void [set_non_modal](#) ()

A "non-modal" window (terminology borrowed from Microsoft Windows) acts like a [modal\(\)](#) one in that it remains on top, but it has no effect on event delivery.

- void **set_override** ()
Activates the flags NOBORDER|OVERRIDE.
- void **set_tooltip_window** ()
Marks the window as a tooltip window.
- const [FL_Image](#) * **shape** ()
Returns the image controlling the window shape or NULL.
- void **shape** (const [FL_Image](#) &b)
Set the window's shape with an [FL_Image](#).
- void **shape** (const [FL_Image](#) *img)
Assigns a non-rectangular shape to the window.
- void **show** () [FL_OVERRIDE](#)
Puts the window on the screen.
- void **show** (int argc, char **argv)
Puts the window on the screen with [show\(\)](#) and parses command-line arguments.
- int **shown** ()
Returns non-zero if [show\(\)](#) has been called (but not [hide\(\)](#)).
- void **size_range** (int minw, int minh, int maxw=0, int maxh=0, int dw=0, int dh=0, int aspect=0)
Sets the allowable range to which the user can resize this window.
- unsigned int **tooltip_window** () const
Returns true if this window is a tooltip window.
- void **un_maximize** ()
Returns a previously maximized top-level window to its previous size.
- void **wait_for_expose** ()
Waits for the window to be displayed after calling [show\(\)](#).
- int **x_root** () const
Gets the x position of the window on the screen.
- const char * **xclass** () const
Returns the xclass for this window, or a default.
- void **xclass** (const char *c)
Sets the xclass for this window.
- int **y_root** () const
Gets the y position of the window on the screen.
- virtual [~FL_Window](#) ()
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *& **_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void **add_resizable** ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * **array** () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * **as_group** () const [FL_OVERRIDE](#)
- [FL_Group](#) * **as_group** () [FL_OVERRIDE](#)

- Returns an *FL_Group* pointer if this widget is an *FL_Group*.

 - void **begin** ()

Sets the current group so you can build the widget tree by just constructing the widgets.
- *FL_Widget* * **child** (int n) const

Returns the *n*'th child.
- int **children** () const

Returns how many child widgets the group has.
- void **clear** ()

Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()

Returns the current clipping mode.
- void **clip_children** (int c)

Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)

Removes the widget at *index* from the group and deletes it.
- void **end** ()

Exactly the same as `current(this->parent())`.
- int **find** (const *FL_Widget* &o) const

See `int FL_Group::find(const FL_Widget *w) const`.
- int **find** (const *FL_Widget* *) const

Searches the children for the widget and returns the index.
- *FL_Group* (int, int, int, int, const char *s)

Creates a new *FL_Group* widget using the given position, size, and label string.
- void **focus** (*FL_Widget* *w)
- void **forms_end** ()

This is for forms compatibility only.
- void **init_sizes** ()

Resets the internal array of widget sizes and positions.
- void **insert** (*FL_Widget* &, int i)

The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** (*FL_Widget* &o, *FL_Widget* *before)

This does `insert(w, find(before))`.
- void **remove** (*FL_Widget* &)

Removes a widget from the group but does not delete it.
- void **remove** (*FL_Widget* *o)

Removes the widget *o* from the group.
- void **remove** (int index)

Removes the widget at *index* from the group but does not delete it.
- *FL_Widget* * **resizable** () const

Returns the group's resizable widget.
- void **resizable** (*FL_Widget* &o)

Sets the group's resizable widget.
- void **resizable** (*FL_Widget* *o)

The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual ~*FL_Group* ()

The destructor also deletes all the children.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.

- void `clear_damage` (`uchar` c=0)
Clears or sets the damage flags.
- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FL_Color` bg)
Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)
Sets the background and selection color of the widget.
- `FL_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FL_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int **`damage_resize`** (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FL_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const

- Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- `int h () const`
Gets the widget height.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FL_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FL_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FL_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FL_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FL_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FL_Color c)`
Sets the label color.
- `FL_Font labelfont () const`
Gets the font to use.
- `void labelfont (FL_Font f)`
Sets the font to use.
- `FL_Fonsize labelsize () const`
Gets the font size in pixels.
- `void labelsize (FL_Fonsize pix)`
Sets the font size in pixels.
- `FL_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (FL_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`
Sets width ww and height hh accordingly with the label size.
- `bool needs_keyboard () const`
Returns whether this widget needs a keyboard.

- void `needs_keyboard` (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int `output` () const
Returns if a widget is used for output only.
- `FL_Group * parent` () const
Returns a pointer to the parent widget.
- void `parent` (`FL_Group *p`)
Internal use only - "for hacks only".
- void `position` (int X, int Y)
Repositions the window or widget.
- void `redraw` ()
Schedules the drawing of the widget.
- void `redraw_label` ()
Schedules the drawing of the label.
- `FL_Color selection_color` () const
Gets the selection color.
- void `selection_color` (`FL_Color a`)
Sets the selection color.
- void `set_active` ()
Marks the widget as active without sending events or changing focus.
- void `set_changed` ()
Marks the value of the widget as changed.
- void `set_output` ()
Sets a widget to output only.
- void `set_visible` ()
Makes the widget visible.
- void `set_visible_focus` ()
Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- void `size` (int W, int H)
Changes the size of the widget.
- int `take_focus` ()
Gives the widget the keyboard focus.
- unsigned int `takeevents` () const
Returns if the widget is able to take events.
- int `test_shortcut` ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const
Gets the current tooltip text.
- void `tooltip` (const char *text)
Sets the current tooltip text.
- `FL_Window * top_window` () const
Returns a pointer to the top-level window for the widget.
- `FL_Window * top_window_offset` (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type` () const
Gets the widget type.
- void `type` (`uchar t`)

- Sets the widget type.*
- int **use_accents_menu** ()
 - Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*
- void * **user_data** () const
 - Gets the user data for this widget.*
- void **user_data** (FI_Callback_User_Data *v, bool auto_free)
 - Sets the user data for this widget.*
- void **user_data** (void *v)
 - Sets the user data for this widget.*
- int **vertical_label_margin** ()
 - Get the spacing between the label and the vertical edge of the widget.*
- void **vertical_label_margin** (int px)
 - Set the spacing between the label and the vertical edge of the widget.*
- unsigned int **visible** () const
 - Returns whether a widget is visible.*
- unsigned int **visible_focus** () const
 - Checks whether this widget has a visible focus.*
- void **visible_focus** (int v)
 - Modifies keyboard focus navigation.*
- int **visible_r** () const
 - Returns whether a widget and all its parents are visible.*
- int **w** () const
 - Gets the widget width.*
- FI_When **when** () const
 - Returns the conditions under which the callback is called.*
- void **when** (uchar i)
 - Sets the flags used to decide when a callback is called.*
- FI_Window * **window** () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
- int **x** () const
 - Gets the widget position in its window.*
- int **y** () const
 - Gets the widget position in its window.*
- virtual ~FI_Widget ()
 - Destroys the widget.*

Additional Inherited Members

Public Types inherited from FI_Window

- typedef struct HICON__ * HICON

Static Public Member Functions inherited from FI_Window

- static FI_Window * **current** ()
 - Returns the last window that was made current.*
- static void **default_callback** (FI_Window *, void *v)
 - Back compatibility: Sets the default callback v for win to call on close event.*
- static void **default_icon** (const FI_RGB_Image *)
 - Sets a single default window icon.*
- static void **default_icons** (const FI_RGB_Image *[], int)
 - Sets the default window icons.*

- static void [default_icons](#) (HICON big_icon, HICON small_icon)
Sets the default window icons (Windows platform only).
- static const char * [default_xclass](#) ()
Returns the default xclass.
- static void [default_xclass](#) (const char *)
Sets the default window xclass.
- static bool [is_a_rescale](#) ()
Returns true when a window is being rescaled.
- static char [show_next_window_iconic](#) ()
Returns the static flag whether the next window should be opened iconified.
- static void [show_next_window_iconic](#) (char stat)
Sets a static flag whether the next window should be opened iconified.

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * [current](#) ()
Returns the currently active group.
- static void [current](#) ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [FI_Widget](#)

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from [FI_Window](#)

- void [default_size_range](#) ()
Protected method to calculate the default size range of a window.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- int [force_position](#) () const

- Returns the internal state of the window's FORCE_POSITION flag.*
- void [force_position](#) (int force)
Sets an internal flag that tells FLTK and the window manager to honor position requests.
- void [free_icons](#) ()
Deletes all icons previously attached to the window.
- int [is_resizable](#) ()
Protected method to determine whether a window is resizable.

Protected Member Functions inherited from [Fl_Group](#)

- [Fl_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw_child](#) ([Fl_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [Fl_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) ([Fl_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int [on_move](#) (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void [on_remove](#) (int)
Allow derived groups to act when a child widget is removed from the group.
- int * [sizes](#) ()
Returns the internal array of widget sizes and positions.
- void [update_child](#) ([Fl_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [Fl_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([Fl_Boxtype](#) t, [Fl_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([Fl_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.
- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [Fl_Widget](#) (int x, int y, int w, int h, const char *label=0L)

- Creates a widget at the given position and size.*

 - unsigned int **flags** () const

Gets the widget flags mask.
- void **h** (int v)

Internal use only.
- void **set_flag** (unsigned int c)

Sets a flag in the flags mask.
- void **w** (int v)

Internal use only.
- void **x** (int v)

Internal use only.
- void **y** (int v)

Internal use only.

Static Protected Attributes inherited from `Fl_Window`

- static `Fl_Window * current_`
- Stores the last window that was made current.*

11.131.1 Detailed Description

This is the same as `Fl_Window`.

However, it is possible that some implementations will provide double-buffered windows by default. This subclass can be used to force single-buffering. This may be useful for modifying existing programs that use incremental update, or for some types of image data, such as a movie flipbook.

11.131.2 Member Function Documentation

11.131.2.1 `flush()`

```
void Fl_Single_Window::flush () [inline], [virtual]
```

Same as `Fl_Window::flush()`

Reimplemented from `Fl_Window`.

11.131.2.2 `show()`

```
void Fl_Single_Window::show () [virtual]
```

Makes a widget visible.

An invisible widget never gets redrawn and does not get keyboard or mouse events, but can receive a few other events like `FL_SHOW`.

The `visible()` method returns true if the widget is set to be visible. The `visible_r()` method returns true if the widget and all of its parents are visible. A widget is only visible if `visible()` is true on it *and all of its parents*.

Changing it will send `FL_SHOW` or `FL_HIDE` events to the widget. *Do not change it if the parent is not visible, as this will send false `FL_SHOW` or `FL_HIDE` events to the widget.* `redraw()` is called if necessary on this or the parent.

See also

`hide()`, `visible()`, `visible_r()`

Reimplemented from `Fl_Widget`.

The documentation for this class was generated from the following files:

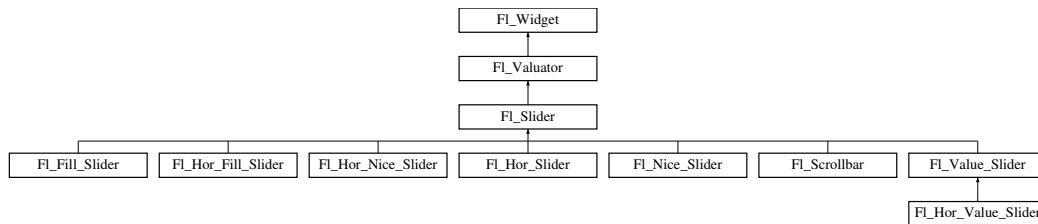
- `Fl_Single_Window.H`
- `Fl_Single_Window.cxx`

11.132 FL_Slider Class Reference

The [FL_Slider](#) widget contains a sliding knob inside a box.

```
#include <FL_Slider.H>
```

Inheritance diagram for [FL_Slider](#):



Public Member Functions

- void [bounds](#) (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- [FL_Slider](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [FL_Slider](#) widget using the given position, size, and label string.
- [FL_Slider](#) (uchar t, int X, int Y, int W, int H, const char *L)
Creates a new [FL_Slider](#) widget using the given type, position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- int [scrollvalue](#) (int pos, int [size](#), int first, int total)
Sets the size and position of the sliding knob in the box.
- [FL_Boxtype](#) [slider](#) () const
Gets the slider box type.
- void [slider](#) ([FL_Boxtype](#) c)
Sets the slider box type.
- float [slider_size](#) () const
Get the dimensions of the moving piece of slider.
- void [slider_size](#) (double v)
Set the dimensions of the moving piece of slider.

Public Member Functions inherited from [FL_Valuator](#)

- void [bounds](#) (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double [clamp](#) (double)
Clamps the passed value to the valuator range.
- virtual int [format](#) (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double [increment](#) (double, int)
Adds n times the step value to the passed value.
- double [maximum](#) () const
Gets the maximum value for the valuator.
- void [maximum](#) (double a)
Sets the maximum value for the valuator.
- double [minimum](#) () const
Gets the minimum value for the valuator.
- void [minimum](#) (double a)
Sets the minimum value for the valuator.

- void **precision** (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void **range** (double a, double b)
Sets the minimum and maximum values for the valuator.
- double **round** (double)
Round the passed value to the nearest step increment.
- double **step** () const
Gets or sets the step value.
- void **step** (double a, int b)
See double [FI_Valuator::step\(\)](#) const
- void **step** (double s)
See double [FI_Valuator::step\(\)](#) const.
- void **step** (int a)
See double [FI_Valuator::step\(\)](#) const
- double **value** () const
Gets the floating point(double) value.
- int **value** (double)
Sets the current value.
- **~FI_Valuator** () **FL_OVERRIDE**
Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (**FI_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FI_Gl_Window** * **as_gl_window** ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class **FI_Gl_Window** const * **as_gl_window** () const
- virtual **FI_Group** * **as_group** ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual **FI_Group** const * **as_group** () const
- virtual **FI_Window** * **as_window** ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual **FI_Window** const * **as_window** () const
- void **bind_deimage** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.

- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)

- Sets the current tooltip text.*
- `uchar damage () const`
 - Returns non-zero if `draw()` needs to be called.*
- `void damage (uchar c)`
 - Sets the damage bits for the widget.*
- `void damage (uchar c, int x, int y, int w, int h)`
 - Sets the damage bits for an area inside the widget.*
- `int damage_resize (int, int, int, int)`
 - Internal use only.*
- `void deactivate ()`
 - Deactivates the widget.*
- `FL_Image * deimage ()`
 - Gets the image that is used as part of the widget label when in the inactive state.*
- `const FL_Image * deimage () const`
 - Gets the image that is used as part of the widget label when in the inactive state.*
- `void deimage (FL_Image &img)`
 - Sets the image to use as part of the widget label when in the inactive state.*
- `void deimage (FL_Image *img)`
 - Sets the image to use as part of the widget label when in the inactive state.*
- `int deimage_bound () const`
 - Returns whether the inactive image is managed by the widget.*
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
 - Calls the widget callback function with default arguments.*
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
 - Calls the widget callback function with arbitrary arguments.*
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
 - Calls the widget callback function with arbitrary arguments.*
- `void draw_label (int, int, int, int, FL_Align) const`
 - Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- `int h () const`
 - Gets the widget height.*
- `virtual void hide ()`
 - Makes a widget invisible.*
- `int horizontal_label_margin ()`
 - Get the spacing between the label and the horizontal edge of the widget.*
- `void horizontal_label_margin (int px)`
 - Set the spacing between the label and the horizontal edge of the widget.*
- `FL_Image * image ()`
 - Gets the image that is used as part of the widget label when in the active state.*
- `const FL_Image * image () const`
 - Gets the image that is used as part of the widget label when in the active state.*
- `void image (FL_Image &img)`
 - Sets the image to use as part of the widget label when in the active state.*
- `void image (FL_Image *img)`
 - Sets the image to use as part of the widget label when in the active state.*
- `int image_bound () const`
 - Returns whether the image is managed by the widget.*
- `int inside (const FL_Widget *wgt) const`
 - Checks if this widget is a child of `wgt`.*
- `int is_label_copied () const`
 - Returns whether the current label was assigned with `copy_label()`.*

- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FI_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FI_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FI_Color c)`
Sets the label color.
- `FI_Font labelfont () const`
Gets the font to use.
- `void labelfont (FI_Font f)`
Sets the font to use.
- `FI_Fonsize labelsiz () const`
Gets the font size in pixels.
- `void labelsiz (FI_Fonsize pix)`
Sets the font size in pixels.
- `FI_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (FI_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`
Sets width ww and height hh accordingly with the label size.
- `bool needs_keyboard () const`
Returns whether this widget needs a keyboard.
- `void needs_keyboard (bool needs)`
Sets whether this widget needs a keyboard.
- `unsigned int output () const`
Returns if a widget is used for output only.
- `FI_Group * parent () const`
Returns a pointer to the parent widget.
- `void parent (FI_Group *p)`
Internal use only - "for hacks only".
- `void position (int X, int Y)`
Repositions the window or widget.
- `void redraw ()`
Schedules the drawing of the widget.
- `void redraw_label ()`
Schedules the drawing of the label.
- `virtual void resize (int x, int y, int w, int h)`
Changes the size or position of the widget.
- `FI_Color selection_color () const`
Gets the selection color.
- `void selection_color (FI_Color a)`
Sets the selection color.
- `void set_active ()`

- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

 - void [set_output](#) ()
- Sets a widget to output only.*

 - void [set_visible](#) ()
- Makes the widget visible.*

 - void [set_visible_focus](#) ()
- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void [shortcut_label](#) (int value)
- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - virtual void [show](#) ()
- Makes a widget visible.*

 - void [size](#) (int W, int H)
- Changes the size of the widget.*

 - int [take_focus](#) ()
- Gives the widget the keyboard focus.*

 - unsigned int [takeevents](#) () const
- Returns if the widget is able to take events.*

 - int [test_shortcut](#) ()
- Returns true if the widget's label contains the entered '&x' shortcut.*

 - const char * [tooltip](#) () const
- Gets the current tooltip text.*

 - void [tooltip](#) (const char *text)
- Sets the current tooltip text.*

 - [FI_Window](#) * [top_window](#) () const
- Returns a pointer to the top-level window for the widget.*

 - [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
- Finds the x/y offset of the current widget relative to the top-level window.*

 - [uchar](#) [type](#) () const
- Gets the widget type.*

 - void [type](#) ([uchar](#) t)
- Sets the widget type.*

 - int [use_accents_menu](#) ()
- Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*

 - void * [user_data](#) () const
- Gets the user data for this widget.*

 - void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
- Sets the user data for this widget.*

 - void [user_data](#) (void *v)
- Sets the user data for this widget.*

 - int [vertical_label_margin](#) ()
- Get the spacing between the label and the vertical edge of the widget.*

 - void [vertical_label_margin](#) (int px)
- Set the spacing between the label and the vertical edge of the widget.*

 - unsigned int [visible](#) () const
- Returns whether a widget is visible.*

 - unsigned int [visible_focus](#) () const
- Checks whether this widget has a visible focus.*

- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **FL_When when** () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **FL_Window * window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~FL_Widget** ()
Destroys the widget.

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
Draws the widget.
- void **draw** (int, int, int, int)
- int **handle** (int, int, int, int, int)

Protected Member Functions inherited from **FL_Valuator**

- **FL_Valuator** (int X, int Y, int W, int H, const char *L)
*Creates a new **FL_Valuator** widget using the given position, size, and label string.*
- void **handle_drag** (double newvalue)
*Called during a drag operation, after an **FL_WHEN_CHANGED** event is received and before the callback.*
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
*Called after an **FL_WHEN_RELEASE** event is received and before the callback.*
- int **horizontal** () const
*Tells if the valuator is an **FL_HORIZONTAL** one.*
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void **value_damage** ()
Asks for partial redraw.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
 ,
[OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
 ,
[MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
 ,
[GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
 = 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.132.1 Detailed Description

The [Fl_Slider](#) widget contains a sliding knob inside a box.

It is often used as a scrollbar. Moving the box all the way to the top/left sets it to the [minimum\(\)](#), and to the bottom/right to the [maximum\(\)](#). The [minimum\(\)](#) may be greater than the [maximum\(\)](#) to reverse the slider direction.

Use void [Fl_Widget::type\(int\)](#) to set how the slider is drawn, which can be one of the following:

- [FL_VERTICAL](#) - Draws a vertical slider (this is the default).
- [FL_HORIZONTAL](#) - Draws a horizontal slider.
- [FL_VERT_FILL_SLIDER](#) - Draws a filled vertical slider, useful as a progress or value meter.
- [FL_HOR_FILL_SLIDER](#) - Draws a filled horizontal slider, useful as a progress or value meter.
- [FL_VERT_NICE_SLIDER](#) - Draws a vertical slider with a nice looking control knob.
- [FL_HOR_NICE_SLIDER](#) - Draws a horizontal slider with a nice looking control knob.

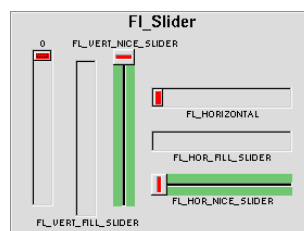


Figure 11.44 [Fl_Slider](#)

11.132.2 Constructor & Destructor Documentation

11.132.2.1 [Fl_Slider\(\)](#)

```
Fl_Slider::Fl_Slider (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Slider](#) widget using the given position, size, and label string.

The default boxtype is [FL_DOWN_BOX](#).

11.132.3 Member Function Documentation

11.132.3.1 bounds()

```
void Fl_Slider::bounds (
    double a,
    double b)
```

Sets the minimum (a) and maximum (b) values for the valuator widget.
if at least one of the values is changed, a partial redraw is asked.

11.132.3.2 draw()

```
void Fl_Slider::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

Reimplemented in [Fl_Value_Slider](#).

11.132.3.3 handle()

```
int Fl_Slider::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited handle() method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your handle() method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the handle() method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

Reimplemented in [Fl_Value_Slider](#).

11.132.3.4 scrollvalue()

```
int Fl_Slider::scrollvalue (
    int pos,
    int size,
    int first,
    int total)
```

Sets the size and position of the sliding knob in the box.

Parameters

in	<i>pos</i>	position of first line displayed
in	<i>size</i>	size of window in lines
in	<i>first</i>	number of first line
in	<i>total</i>	total number of lines Returns <code>Fl_Valuator::value(p)</code>

11.132.3.5 slider_size()

```
void Fl_Slider::slider_size (
    double v)
```

Set the dimensions of the moving piece of slider.

This is the fraction of the size of the entire widget. If you set this to 1 then the slider cannot move. The default value is .08.

For the "fill" sliders this is the size of the area around the end that causes a drag effect rather than causing the slider to jump to the mouse.

The documentation for this class was generated from the following files:

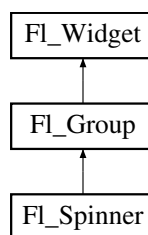
- `Fl_Slider.H`
- `Fl_Slider.cxx`

11.133 Fl_Spinner Class Reference

This widget is a combination of a numerical input widget and repeat buttons.

```
#include <Fl_Spinner.H>
```

Inheritance diagram for `Fl_Spinner`:



Classes

- class [Fl_Spinner_Input](#)

Public Member Functions

- [Fl_Color](#) **color** () const
Returns the background color of the spinner widget's input field.
- void **color** ([Fl_Color](#) v)
Sets the background color of the spinner widget's input field.
- [Fl_Spinner](#) (int X, int Y, int W, int H, const char *L=0)

- Creates a new [FL_Spinner](#) widget using the given position, size, and label string.*

 - `const char * format () const`
Returns the format string for the value.
 - `void format (const char *f)`
Sets the format string for the value.
 - `int handle (int event) FL\_OVERRIDE`
Handles the specified event.
 - `double maximum () const`
Gets the maximum value of the widget.
 - `void maximum (double m)`
Sets the maximum value of the widget.
 - `int maximum_size () const`
Returns the maximum width of the input field.
 - `void maximum_size (int m)`
Sets the maximum width of the input field.
 - `double minimum () const`
Gets the minimum value of the widget.
 - `void minimum (double m)`
Sets the minimum value of the widget.
 - `void range (double a, double b)`
Sets the minimum and maximum values for the widget.
 - `void resize (int X, int Y, int W, int H) FL\_OVERRIDE`
Resizes the [FL_Group](#) widget and all of its children.
 - `FL\_Color selection_color () const`
Returns the selection color of the spinner widget's input field.
 - `void selection_color (FL\_Color val)`
Sets the selection color of the spinner widget's input field.
 - `double step () const`
Gets the amount to change the value when the user clicks a button.
 - `void step (double s)`
Sets or returns the amount to change the value when the user clicks a button.
 - `FL\_Color textcolor () const`
Gets the color of the text in the input field.
 - `void textcolor (FL\_Color c)`
Sets the color of the text in the input field.
 - `FL\_Font textfont () const`
Gets the font of the text in the input field.
 - `void textfont (FL\_Font f)`
Sets the font of the text in the input field.
 - `FL\_Fonsize textsize () const`
Gets the size of the text in the input field.
 - `void textsize (FL\_Fonsize s)`
Sets the size of the text in the input field.
 - `uchar type () const`
Gets the numeric representation in the input field.
 - `void type (uchar v)`
Sets the numeric representation in the input field.
 - `double value () const`
Gets the current value of the widget.
 - `void value (double v)`
Sets the current value of the input widget.

- `int wrap () const`
Gets the wrap mode of the `FL_Spinner` widget.
- `void wrap (int set)`
Sets whether the spinner wraps around at upper and lower bounds.

Public Member Functions inherited from `FL_Group`

- `FL_Widget *& _ddfdesign_kludge ()`
This is for forms compatibility only.
- `void add (FL_Widget &)`
The widget is removed from its current group (if any) and then added to the end of this group.
- `void add (FL_Widget *o)`
See void `FL_Group::add(FL_Widget &w)`
- `void add_resizable (FL_Widget &o)`
Adds a widget to the group and makes it the resizable widget.
- `FL_Widget *const * array () const`
Returns a pointer to the internal array of children.
- `FL_Group const * as_group () const FL_OVERRIDE`
- `FL_Group * as_group () FL_OVERRIDE`
Returns an `FL_Group` pointer if this widget is an `FL_Group`.
- `void begin ()`
Sets the current group so you can build the widget tree by just constructing the widgets.
- `FL_Widget * child (int n) const`
Returns the n'th child.
- `int children () const`
Returns how many child widgets the group has.
- `void clear ()`
Deletes all child widgets from memory recursively.
- `unsigned int clip_children ()`
Returns the current clipping mode.
- `void clip_children (int c)`
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- `virtual int delete_child (int n)`
Removes the widget at `index` from the group and deletes it.
- `void end ()`
Exactly the same as `current(this->parent())`.
- `int find (const FL_Widget &o) const`
See int `FL_Group::find(const FL_Widget *w) const`.
- `int find (const FL_Widget *) const`
Searches the children for the widget and returns the index.
- `FL_Group (int, int, int, int, const char * = 0)`
Creates a new `FL_Group` widget using the given position, size, and label string.
- `void focus (FL_Widget *W)`
- `void forms_end ()`
This is for forms compatibility only.
- `void init_sizes ()`
Resets the internal array of widget sizes and positions.
- `void insert (FL_Widget &, int i)`
The widget is removed from its current group (if any) and then inserted into this group.
- `void insert (FL_Widget &o, FL_Widget *before)`
This does `insert(w, find(before))`.

- void **remove** (FL_Widget &)
Removes a widget from the group but does not delete it.
- void **remove** (FL_Widget *o)
Removes the widget o from the group.
- void **remove** (int index)
Removes the widget at index from the group but does not delete it.
- FL_Widget * **resizable** () const
Returns the group's resizable widget.
- void **resizable** (FL_Widget &o)
Sets the group's resizable widget.
- void **resizable** (FL_Widget *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual ~FL_Group ()
The destructor also deletes all the children.

Public Member Functions inherited from FL_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- FL_Align **align** () const
Gets the label alignment.
- void **align** (FL_Align alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class FL_GL_Window * **as_gl_window** ()
Returns an FL_GL_Window pointer if this widget is an FL_GL_Window.
- virtual class FL_GL_Window const * **as_gl_window** () const
- virtual FL_Window * **as_window** ()
Returns an FL_Window pointer if this widget is an FL_Window.
- virtual FL_Window const * **as_window** () const
- void **bind_deimage** (FL_Image *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (FL_Image *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- FL_Boxtype **box** () const
Gets the box type of the widget.
- void **box** (FL_Boxtype new_box)
Sets the box type for the widget.

- [FI_Callback_p callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) ([uchar](#) c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()

- Deactivates the widget.*
- [FL_Image](#) * [deimage](#) ()
 - Gets the image that is used as part of the widget label when in the inactive state.*
- const [FL_Image](#) * [deimage](#) () const
 - Gets the image that is used as part of the widget label when in the inactive state.*
- void [deimage](#) ([FL_Image](#) &img)
 - Sets the image to use as part of the widget label when in the inactive state.*
- void [deimage](#) ([FL_Image](#) *img)
 - Sets the image to use as part of the widget label when in the inactive state.*
- int [deimage_bound](#) () const
 - Returns whether the inactive image is managed by the widget.*
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
 - Calls the widget callback function with default arguments.*
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
 - Calls the widget callback function with arbitrary arguments.*
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
 - Calls the widget callback function with arbitrary arguments.*
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
 - Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int [h](#) () const
 - Gets the widget height.*
- virtual void [hide](#) ()
 - Makes a widget invisible.*
- int [horizontal_label_margin](#) ()
 - Get the spacing between the label and the horizontal edge of the widget.*
- void [horizontal_label_margin](#) (int px)
 - Set the spacing between the label and the horizontal edge of the widget.*
- [FL_Image](#) * [image](#) ()
 - Gets the image that is used as part of the widget label when in the active state.*
- const [FL_Image](#) * [image](#) () const
 - Gets the image that is used as part of the widget label when in the active state.*
- void [image](#) ([FL_Image](#) &img)
 - Sets the image to use as part of the widget label when in the active state.*
- void [image](#) ([FL_Image](#) *img)
 - Sets the image to use as part of the widget label when in the active state.*
- int [image_bound](#) () const
 - Returns whether the image is managed by the widget.*
- int [inside](#) (const [FL_Widget](#) *wgt) const
 - Checks if this widget is a child of wgt.*
- int [is_label_copied](#) () const
 - Returns whether the current label was assigned with [copy_label\(\)](#).*
- const char * [label](#) () const
 - Gets the current label text.*
- void [label](#) (const char *text)
 - Sets the current label pointer.*
- void [label](#) ([FL_Labeltype](#) a, const char *b)
 - Shortcut to set the label text and type in one call.*
- int [label_image_spacing](#) ()
 - Return the gap size between the label and the image.*
- void [label_image_spacing](#) (int gap)
 - Set the gap between the label and the image in pixels.*

- [FL_Color labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize labelsiz](#)e () const
Gets the font size in pixels.
- void [labelsiz](#)e ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group * parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [FL_Color selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - virtual void **show** ()
- Makes a widget visible.*

 - void **size** (int W, int H)
- Changes the size of the widget.*

 - int **take_focus** ()
- Gives the widget the keyboard focus.*

 - unsigned int **takeevents** () const
- Returns if the widget is able to take events.*

 - int **test_shortcut** ()
- Returns true if the widget's label contains the entered '&x' shortcut.*

 - const char * **tooltip** () const
- Gets the current tooltip text.*

 - void **tooltip** (const char *text)
- Sets the current tooltip text.*

 - Fl_Window * **top_window** () const
- Returns a pointer to the top-level window for the widget.*

 - Fl_Window * **top_window_offset** (int &xoff, int &yoff) const
- Finds the x/y offset of the current widget relative to the top-level window.*

 - uchar **type** () const
- Gets the widget type.*

 - void **type** (uchar t)
- Sets the widget type.*

 - int **use_accents_menu** ()
- Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*

 - void * **user_data** () const
- Gets the user data for this widget.*

 - void **user_data** (Fl_Callback_User_Data *v, bool auto_free)
- Sets the user data for this widget.*

 - void **user_data** (void *v)
- Sets the user data for this widget.*

 - int **vertical_label_margin** ()
- Get the spacing between the label and the vertical edge of the widget.*

 - void **vertical_label_margin** (int px)
- Set the spacing between the label and the vertical edge of the widget.*

 - unsigned int **visible** () const
- Returns whether a widget is visible.*

 - unsigned int **visible_focus** () const
- Checks whether this widget has a visible focus.*

 - void **visible_focus** (int v)
- Modifies keyboard focus navigation.*

 - int **visible_r** () const
- Returns whether a widget and all its parents are visible.*

 - int **w** () const
- Gets the widget width.*

 - Fl_When **when** () const
- Returns the conditions under which the callback is called.*

 - void **when** (uchar i)
- Sets the flags used to decide when a callback is called.*

 - Fl_Window * **window** () const
- Returns a pointer to the nearest parent window up the widget hierarchy.*

- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~FL_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from `FL_Group`

- `FL_Rect * bounds ()`
Returns the internal array of widget sizes and positions.
- `void draw_child (FL_Widget &widget) const`
Forces a child to redraw.
- `void draw_children ()`
Draws all children of the group.
- `void draw_outside_label (const FL_Widget &widget) const`
Parents normally call this to draw outside labels of child widgets.
- `virtual int on_insert (FL_Widget *, int)`
Allow derived groups to act when a widget is added as a child.
- `virtual int on_move (int, int)`
Allow derived groups to act when a widget is moved within the group.
- `virtual void on_remove (int)`
Allow derived groups to act when a child widget is removed from the group.
- `int * sizes ()`
Returns the internal array of widget sizes and positions.
- `void update_child (FL_Widget &widget) const`
Draws a child only if it needs it.

Protected Member Functions inherited from `FL_Widget`

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If `FL_ALIGN_IMAGE_BACKDROP` is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const`
Draws a focus box for the widget at the given position and size.
- `void draw_label () const`

- Draws the widget's label at the defined label position.*
- void [draw_label](#) (int, int, int, int) const
- Draws the label in an arbitrary bounding box.*
- [FI_Widget](#) (int [x](#), int [y](#), int [w](#), int [h](#), const char *[label](#)=0L)
- Creates a widget at the given position and size.*
- unsigned int **flags** () const
- Gets the widget flags mask.*
- void [h](#) (int v)
- Internal use only.*
- void **set_flag** (unsigned int c)
- Sets a flag in the flags mask.*
- void [w](#) (int v)
- Internal use only.*
- void [x](#) (int v)
- Internal use only.*
- void [y](#) (int v)
- Internal use only.*

Protected Attributes

- [FI_Repeat_Button](#) **down_button_**
- [FI_Spinner_Input](#) **input_**
- [FI_Repeat_Button](#) **up_button_**

Additional Inherited Members

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * **current** ()
- Returns the currently active group.*
- static void **current** ([FI_Group](#) *g)
- Sets the current group.*

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
- The default callback for all widgets that don't set a callback.*
- static unsigned int **label_shortcut** (const char *t)
- Returns the Unicode value of the '&x' shortcut in a given text.*
- static int **test_shortcut** (const char *, const bool require_alt=false)
- Returns true if the given text t contains the entered '&x' shortcut.*

Protected Types inherited from [FI_Widget](#)

- enum {
- [INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
- [FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
- ,
- [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
- ,
- [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
- ,
- [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
- = 1<<19 ,
- [NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,

```
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , POPUP = 1<<25 , USERFLAG3 = 1<<29 , USERFLAG2 = 1<<30 ,
USERFLAG1 = 1<<31 }
```

flags possible values enumeration.

11.133.1 Detailed Description

This widget is a combination of a numerical input widget and repeat buttons. The user can either type into the input area or use the buttons to change the value.

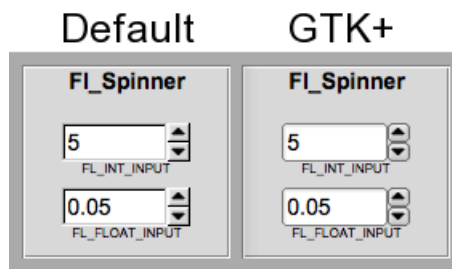


Figure 11.45 Fl_Spinner widget

11.133.2 Constructor & Destructor Documentation

11.133.2.1 Fl_Spinner()

```
Fl_Spinner::Fl_Spinner (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Spinner](#) widget using the given position, size, and label string. The inherited destructor destroys the widget and any value associated with it.

11.133.3 Member Function Documentation

11.133.3.1 draw()

```
void Fl_Spinner::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own draw() method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Reimplemented from [Fl_Group](#).

11.133.3.2 handle()

```
int Fl_Spinner::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	event	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Group](#).

11.133.3.3 [resize\(\)](#)

```
void Fl_Spinner::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Resizes the [Fl_Group](#) widget and all of its children.

The [Fl_Group](#) widget first resizes itself, and then it moves and resizes all its children according to the rules documented for [Fl_Group::resizable\(Fl_Widget*\)](#)

See also

[Fl_Group::resizable\(Fl_Widget*\)](#)

[Fl_Group::resizable\(\)](#)

[Fl_Widget::resize\(int,int,int,int\)](#)

Reimplemented from [Fl_Group](#).

11.133.3.4 [step\(\)](#) [1/2]

```
double Fl_Spinner::step () const [inline]
```

Gets the amount to change the value when the user clicks a button.

See also

[Fl_Spinner::step\(double\)](#)

11.133.3.5 [step\(\)](#) [2/2]

```
void Fl_Spinner::step (
    double s)
```

Sets or returns the amount to change the value when the user clicks a button.

Before setting step to a non-integer value, the spinner [type\(\)](#) should be changed to floating point.

See also

[double Fl_Spinner::step\(\) const](#)

11.133.3.6 type() [1/2]

```
uchar Fl_Spinner::type () const [inline]
```

Gets the numeric representation in the input field.

See also

[Fl_Spinner::type\(uchar\)](#)

11.133.3.7 type() [2/2]

```
void Fl_Spinner::type (
    uchar v)
```

Sets the numeric representation in the input field.

Valid values are FL_INT_INPUT and FL_FLOAT_INPUT. Also changes the [format\(\)](#) template. Setting a new spinner type via a superclass pointer will not work.

Note

[type\(\)](#) is not a virtual function.

11.133.3.8 value()

```
void Fl_Spinner::value (
    double v) [inline]
```

Sets the current value of the input widget.

Before setting value to a non-integer value, the spinner [type\(\)](#) should be changed to floating point.

11.133.3.9 wrap() [1/2]

```
int Fl_Spinner::wrap () const [inline]
```

Gets the wrap mode of the [Fl_Spinner](#) widget.

See also

void [wrap\(int\)](#)

Since

1.4.0

11.133.3.10 wrap() [2/2]

```
void Fl_Spinner::wrap (
    int set) [inline]
```

Sets whether the spinner wraps around at upper and lower bounds.

If wrap mode is on the spinner value is set to the [minimum\(\)](#) or [maximum\(\)](#) if the value exceeds the upper or lower bounds, resp., if it was changed by one of the buttons or the FL_Up or FL_Down keys.

The spinner stops at the upper and lower bounds if wrap mode is off.

The default wrap mode is on for backwards compatibility with FLTK 1.3.x and older versions.

Note

Wrap mode does not apply to the input field if the input value is edited directly as a number. The input value is always clipped to the allowed range as if wrap mode was off when the input field is left (i.e. loses focus).

See also

[minimum\(\)](#), [maximum\(\)](#)

Parameters

<code>in</code>	<code>set</code>	non-zero sets wrap mode, zero resets wrap mode

Since

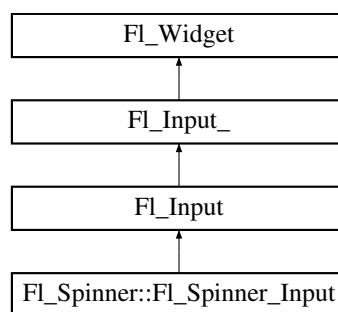
1.4.0

The documentation for this class was generated from the following files:

- FI_Spinner.H
- FI_Spinner.cxx

11.134 FI_Spinner::FI_Spinner_Input Class Reference

Inheritance diagram for FI_Spinner::FI_Spinner_Input:



Public Member Functions

- **FI_Spinner_Input** (int X, int Y, int W, int H)
- int **handle** (int event) **FL_OVERRIDE**
Handles events of [FI_Spinner](#)'s embedded input widget.

Public Member Functions inherited from [FI_Input](#)

- [FI_Input](#) (int, int, int, int, const char *=0)
Creates a new [FI_Input](#) widget using the given position, size, and label string.

Public Member Functions inherited from [FI_Input_](#)

- int [append](#) (const char *t, int l=0, char keep_selection=0)
Append text at the end.
- bool [can_redo](#) () const
Check if there is a redo action available.
- bool [can_undo](#) () const
Check if the last operation can be undone.
- int [copy](#) (int clipboard)
Put the current selection into the clipboard.
- int [copy_cuts](#) ()
Copies the yank buffer to the clipboard.
- [FI_Color](#) [cursor_color](#) () const
Gets the color of the cursor.
- void [cursor_color](#) ([FI_Color](#) n)
Sets the color of the cursor.

- `int cut ()`
Deletes the current selection.
- `int cut (int a, int b)`
*Deletes all characters between index *a* and *b*.*
- `int cut (int n)`
*Deletes the next *n* bytes rounded to characters before or after the cursor.*
- `double dvalue () const`
Returns the widget text interpreted as a floating point number.
- `FL_Input_ (int, int, int, int, const char *=0)`
*Creates a new *FL_Input_* widget.*
- `unsigned int index (int i) const`
*Returns the character at index *i*.*
- `int input_type () const`
Gets the input field type.
- `void input_type (int t)`
Sets the input field type.
- `int insert (const char *t, int l=0)`
Inserts text at the cursor position.
- `int insert_position () const`
Gets the position of the text cursor.
- `int insert_position (int p)`
Sets the cursor position and mark.
- `int insert_position (int p, int m)`
Sets the index for the cursor and mark.
- `int ivalue () const`
Returns the widget text interpreted as a signed integer.
- `int mark () const`
Gets the current selection mark.
- `int mark (int m)`
Sets the current selection mark.
- `int maximum_size () const`
Gets the maximum length of the input field in characters.
- `void maximum_size (int m)`
Sets the maximum length of the input field in characters.
- `int position () const`
- `int position (int p)`
- `int position (int p, int m)`
- `int readonly () const`
Gets the read-only state of the input field.
- `void readonly (int b)`
Sets the read-only state of the input field.
- `int redo ()`
Redo previous undo operation.
- `int replace (int b, int e, const char *text, int ilen=0)`
*Deletes text from *b* to *e* and inserts the new string *text*.*
- `void resize (int, int, int, int) FL_OVERRIDE`
Changes the size of the widget.
- `int shortcut () const`
Return the shortcut key associated with this widget.
- `void shortcut (int s)`
Sets the shortcut key associated with this widget.

- int **size** () const
*Returns the number of bytes in **value()**.*
- void **size** (int W, int H)
Sets the width and height of this widget.
- int **static_value** (const char *)
Changes the widget text.
- int **static_value** (const char *, int)
Changes the widget text.
- int **tab_nav** () const
Gets whether the Tab key causes focus navigation in multiline input fields or not.
- void **tab_nav** (int val)
*Sets whether the Tab key does focus navigation, or inserts tab characters into **Fl_Multiline_Input**.*
- **Fl_Color** **textcolor** () const
Gets the color of the text in the input field.
- void **textcolor** (**Fl_Color** n)
Sets the color of the text in the input field.
- **Fl_Font** **textfont** () const
Gets the font of the text in the input field.
- void **textfont** (**Fl_Font** s)
Sets the font of the text in the input field.
- **Fl_Fontsize** **textsize** () const
Gets the size of the text in the input field.
- void **textsize** (**Fl_Fontsize** s)
Sets the size of the text in the input field.
- int **undo** ()
Undoes previous changes to the text buffer.
- const char * **value** () const
Returns the text displayed in the widget.
- int **value** (const char *)
Changes the widget text.
- int **value** (const char *, int)
Changes the widget text.
- int **value** (double value)
Changes the widget text to a floating point number ("%g").
- int **value** (int value)
Changes the widget text to a signed integer number.
- int **wrap** () const
Gets the word wrapping state of the input field.
- void **wrap** (int b)
Sets the word wrapping state of the input field.
- ~**Fl_Input** ()
Destroys the widget.

Public Member Functions inherited from **Fl_Widget**

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.

- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [Fl_Align](#) align () const
Gets the label alignment.
- void [align](#) ([Fl_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()
Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [Fl_Group](#) * [as_group](#) ()
Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).
- virtual [Fl_Group](#) const * [as_group](#) () const
- virtual [Fl_Window](#) * [as_window](#) ()
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- virtual [Fl_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype](#) box () const
Gets the box type of the widget.
- void [box](#) ([Fl_Boxtype](#) new_box)
Sets the box type for the widget.
- [Fl_Callback_p](#) callback () const
Gets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([Fl_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.

- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FL_Color` bg)
Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)
Sets the background and selection color of the widget.
- `FL_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FL_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FL_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const

- Gets the widget height.*

 - virtual void `hide` ()
- Makes a widget invisible.*

 - int `horizontal_label_margin` ()

Get the spacing between the label and the horizontal edge of the widget.

 - void `horizontal_label_margin` (int px)

Set the spacing between the label and the horizontal edge of the widget.

 - `FL_Image` * `image` ()

Gets the image that is used as part of the widget label when in the active state.

 - const `FL_Image` * `image` () const

Gets the image that is used as part of the widget label when in the active state.

 - void `image` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the active state.

 - void `image` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the active state.

 - int `image_bound` () const

Returns whether the image is managed by the widget.

 - int `inside` (const `FL_Widget` *wgt) const

Checks if this widget is a child of wgt.

 - int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.

 - const char * `label` () const

Gets the current label text.

 - void `label` (const char *text)

Sets the current label pointer.

 - void `label` (`FL_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.

 - int `label_image_spacing` ()

Return the gap size between the label and the image.

 - void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.

 - `FL_Color` `labelcolor` () const

Gets the label color.

 - void `labelcolor` (`FL_Color` c)

Sets the label color.

 - `FL_Font` `labelfont` () const

Gets the font to use.

 - void `labelfont` (`FL_Font` f)

Sets the font to use.

 - `FL_Fonsize` `labelsize` () const

Gets the font size in pixels.

 - void `labelsize` (`FL_Fonsize` pix)

Sets the font size in pixels.

 - `FL_Labeltype` `labeltype` () const

Gets the label type.

 - void `labeltype` (`FL_Labeltype` a)

Sets the label type.

 - void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.

 - bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.

- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- [FL_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FL_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FL_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const

- Gets the widget type.*

 - void **type** (uchar t)

Sets the widget type.
- int **use_accents_menu** ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const

Gets the user data for this widget.
- void **user_data** (FI_Callback_User_Data *v, bool auto_free)

Sets the user data for this widget.
- void **user_data** (void *v)

Sets the user data for this widget.
- int **vertical_label_margin** ()

Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const

Returns whether a widget is visible.
- unsigned int **visible_focus** () const

Checks whether this widget has a visible focus.
- void **visible_focus** (int v)

Modifies keyboard focus navigation.
- int **visible_r** () const

Returns whether a widget and all its parents are visible.
- int **w** () const

Gets the widget width.
- **FI_When when** () const

Returns the conditions under which the callback is called.
- void **when** (uchar i)

Sets the flags used to decide when a callback is called.
- **FI_Window * window** () const

Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const

Gets the widget position in its window.
- int **y** () const

Gets the widget position in its window.
- virtual **~FI_Widget** ()

Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)

The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)

Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)

Returns true if the given text t contains the entered '&x' shortcut.

Static Public Attributes inherited from FI_Input

- static const char * **copy_menu_text** = "Copy"
[this text may be customized at run-time]
- static const char * **cut_menu_text** = "Cut"
[this text may be customized at run-time]
- static const char * **paste_menu_text** = "Paste"
[this text may be customized at run-time]

Protected Types inherited from FI_Widget

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from FI_Input

- void **draw** () **FL_OVERRIDE**
Draws the widget.
- int **handle_key** ()
Handles a keystroke.
- int **handle_rmb** ()
Handle right mouse button down events.

Protected Member Functions inherited from FI_Input_

- int **apply_undo** ()
Apply the current undo/redo operation.
- void **drawtext** (int, int, int, int)
Draws the text in the passed bounding box.
- void **drawtext** (int, int, int, int, bool draw_active)
Draws the text in the passed bounding box.
- void **handle_mouse** (int, int, int, int, int keepmark=0)
Handles mouse clicks and mouse moves.
- int **handletext** (int e, int, int, int, int)
Handles all kinds of text field related events.
- int **line_end** (int i) const
Finds the end of a line.
- int **line_start** (int i) const
Finds the start of a line.
- int **linesPerPage** ()
- void **maybe_do_callback** (FI_Callback_Reason reason=FL_REASON_UNKNOWN)

- int **up_down_position** (int, int keepmark=0)
Moves the cursor to the column given by up_down_pos.
- int **word_end** (int i) const
Finds the end of a word.
- int **word_start** (int i) const
Finds the start of a word.
- int **xscroll** () const
- int **yscroll** () const
- void **yscroll** (int yOffset)

Protected Member Functions inherited from **Fl_Widget**

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (Fl_Boxtype t, Fl_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **Fl_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

11.134.1 Member Function Documentation

11.134.1.1 handle()

```
int Fl_Spinner::Fl_Spinner_Input::handle (
    int event) [virtual]
```

Handles events of [Fl_Spinner](#)'s embedded input widget.

Works like [Fl_Input::handle\(\)](#) but ignores FL_Up and FL_Down keys so they can be handled by the parent widget ([Fl_Spinner](#)).

Reimplemented from [Fl_Input](#).

The documentation for this class was generated from the following files:

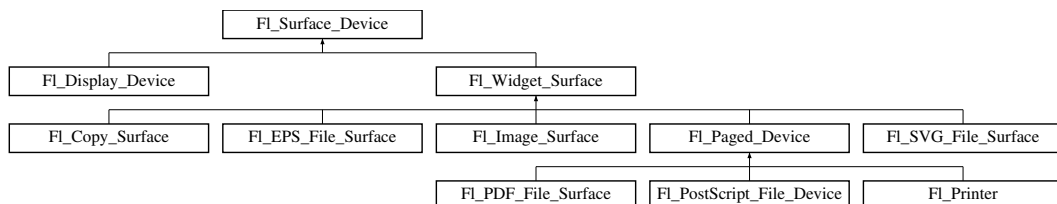
- [Fl_Spinner.H](#)
- [Fl_Spinner.cxx](#)

11.135 Fl_Surface_Device Class Reference

A drawing surface that's susceptible to receive graphical output.

```
#include <Fl_Device.H>
```

Inheritance diagram for [Fl_Surface_Device](#):



Public Member Functions

- [Fl_Graphics_Driver](#) * **driver** ()
Returns the graphics driver of this drawing surface.
- virtual bool [is_current](#) ()
Is this surface the current drawing surface?
- virtual void [set_current](#) (void)
Make this surface the current drawing surface.
- virtual ~**Fl_Surface_Device** ()
The destructor.

Static Public Member Functions

- static [Fl_Surface_Device](#) * [pop_current](#) ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void [push_current](#) ([Fl_Surface_Device](#) *new_current)
Pushes new_current on top of the stack of current drawing surfaces, and makes it current.
- static [Fl_Surface_Device](#) * [surface](#) ()
The current drawing surface.

Protected Member Functions

- void **driver** ([Fl_Graphics_Driver](#) *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void [end_current](#) ()
FLTK calls this each time a surface ceases to be the current drawing surface.
- **Fl_Surface_Device** ([Fl_Graphics_Driver](#) *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

11.135.1 Detailed Description

A drawing surface that's susceptible to receive graphical output.

Any FLTK application has at any time a current drawing surface to which all drawing requests are directed. The current surface is given by [Fl_Surface_Device::surface\(\)](#). When `main()` begins running, the current drawing surface has been set to the computer's display, an instance of the [Fl_Display_Device](#) class.

A drawing surface other than the computer's display, is typically used as follows:

1. Create `surface`, an object from an [Fl_Surface_Device](#) derived class (e.g., [Fl_Image_Surface](#), [Fl_Printer](#), [Fl_Copy_Surface](#)).
2. Call `Fl_Surface_Device::push_current(surface);` to redirect all graphics requests to `surface` which becomes the new current drawing surface (not necessary with classes [Fl_Printer](#) / [Fl_PostScript_File_Device](#) / [Fl_PDF_File_Surface](#) because it is done by [Fl_Paged_Device::begin_page\(\)](#)).
3. At this point all of the [Drawing functions](#) (e.g., [fl_rect\(\)](#)) or the [Color & Font functions](#) or [Drawing Images](#) functions (e.g., [fl_draw_image\(\)](#), [Fl_Image::draw\(\)](#)) operate on the new current drawing surface. It's also possible to draw to the current surface any widget with [Fl_Widget_Surface::draw\(Fl_Widget*, int, int\)](#), a window and its titlebar with [Fl_Widget_Surface::draw_decorated_window\(\)](#), or the content of a rectangular zone of a window with [Fl_Widget_Surface::print_window_part\(\)](#).
4. After all drawing requests have been performed, redirect graphics requests back to their previous destination with `Fl_Surface_Device::pop_current();` (not necessary with classes [Fl_Printer](#) / [Fl_PostScript_File_Device](#) / [Fl_PDF_File_Surface](#)).
5. Delete `surface`.

For back-compatibility, it is also possible to use the [Fl_Surface_Device::set_current\(\)](#) member function to change the current drawing surface, once to the new surface, once to the previous one.

11.135.2 Member Function Documentation

11.135.2.1 end_current()

```
virtual void Fl_Surface_Device::end_current () [inline], [protected], [virtual]
```

FLTK calls this each time a surface ceases to be the current drawing surface.

This member function is mostly of interest to developers of new [Fl_Surface_Device](#) derived classes. It allows to perform surface-specific operations necessary when this surface ceases to be current. Each implementation should end with a call to [Fl_Surface_Device::end_current\(\)](#).

Reimplemented in [Fl_PostScript_File_Device](#).

11.135.2.2 is_current()

```
bool Fl_Surface_Device::is_current () [virtual]
```

Is this surface the current drawing surface?

Reimplemented in [Fl_Copy_Surface](#), [Fl_Image_Surface](#), [Fl_PDF_File_Surface](#), and [Fl_Printer](#).

11.135.2.3 pop_current()

```
Fl_Surface_Device * Fl_Surface_Device::pop_current () [static]
```

Removes the top element from the current drawing surface stack, and makes the new top element current.

Returns

A pointer to the new current drawing surface.

See also

[Fl_Surface_Device::push_current\(Fl_Surface_Device *\)](#)

Version

1.4.0

11.135.2.4 push_current()

```
void Fl_Surface_Device::push_current (
    Fl_Surface_Device * new_current) [static]
```

Pushes `new_current` on top of the stack of current drawing surfaces, and makes it current.

`new_current` will receive all future graphics requests.

Any call to `push_current()` must be matched by a subsequent call to `Fl_Surface_Device::pop_current()`. The max height of this stack is 16.

Version

1.4.0

11.135.2.5 set_current()

```
void Fl_Surface_Device::set_current (
    void ) [virtual]
```

Make this surface the current drawing surface.

This surface will receive all future graphics requests.

Since FLTK 1.4.0 the preferred API to change the current drawing surface is `Fl_Surface_Device::push_current()` / `Fl_Surface_Device::pop_current()`.

Note

It is recommended to use this function only as follows :

- The current drawing surface is the display;
- make current another surface, e.g., an `Fl_Printer` or an `Fl_Image_Surface` object, calling `set_current()` on this object;
- draw to that surface;
- make the display current again with `Fl_Display_Device::display_device()->set_current()`;
don't do any other call to `set_current()` before this one.

Other scenarios of drawing surface changes should be performed via `Fl_Surface_Device::push_current()` and `Fl_Surface_Device::pop_current()`.

Reimplemented in `Fl_Copy_Surface`, `Fl_Image_Surface`, `Fl_PDF_File_Surface`, `Fl_PostScript_File_Device`, and `Fl_Printer`.

11.135.2.6 surface()

```
static Fl_Surface_Device * Fl_Surface_Device::surface () [inline], [static]
```

The current drawing surface.

In other words, the `Fl_Surface_Device` object that currently receives all graphics requests.

Note

It's possible to transiently remove the GUI scaling factor in place in the current drawing surface with `fl_override_scale()`.

The documentation for this class was generated from the following files:

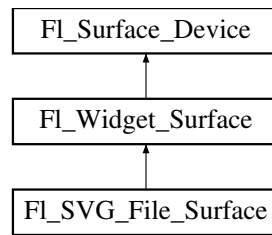
- `Fl_Device.H`
- `Fl_Device.cxx`

11.136 Fl_SVG_File_Surface Class Reference

A drawing surface producing a Scalable Vector Graphics (SVG) file.

```
#include <Fl_SVG_File_Surface.H>
```

Inheritance diagram for `Fl_SVG_File_Surface`:



Public Member Functions

- `int close ()`
Closes the FILE pointer where SVG data is output.
- `FILE * file ()`
Returns the underlying FILE pointer.
- `Fl_SVG_File_Surface (int width, int height, FILE *svg, int(*closef)(FILE *)=NULL)`
Constructor of the SVG drawing surface.
- `void origin (int *x, int *y) FL_OVERRIDE`
Computes the coordinates of the current origin of graphics functions.
- `void origin (int x, int y) FL_OVERRIDE`
Sets the position of the origin of graphics in the drawable part of the drawing surface.
- `int printable_rect (int *w, int *h) FL_OVERRIDE`
Computes the width and height of the drawable area of the drawing surface.
- `void translate (int x, int y) FL_OVERRIDE`
Translates the current graphics origin accounting for the current rotation.
- `void untranslate () FL_OVERRIDE`
Undoes the effect of a previous `translate()` call.
- `~Fl_SVG_File_Surface ()`
Destructor.

Public Member Functions inherited from `Fl_Widget_Surface`

- `void draw (Fl_Widget *widget, int delta_x=0, int delta_y=0)`
Draws the widget on the drawing surface.
- `void draw_decorated_window (Fl_Window *win, int x_offset=0, int y_offset=0)`
Draws a window with its title bar and frame if any.
- `void print_window_part (Fl_Window *win, int x, int y, int w, int h, int delta_x=0, int delta_y=0)`
Draws a rectangular part of an on-screen window.

Public Member Functions inherited from `Fl_Surface_Device`

- `Fl_Graphics_Driver * driver ()`
Returns the graphics driver of this drawing surface.
- `virtual bool is_current ()`
Is this surface the current drawing surface?
- `virtual void set_current (void)`
Make this surface the current drawing surface.
- `virtual ~Fl_Surface_Device ()`
The destructor.

Additional Inherited Members

Static Public Member Functions inherited from [Fl_Surface_Device](#)

- static [Fl_Surface_Device](#) * [pop_current](#) ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void [push_current](#) ([Fl_Surface_Device](#) *new_current)
Pushes new_current on top of the stack of current drawing surfaces, and makes it current.
- static [Fl_Surface_Device](#) * [surface](#) ()
The current drawing surface.

Protected Member Functions inherited from [Fl_Widget_Surface](#)

- [Fl_Widget_Surface](#) ([Fl_Graphics_Driver](#) *d)
The constructor.

Protected Member Functions inherited from [Fl_Surface_Device](#)

- void [driver](#) ([Fl_Graphics_Driver](#) *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void [end_current](#) ()
FLTK calls this each time a surface ceases to be the current drawing surface.
- [Fl_Surface_Device](#) ([Fl_Graphics_Driver](#) *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

Protected Attributes inherited from [Fl_Widget_Surface](#)

- int [x_offset](#)
horizontal offset to the origin of graphics coordinates
- int [y_offset](#)
vertical offset to the origin of graphics coordinates

11.136.1 Detailed Description

A drawing surface producing a Scalable Vector Graphics (SVG) file.

This drawing surface allows to store any FLTK graphics in vectorial form in a "Scalable Vector Graphics" file.

Usage example:

```
Fl_Window *win = ...// Window to draw to a .svg file
int ww = win->decorated_w();
int wh = win->decorated_h();
FILE *svg = fl_fopen("/path/to/mywindow.svg", "w");
if (svg) {
    Fl_SVG_File_Surface *surface = new Fl_SVG_File_Surface(ww, wh, svg);
    Fl_Surface_Device::push_current(surface);
    fl_color(FL_WHITE);
    fl_rectf(0, 0, ww, wh);
    surface->draw_decorated_window(win);
    Fl_Surface_Device::pop_current();
    delete surface; // the .svg file is not complete until the destructor was run
    fclose(svg);
}
```

Note

FLTK uses the PNG and JPEG libraries to encode images to the SVG format. For this reason, class [Fl_SVG_File_Surface](#) is placed in the fltk_images library. If JPEG is not available at application build time, PNG is enough (but produces a quite larger output). If PNG isn't available either, images don't appear in the SVG output.

11.136.2 Constructor & Destructor Documentation

11.136.2.1 Fl_SVG_File_Surface()

```
Fl_SVG_File_Surface::Fl_SVG_File_Surface (
    int width,
    int height,
    FILE * svg,
    int (* closef ) (FILE *) = NULL)
```

Constructor of the SVG drawing surface.

Parameters

<i>width,height</i>	Width and height of the graphics area in FLTK drawing units
<i>svg</i>	A writable FILE pointer where the SVG data are to be sent. The resulting SVG data are not complete until after destruction of the Fl_SVG_File_Surface object or after calling close() .
<i>closef</i>	If not NULL, the destructor and close() will call <code>closef(svg)</code> after all SVG data has been sent. If NULL, <code>fclose(svg)</code> is called instead. This allows to close the FILE pointer by, e.g., <code>pclose</code> , or, using a function such as <code>"int keep_open(FILE*) {return 0;}"</code> , to keep it open after completion of all output to <code>svg</code> . Function <code>closef</code> should return non zero to indicate an error.

11.136.2.2 ~Fl_SVG_File_Surface()

```
Fl_SVG_File_Surface::~~Fl_SVG_File_Surface ()
```

Destructor.

The underlying FILE pointer is processed as by [close\(\)](#).

11.136.3 Member Function Documentation

11.136.3.1 close()

```
int Fl_SVG_File_Surface::close ()
```

Closes the FILE pointer where SVG data is output.

The underlying FILE is closed by function `fclose()` unless another function was set at object's construction time. The only operation possible after this on the [Fl_SVG_File_Surface](#) object is its destruction.

Returns

The value returned by the closing function call.

11.136.3.2 origin() [1/2]

```
void Fl_SVG_File_Surface::origin (
    int * x,
    int * y) [virtual]
```

Computes the coordinates of the current origin of graphics functions.

Parameters

out	<i>x,y</i>	If non-null, *x and *y are set to the horizontal and vertical coordinates of the graphics origin.

Reimplemented from [Fl_Widget_Surface](#).

11.136.3.3 origin() [2/2]

```
void Fl_SVG_File_Surface::origin (
    int x,
    int y) [virtual]
```

Sets the position of the origin of graphics in the drawable part of the drawing surface.

Arguments should be expressed relatively to the result of a previous [printable_rect\(\)](#) call. That is, `printable_rect(&w, &h); origin(w/2, 0);` sets the graphics origin at the top center of the drawable area. Successive [origin\(\)](#) calls don't combine their effects. [Origin\(\)](#) calls are not affected by [rotate\(\)](#) calls (for classes derived from [Fl_Paged_Device](#)).

Parameters

<code>in</code>	<code>x,y</code>	Horizontal and vertical positions in the drawing surface of the desired origin of graphics.

Reimplemented from [Fl_Widget_Surface](#).

11.136.3.4 printable_rect()

```
int Fl_SVG_File_Surface::printable_rect (
    int * w,
    int * h) [virtual]
```

Computes the width and height of the drawable area of the drawing surface.

Values are in the same unit as that used by FLTK drawing functions and are unchanged by calls to [origin\(\)](#). If the object is derived from class [Fl_Paged_Device](#), values account for the user-selected paper type and print orientation and are changed by [scale\(\)](#) calls.

Returns

0 if OK, non-zero if any error

Reimplemented from [Fl_Widget_Surface](#).

11.136.3.5 translate()

```
void Fl_SVG_File_Surface::translate (
    int x,
    int y) [virtual]
```

Translates the current graphics origin accounting for the current rotation.

Each [translate\(\)](#) call must be matched by an [untranslate\(\)](#) call. Successive [translate\(\)](#) calls add up their effects.

Reimplemented from [Fl_Widget_Surface](#).

11.136.3.6 untranslate()

```
void Fl_SVG_File_Surface::untranslate (
    void ) [virtual]
```

Undoes the effect of a previous [translate\(\)](#) call.

Reimplemented from [Fl_Widget_Surface](#).

The documentation for this class was generated from the following file:

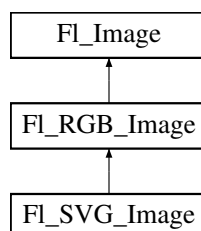
- [Fl_SVG_File_Surface.H](#)

11.137 Fl_SVG_Image Class Reference

The [Fl_SVG_Image](#) class supports loading, caching and drawing of scalable vector graphics (SVG) images.

```
#include <Fl_SVG_Image.H>
```

Inheritance diagram for [Fl_SVG_Image](#):



Public Member Functions

- [FI_SVG_Image](#) * [as_svg_image](#) () [FL_OVERRIDE](#)
Returns whether an image is an [FI_SVG_Image](#) or not.
- void [color_average](#) ([FI_Color](#) c, float i) [FL_OVERRIDE](#)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [FI_Image](#) * [copy](#) () const
- [FI_Image](#) * [copy](#) (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void [desaturate](#) () [FL_OVERRIDE](#)
The [desaturate\(\)](#) method converts an image to grayscale.
- void [draw](#) (int X, int Y)
- void [draw](#) (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [FI_SVG_Image](#) (const char *filename)
Load an SVG image from a file.
- [FI_SVG_Image](#) (const char *sharedname, const char *svg_data)
Load an SVG image from memory.
- [FI_SVG_Image](#) (const char *sharedname, const unsigned char *svg_data, size_t length)
Load an SVG image from memory.
- void [normalize](#) () [FL_OVERRIDE](#)
Makes sure the object is fully initialized.
- void [resize](#) (int width, int height)
Have the svg data (re-)rasterized using the given width and height values.
- virtual ~[FI_SVG_Image](#) ()
The destructor frees all memory and server resources that are used by the SVG image.

Public Member Functions inherited from [FI_RGB_Image](#)

- int [cache_h](#) ()
- int [cache_w](#) ()
- [FI_Image](#) * [copy](#) () const
- void [draw](#) (int X, int Y)
- [FI_RGB_Image](#) (const [FI_Pixmap](#) *pxm, [FI_Color](#) bg=[FL_GRAY](#))
The constructor creates a new RGBA image from the specified [FI_Pixmap](#).
- [FI_RGB_Image](#) (const [uchar](#) *bits, int bits_length, int W, int H, int D, int LD)
The constructor creates a new image from the specified data.
- [FI_RGB_Image](#) (const [uchar](#) *bits, int W, int H, int D=3, int LD=0)
The constructor creates a new image from the specified data.
- void [label](#) ([FI_Menu_Item](#) *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.
- void [label](#) ([FI_Widget](#) *w) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a widget or menu item.
- void [uncache](#) () [FL_OVERRIDE](#)
If the image has been cached for display, delete the cache data.
- ~[FI_RGB_Image](#) () [FL_OVERRIDE](#)
The destructor frees all memory and server resources that are used by the image.

Public Member Functions inherited from FI_Image

- virtual class FI_Shared_Image * **as_shared_image** ()
Returns whether an image is an FI_Shared_Image or not.
- FI_Image * **copy** () const
Creates a copy of the image in the same size.
- int **count** () const
Returns the number of data values associated with the image.
- int **d** () const
Returns the image depth.
- const char *const * **data** () const
Returns a pointer to the current image data array.
- int **data_h** () const
Returns the height of the image data.
- int **data_w** () const
Returns the width of the image data.
- void **draw** (int X, int Y)
Draws the image to the current drawing surface.
- int **fail** () const
Returns a value that is not 0 if there is currently no image available.
- FI_Image (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int **h** () const
Returns the current image drawing height in FLTK units.
- void **inactive** ()
*The **inactive()** method calls **color_average(FL_BACKGROUND_COLOR, 0.33f)** to produce an image that appears grayed out.*
- int **ld** () const
Returns the current line data size in bytes.
- virtual void **release** ()
Releases an FI_Image - the same as 'delete this'.
- virtual void **scale** (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- int **w** () const
Returns the current image drawing width in FLTK units.
- virtual ~FI_Image ()
The destructor is a virtual method that frees all memory used by the image.

Public Attributes

- bool **proportional**
*Set this to *false* to allow image re-scaling that alters the image aspect ratio.*

Public Attributes inherited from FI_RGB_Image

- int **alloc_array**
If non-zero, the object's data array is delete[]'d when deleting the object.
- const uchar * **array**
Points to the start of the object's data array.

Additional Inherited Members

Static Public Member Functions inherited from [FI_RGB_Image](#)

- static `size_t max_size ()`
Returns the maximum allowed image size in bytes when creating an [FI_RGB_Image](#) object.
- static void `max_size (size_t size)`
Sets the maximum allowed image size in bytes when creating an [FI_RGB_Image](#) object.

Static Public Member Functions inherited from [FI_Image](#)

- static `FI_Labeltype define_FL_IMAGE_LABEL ()`
- static `FI_RGB_Scaling RGB_scaling ()`
Returns the currently used RGB image scaling method.
- static void `RGB_scaling (FI_RGB_Scaling)`
Sets the RGB image scaling method used for copy(int, int).
- static `FI_RGB_Scaling scaling_algorithm ()`
Gets what algorithm is used when resizing a source image to draw it.
- static void `scaling_algorithm (FI_RGB_Scaling algorithm)`
Sets what algorithm is used when resizing a source image to draw it.

Static Public Attributes inherited from [FI_Image](#)

- static const int `ERR_FILE_ACCESS = -2`
- static const int `ERR_FORMAT = -3`
- static const int `ERR_MEMORY_ACCESS = -4`
- static const int `ERR_NO_IMAGE = -1`
- static bool `register_images_done = false`
True after [fl_register_images\(\)](#) was called, false before.

Protected Member Functions inherited from [FI_Image](#)

- void `d (int D)`
Sets the current image depth.
- void `data (const char *const *p, int c)`
Sets the current data pointer and count of pointers in the array.
- void `draw_empty (int X, int Y)`
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int `draw_scaled (int X, int Y, int W, int H)`
Draw the image to the current drawing surface rescaled to a given width and height.
- void `h (int H)`
Sets the height of the image data.
- void `ld (int LD)`
Sets the current line data size in bytes.
- void `w (int W)`
Sets the width of the image data.

Static Protected Member Functions inherited from [FI_Image](#)

- static void `labeltype (const FI\_Label *lo, int lx, int ly, int lw, int lh, FI\_Align la)`
- static void `measure (const FI\_Label *lo, int &lw, int &lh)`

11.137.1 Detailed Description

The `Fl_SVG_Image` class supports loading, caching and drawing of scalable vector graphics (SVG) images.

The FLTK library performs parsing and rasterization of SVG data using a modified version of the `nanosvg` software (<https://github.com/memononen/nanosvg>). The software modification allows the option to change the image ratio while performing rasterization.

Use `Fl_Image::fail()` to check if the `Fl_SVG_Image` failed to load. `fail()` returns `ERR_FILE_ACCESS` if the file could not be opened or read, and `ERR_FORMAT` if the SVG format could not be decoded. If the image has loaded correctly, `w()`, `h()`, and `d()` should return values greater than zero.

Rasterization is not done until the image is first drawn or `resize()` or `normalize()` is called. Therefore, `array` is NULL until then. The delayed rasterization ensures an `Fl_SVG_Image` is always rasterized to the exact screen resolution at which it is drawn.

The `Fl_SVG_Image` class draws images computed by `nanosvg` with the following known limitations

- text between `<text>` and `</text>` marks,
- image elements, and
- `<use>` statements

are not rendered.

The FLTK library can optionally be built without SVG support; in that case, class `Fl_SVG_Image` is unavailable.

Example of displaying a hard-coded svg file:

```
#include <FL/Fl.H>
#include <FL/Fl_Window.H>
#include <FL/Fl_Box.H>
#include <FL/Fl_SVG_Image.H>

// A black rotated rectangle
const char *svg_data = "<svg viewBox=\"0 0 200 200\" version = \"1.1\">\n"
    "<rect x=\"25\" y=\"50\" width=\"150\" height=\"100\" fill=\"black\" "
    "transform=\"rotate(45 100 100)\"> </svg>\n";

int main(int argc, char **argv) {
    Fl_SVG_Image *svg = new Fl_SVG_Image(0, svg_data); // create SVG object
    Fl_Window *win = new Fl_Window(720, 486, "svg test");
    Fl_Box *box = new Fl_Box(0, 0, win->w(), win->h());
    box->image(svg); // assign svg object to Fl_Box
    win->end();
    win->show(argc, argv);
    return(Fl::run());
}
```

Example of displaying an svg image from a file:

```
#include <errno.h> // errno
#include <string.h> // strerror
#include <FL/Fl.H>
#include <FL/Fl_Window.H>
#include <FL/Fl_Box.H>
#include <FL/Fl_SVG_Image.H>
#include <FL/fl_message.H>

int main(int argc, char **argv) {
    Fl_Window *win = new Fl_Window(720, 486, "svg test");
    Fl_Box *box = new Fl_Box(0, 0, win->w(), win->h());

    // Load svg image from disk, assign to a box
    const char *svgpath = "/var/tmp/simple.svg";
    Fl_SVG_Image *svg = new Fl_SVG_Image(svgpath); // load SVG object from disk
    switch (svg->fail()) {
        case Fl_Image::ERR_FILE_ACCESS:
            // File couldn't load? show path + os error to user
            fl_alert("%s: %s", svgpath, strerror(errno));
            return 1;
        case Fl_Image::ERR_FORMAT:
            // Parsing error
            fl_alert("%s: couldn't decode image", svgpath);
            return 1;
    }
    box->image(svg); // assign svg object to box

    win->end();
    win->show(argc, argv);
    return(Fl::run());
}
```

Example of fitting an svg image to a resizable `Fl_Box`:

```
#include <FL/Fl_Window.H>
#include <FL/Fl_SVG_Image.H>
#include <FL/Fl_Box.H>

class resizable_box : public Fl_Box {
```

```

public:
    resizable_box(int w, int h) : Fl_Box(0, 0, w, h, NULL) {}
    virtual void resize(int x, int y, int w, int h) {
        image()->scale(w, h, 1, 1); // p3 = proportional, p4 = can_expand
        Fl_Box::resize(x, y, w, h);
    }
};

int main(int argc, char **argv) {
    Fl_Window *win = new Fl_Window(130, 130);
    resizable_box *box = new resizable_box(win->w(), win->h());
    Fl_SVG_Image *svg = new Fl_SVG_Image("/path/to/image.svg");
    box->image(svg);
    svg->scale(box->w(), box->h());
    win->end();
    win->resizable(win);
    win->show(argc, argv);
    return Fl::run();
}

```

11.137.2 Constructor & Destructor Documentation

11.137.2.1 Fl_SVG_Image() [1/3]

```

Fl_SVG_Image::Fl_SVG_Image (
    const char * filename)

```

Load an SVG image from a file.

This constructor loads the SVG image from a .svg or .svgz file. The reader recognizes if the data is compressed, and decompresses it if zlib is available (HAVE_LIBZ).

Parameters

<i>filename</i>	the filename for a .svg or .svgz file

11.137.2.2 Fl_SVG_Image() [2/3]

```

Fl_SVG_Image::Fl_SVG_Image (
    const char * sharedname,
    const char * svg_data)

```

Load an SVG image from memory.

This constructor loads the SVG image from a block of memory. This version is commonly used for uncompressed text data, but the reader recognizes if the data is compressed, and decompresses it if zlib is available (HAVE_LIBZ).

Parameters

<i>sharedname</i>	if not NULL, a shared image will be generated with this name
<i>svg_data</i>	a pointer to the memory location of the SVG image data

Note

In-memory SVG data is parsed by the object constructor and is no longer needed after construction.

11.137.2.3 Fl_SVG_Image() [3/3]

```

Fl_SVG_Image::Fl_SVG_Image (
    const char * name,
    const unsigned char * svg_data,
    size_t length)

```

Load an SVG image from memory.

This constructor loads the SVG image from a block of memory. This version is commonly used for compressed binary data, but the reader recognizes if the data is uncompressed, and reads it as a text block.

Parameters

<i>name</i>	if not NULL, a shared image will be generated with this name
<i>svg_data</i>	a pointer to the memory location of the SVG image data
<i>length</i>	of <i>svg_data</i> or 0 if the length is unknown. This will protect memory outside of the <i>svg_data</i> array from illegal read operations for compressed SVG data

Note

In-memory SVG data is parsed by the object constructor and is no longer needed after construction.

11.137.3 Member Function Documentation

11.137.3.1 as_svg_image()

```
Fl_SVG_Image * Fl_SVG_Image::as_svg_image () [inline], [virtual]
```

Returns whether an image is an [Fl_SVG_Image](#) or not.

This virtual method returns a pointer to the [Fl_SVG_Image](#) if this object is an instance of [Fl_SVG_Image](#) or NULL if not.

Reimplemented from [Fl_RGB_Image](#).

11.137.3.2 color_average()

```
void Fl_SVG_Image::color_average (
    Fl_Color c,
    float i) [virtual]
```

The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.

The first argument specifies the FLTK color to be used.

The second argument specifies the amount of the original image to combine with the color, so a value of 1.0 results in no color blend, and a value of 0.0 results in a constant image of the specified color.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented from [Fl_RGB_Image](#).

11.137.3.3 copy()

```
Fl_Image * Fl_SVG_Image::copy (
    int W,
    int H) const [virtual]
```

Creates a resized copy of the image.

It is recommended not to call this member function to reduce the size of an image to the size of the area where this image will be drawn, and to use [Fl_Image::scale\(\)](#) instead.

The new image should be released when you are done with it.

Note: since FLTK 1.4.0 you can use [Fl_Image::release\(\)](#) for all types of images (i.e. all subclasses of [Fl_Image](#)) instead of operator *delete* for [Fl_Image](#)'s and [Fl_Image::release\(\)](#) for [Fl_Shared_Image](#)'s.

The new image data will be converted to the requested size. RGB images are resized using the algorithm set by [Fl_Image::RGB_scaling\(\)](#).

For the new image the following equations are true:

- $w() == data\_w() == W$
- $h() == data\_h() == H$

Parameters

in	<i>W,H</i>	Requested width and height of the new image

Note

The returned image can be safely cast to the same image type as that of the source image provided this type is one of [Fl_RGB_Image](#), [Fl_SVG_Image](#), [Fl_Pixmap](#), [Fl_Bitmap](#), [Fl_Tiled_Image](#), [Fl_Anim_GIF_Image](#) and [Fl_Shared_Image](#). Returned objects copied from images of other, derived, image classes belong to the parent class appearing in this list. For example, the copy of an [Fl_GIF_Image](#) is an object of class [Fl_Pixmap](#).

Since FLTK 1.4.0 this method is 'const'. If you derive your own class from [Fl_Image](#) or any subclass your overridden methods of '[Fl_Image::copy\(\) const](#)' and '[Fl_Image::copy\(int, int\) const](#)' **must** also be 'const' for inheritance to work properly. This is different than in FLTK 1.3.x and earlier where these methods have not been 'const'.

Reimplemented from [Fl_RGB_Image](#).

11.137.3.4 desaturate()

```
void Fl_SVG_Image::desaturate () [virtual]
```

The [desaturate\(\)](#) method converts an image to grayscale.

If the image contains an alpha channel (depth = 4), the alpha channel is preserved.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented from [Fl_RGB_Image](#).

11.137.3.5 draw()

```
void Fl_SVG_Image::draw (
    int X,
    int Y,
    int W,
    int H,
    int cx = 0,
    int cy = 0) [virtual]
```

Draws the image to the current drawing surface with a bounding box.

Arguments X, Y, W, H specify a bounding box for the image, with the origin (upper-left corner) of the image offset by the cx and cy arguments.

In other words: `fl_push_clip(X,Y,W,H)` is applied, the image is drawn with its upper-left corner at X-cx, Y-cy and its own width and height, `fl_pop_clip()` is applied.

Reimplemented from [Fl_RGB_Image](#).

11.137.3.6 normalize()

```
void Fl_SVG_Image::normalize () [virtual]
```

Makes sure the object is fully initialized.

This function rasterizes the SVG image, and consequently initializes its [array](#) member, if that was not done before.

Reimplemented from [Fl_RGB_Image](#).

11.137.3.7 resize()

```
void Fl_SVG_Image::resize (
    int width,
    int height)
```

Have the svg data (re-)rasterized using the given width and height values.

By default, the resulting image w() and h() will be close to width and height while preserving the width/height ratio of the SVG data. If [proportional](#) was set to false, the image is rasterized to the exact width and height values. In both cases, [data_w\(\)](#) and [data_h\(\)](#) values are set to w() and h(), respectively.

11.137.4 Member Data Documentation**11.137.4.1 proportional**

```
bool Fl_SVG_Image::proportional
```

Set this to `false` to allow image re-scaling that alters the image aspect ratio.

Upon object creation, `proportional` is set to `true`, and the aspect ratio is kept constant.

The documentation for this class was generated from the following files:

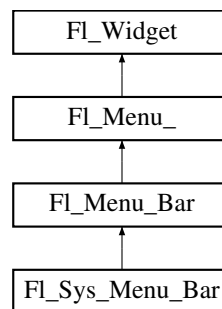
- `Fl_SVG_Image.H`
- `Fl_SVG_Image.cxx`

11.138 Fl_Sys_Menu_Bar Class Reference

A class to create and modify menus that appear on macOS in the menu bar at the top of the screen.

```
#include <Fl_Sys_Menu_Bar.H>
```

Inheritance diagram for `Fl_Sys_Menu_Bar`:



Public Types

- enum `window_menu_style_enum` { `no_window_menu` = 0, `tabbing_mode_none`, `tabbing_mode_automatic`, `tabbing_mode_preferred` }

Possible styles of the Window menu in the system menu bar.

Public Member Functions

- `int add` (const char *label, const char *shortcut, `Fl_Callback` *cb, void *user_data=0, int flags=0)
Adds a new menu item.
- `int add` (const char *label, int shortcut, `Fl_Callback` *, void *user_data=0, int flags=0)
Add a new menu item to the system menu bar.
- `int add` (const char *str)
Forms-compatible procedure to add items to the system menu bar.
- `void clear` ()
Set the `Fl_Menu_Item` array pointer to null, indicating a zero-length menu.
- `int clear_submenu` (int index)
Clears the specified submenu pointed to by index of all menu items.
- `Fl_Sys_Menu_Bar` (int x, int y, int w, int h, const char *l=0)
The constructor.
- `int insert` (int index, const char *label, const char *shortcut, `Fl_Callback` *cb, void *user_data=0, int flags=0)
Insert a new menu item.
- `int insert` (int index, const char *label, int shortcut, `Fl_Callback` *cb, void *user_data=0, int flags=0)
insert in the system menu bar a new menu item
- `const Fl_Menu_Item * menu` () const
Return the system menu's array of `Fl_Menu_Item`'s.
- `void menu` (const `Fl_Menu_Item` *m)
create a system menu bar using the given list of menu structs
- `int mode` (int i) const
Gets the flags of item i.

- void **mode** (int i, int fl)
Sets the flags of item i.
- void **play_menu** (const **FI_Menu_Item** *) **FL_OVERRIDE**
Opens the 1st level submenu of the menubar corresponding to item.
- void **remove** (int n)
remove an item from the system menu bar
- void **replace** (int index, const char *name)
rename an item from the system menu bar
- void **setonly** (**FI_Menu_Item** *item)
Turns the radio item "on" for the menu item and turns "off" adjacent radio items of the same group.
- void **shortcut** (int i, int s)
Changes the shortcut of item i to n.
- void **update** () **FL_OVERRIDE**
Updates the menu bar after any change to its items.
- virtual ~**FI_Sys_Menu_Bar** ()
The destructor.

Public Member Functions inherited from **FI_Menu_Bar**

- **FI_Menu_Bar** (int X, int Y, int W, int H, const char *l=0)
*Creates a new **FI_Menu_Bar** widget using the given position, size, and label string.*
- int **handle** (int) **FL_OVERRIDE**
Handles the specified event.

Public Member Functions inherited from **FI_Menu_**

- int **add** (const char *)
This is a Forms (and SGI GL library) compatible add function, it adds many menu items, with '|' separating the menu items, and tab separating the menu item names from an optional shortcut string.
- int **add** (const char *, int **shortcut**, **FI_Callback** *, void *d=0, int e=0)
Adds a new menu item.
- int **add** (const char *a, const char *b, **FI_Callback** *c, void *d=0, int e=0)
*See int **FI_Menu_::add**(const char* label, int shortcut, **FI_Callback***, void *user_data=0, int flags=0)*
- void **clear** ()
Same as menu(NULL), set the array pointer to null, indicating a zero-length menu.
- int **clear_submenu** (int index)
Clears the specified submenu pointed to by index of all menu items.
- void **copy** (const **FI_Menu_Item** *m, void *user_data=0)
Sets the menu array pointer with a copy of m that will be automatically deleted.
- **FI_Boxtype** **down_box** () const
This box type is used to surround the currently-selected items in the menus.
- void **down_box** (**FI_Boxtype** b)
Sets the box type used to surround the currently-selected items in the menus.
- **FI_Color** **down_color** () const
*For back compatibility, same as **selection_color**()*
- void **down_color** (unsigned c)
*For back compatibility, same as **selection_color**()*
- int **find_index** (const char *name) const
Find the menu item index for a given menu pathname, such as "Edit/Copy".
- int **find_index** (const **FI_Menu_Item** *item) const
Find the index into the menu array for a given item.
- int **find_index** (**FI_Callback** *cb) const

- Find the index into the menu array for a given callback `cb`.*

 - const `Fl_Menu_Item * find_item` (const char *name)
- Find the menu item for a given menu `pathname`, such as "Edit/Copy".*

 - const `Fl_Menu_Item * find_item` (Fl_Callback *)
- Find the menu item for the given callback `cb`.*

 - const `Fl_Menu_Item * find_item_with_argument` (long)
- Find the menu item for the given user argument `v`.*

 - const `Fl_Menu_Item * find_item_with_user_data` (void *)
- Find the menu item for the given user data `v`.*

 - `Fl_Menu_` (int, int, int, int, const char **=0)
- Creates a new `Fl_Menu_` widget using the given position, size, and label string.*

 - void `global` ()
- Make the shortcuts for this menu work no matter what window has the focus when you type it.*

 - int `insert` (int index, const char *, int `shortcut`, Fl_Callback *, void **=0, int=0)
- Inserts a new menu item at the specified `index` position.*

 - int `insert` (int index, const char *a, const char *b, Fl_Callback *c, void *d=0, int e=0)
- See int `Fl_Menu_::insert`(const char* label, int shortcut, Fl_Callback*, void *user_data=0, int flags=0)*

 - int `item_pathname` (char *name, int namelen, const `Fl_Menu_Item` *finditem=0) const
- Get the menu 'pathname' for the specified menuitem.*

 - const `Fl_Menu_Item * menu` () const
- Returns a pointer to the array of `Fl_Menu_Items`.*

 - void `menu` (const `Fl_Menu_Item` *m)
- Sets the menu array pointer directly.*

 - `Fl_Boxtype menu_box` () const
- Get the box type for the menu popup windows.*

 - void `menu_box` (Fl_Boxtype b)
- Set the box type for the menu popup windows.*

 - const `Fl_Menu_Item * menu_end` ()
- Finishes menu modifications and returns `menu()`.*

 - int `mode` (int i) const
- Get the flags of item `i`.*

 - void `mode` (int i, int fl)
- Set the flags of item `i`.*

 - const `Fl_Menu_Item * mvalue` () const
- Return a pointer to the last menu item that was picked.*

 - const `Fl_Menu_Item * picked` (const `Fl_Menu_Item` *)
- When user picks a menu item, call this.*

 - const `Fl_Menu_Item * prev_mvalue` () const
- Return a pointer to the menu item that was picked before the current one was picked.*

 - void `remove` (int)
- Deletes item `i` from the menu.*

 - void `replace` (int, const char *)
- Changes the text of item `i`.*

 - void `setonly` (Fl_Menu_Item *item)
- Turns the radio item "on" for the menu item and turns "off" adjacent radio items of the same group.*

 - void `shortcut` (int i, int s)
- Change the shortcut of item `i` to `s`.*

 - int `size` () const
- This returns the number of `Fl_Menu_Item` structures that make up the menu, correctly counting submenus.*

 - void `size` (int W, int H)
- const `Fl_Menu_Item * test_shortcut` ()

- Returns the menu item with the entered shortcut (key value).*

 - `const char * text () const`

Returns the title of the last item chosen.
- `const char * text (int i) const`

*Returns the title of item *i*.*
- `Fl\_Color textcolor () const`

Get the current color of menu item labels.
- `void textcolor (Fl\_Color c)`

Sets the current color of menu item labels.
- `Fl\_Font textfont () const`

Gets the current font of menu item labels.
- `void textfont (Fl\_Font c)`

Sets the current font of menu item labels.
- `Fl\_Fontsize textsize () const`

Gets the font size of menu item labels.
- `void textsize (Fl\_Fontsize c)`

Sets the font size of menu item labels.
- `int value () const`

Return the index into the [menu\(\)](#) of the last item chosen by the user.
- `int value (const Fl\_Menu\_Item *)`

*Set the value of a menu to the menu item *m*.*
- `int value (int i)`

*Set the value of the menu to index *i*.*

Public Member Functions inherited from [Fl_Widget](#)

- `void \_clear\_fullscreen ()`
- `void \_set\_fullscreen ()`
- `void activate ()`

Activates the widget.
- `unsigned int active () const`

Returns whether the widget is active.
- `int active\_r () const`

Returns whether the widget and all of its parents are active.
- `Fl\_Align align () const`

Gets the label alignment.
- `void align (Fl\_Align alignment)`

Sets the label alignment.
- `long argument () const`

Gets the current user data (long) argument that is passed to the callback function.
- `void argument (long v)`

Sets the current user data (long) argument that is passed to the callback function.
- `virtual class Fl\_Gl\_Window * as\_gl\_window ()`

Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- `virtual class Fl\_Gl\_Window const * as\_gl\_window () const`
- `virtual Fl\_Group * as\_group ()`

Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).
- `virtual Fl\_Group const * as\_group () const`
- `virtual Fl\_Window * as\_window ()`

Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- `virtual Fl\_Window const * as\_window () const`

- void [bind_deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([Fl_Boxtype](#) new_box)
Sets the box type for the widget.
- [Fl_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([Fl_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) ([uchar](#) c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [Fl_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([Fl_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([Fl_Color](#) bg, [Fl_Color](#) sel)
Sets the background and selection color of the widget.
- [Fl_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [Fl_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)

- Sets the current label.*

 - void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.
- void `do_callback` (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with arbitrary arguments.
- void `do_callback` (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, FL_Align) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const

Gets the widget height.
- virtual void `hide` ()

Makes a widget invisible.
- int `horizontal_label_margin` ()

Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)

Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image` ()

Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image * image` () const

Gets the image that is used as part of the widget label when in the active state.
- void `image` (FL_Image &img)

Sets the image to use as part of the widget label when in the active state.
- void `image` (FL_Image *img)

Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const

Returns whether the image is managed by the widget.
- int `inside` (const FL_Widget *wgt) const

Checks if this widget is a child of wgt.

- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (Fl_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `Fl_Color labelcolor () const`
Gets the label color.
- `void labelcolor (Fl_Color c)`
Sets the label color.
- `Fl_Font labelfont () const`
Gets the font to use.
- `void labelfont (Fl_Font f)`
Sets the font to use.
- `Fl_Fontsize labelsz () const`
Gets the font size in pixels.
- `void labelsz (Fl_Fontsize pix)`
Sets the font size in pixels.
- `Fl_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (Fl_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`
Sets width ww and height hh accordingly with the label size.
- `bool needs_keyboard () const`
Returns whether this widget needs a keyboard.
- `void needs_keyboard (bool needs)`
Sets whether this widget needs a keyboard.
- `unsigned int output () const`
Returns if a widget is used for output only.
- `Fl_Group * parent () const`
Returns a pointer to the parent widget.
- `void parent (Fl_Group *p)`
Internal use only - "for hacks only".
- `void position (int X, int Y)`
Repositions the window or widget.
- `void redraw ()`
Schedules the drawing of the widget.
- `void redraw_label ()`
Schedules the drawing of the label.
- `virtual void resize (int x, int y, int w, int h)`
Changes the size or position of the widget.
- `Fl_Color selection_color () const`
Gets the selection color.
- `void selection_color (Fl_Color a)`

- Sets the selection color.*

 - void `set_active` ()
- Marks the widget as active without sending events or changing focus.*

 - void `set_changed` ()
- Marks the value of the widget as changed.*

 - void `set_output` ()
- Sets a widget to output only.*

 - void `set_visible` ()
- Makes the widget visible.*

 - void `set_visible_focus` ()
- Enables keyboard focus navigation with this widget.*

 - int `shortcut_label` () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void `shortcut_label` (int value)
- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - virtual void `show` ()
- Makes a widget visible.*

 - void `size` (int W, int H)
- Changes the size of the widget.*

 - int `take_focus` ()
- Gives the widget the keyboard focus.*

 - unsigned int `takeevents` () const
- Returns if the widget is able to take events.*

 - int `test_shortcut` ()
- Returns true if the widget's label contains the entered '&x' shortcut.*

 - const char * `tooltip` () const
- Gets the current tooltip text.*

 - void `tooltip` (const char *text)
- Sets the current tooltip text.*

 - `Fl_Window` * `top_window` () const
- Returns a pointer to the top-level window for the widget.*

 - `Fl_Window` * `top_window_offset` (int &xoff, int &yoff) const
- Finds the x/y offset of the current widget relative to the top-level window.*

 - `uchar` `type` () const
- Gets the widget type.*

 - void `type` (`uchar` t)
- Sets the widget type.*

 - int `use_accents_menu` ()
- Returns non zero if `MAC_USE_ACCENTS_MENU` flag is set, 0 otherwise.*

 - void * `user_data` () const
- Gets the user data for this widget.*

 - void `user_data` (`Fl_Callback_User_Data` *v, bool auto_free)
- Sets the user data for this widget.*

 - void `user_data` (void *v)
- Sets the user data for this widget.*

 - int `vertical_label_margin` ()
- Get the spacing between the label and the vertical edge of the widget.*

 - void `vertical_label_margin` (int px)
- Set the spacing between the label and the vertical edge of the widget.*

 - unsigned int `visible` () const
- Returns whether a widget is visible.*

- unsigned int `visible_focus` () const
Checks whether this widget has a visible focus.
- void `visible_focus` (int v)
Modifies keyboard focus navigation.
- int `visible_r` () const
Returns whether a widget and all its parents are visible.
- int `w` () const
Gets the widget width.
- `FI_When` when () const
Returns the conditions under which the callback is called.
- void `when` (uchar i)
Sets the flags used to decide when a callback is called.
- `FI_Window` * `window` () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual `~FI_Widget` ()
Destroys the widget.

Static Public Member Functions

- static void `about` (`FI_Callback` *cb, void *data)
Attaches a callback to the "About myprog" item of the system application menu.
- static void `create_window_menu` ()
Adds a Window menu, to the end of the system menu bar.
- static `window_menu_style_enum` `window_menu_style` ()
Get the style of the Window menu in the system menu bar.
- static void `window_menu_style` (`window_menu_style_enum` style)
Set the desired style of the Window menu in the system menu bar.

Static Public Member Functions inherited from `FI_Widget`

- static void `default_callback` (`FI_Widget` *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Member Functions

- void `draw` () `FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from `FI_Menu_Bar`

Protected Member Functions inherited from `FI_Menu_`

- int `item_pathname` (char *name, int namelen, const `FI_Menu_Item` *finditem, const `FI_Menu_Item` *menu=0) const

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 , [NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 , [MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

Protected Attributes inherited from [Fl_Menu_](#)

- [uchar](#) `alloc`
- [uchar](#) `down_box_`
- [Fl_Boxtype](#) `menu_box_`
- [Fl_Color](#) `textcolor_`
- [Fl_Font](#) `textfont_`
- [Fl_Fontsize](#) `textsize_`

11.138.1 Detailed Description

A class to create and modify menus that appear on macOS in the menu bar at the top of the screen.

On other than macOS platforms, [Fl_Sys_Menu_Bar](#) is a synonym of class [Fl_Menu_Bar](#).

On the macOS platform, replace [Fl_Menu_Bar](#) with [Fl_Sys_Menu_Bar](#), and a system menu at the top of the screen will be available. This menu will match an array of [Fl_Menu_Item](#)'s exactly as in all other FLTK menus (except for the submenu with the application's own name and the 'Window' menu; see below). There is, though, an important difference between an [Fl_Sys_Menu_Bar](#) object under macOS and under other platforms: only a single object from this class can be created, because macOS uses a single system menu bar. Therefore, porting to macOS an app that creates, on other platforms, several [Fl_Menu_Bar](#) objects, one for each of several windows, is more complex than just replacing [Fl_Menu_Bar](#) by [Fl_Sys_Menu_Bar](#).

On the macOS platform, the system menu bar of any FLTK app begins with the Application menu which the FLTK library automatically constructs. Functions [Fl_Mac_App_Menu::custom_application_menu_items\(\)](#) and [Fl_Sys_Menu_Bar::about\(\)](#) can be used to further customize the Application menu. The FLTK library also automatically constructs and handles a Window menu which can be further customized (or even removed) calling [Fl_Sys_Menu_Bar::window_menu_style\(window_menu_style_enum style\)](#). Other member functions of this class allow the app to generate the rest of the system menu bar. It is recommended to localize the system menu bar using the standard Mac OS X localization procedure (see [Internationalization](#)).

Changes to the menu state are immediately visible in the menubar when they are made using member functions of the [Fl_Sys_Menu_Bar](#) class. Other changes (e.g., by a call to [Fl_Menu_Item::set\(\)](#)) should be followed by a call to [update\(\)](#) to be visible in the menubar across all platforms. macOS global variable `fl_sys_menu_bar` points to the unique, current system menu bar.

A few FLTK menu features are not supported by the Mac System menu:

- no symbolic labels
- no embossed labels
- no font sizes

As described above, the submenu with the application's own name (usually the second submenu from the left, immediately following the "Apple" submenu) is a special case, and can be managed with [Fl_Mac_App_Menu::custom_application_menu_items\(\)](#). For example, to make your own "Appname -> Preferences" dialog, you might use:

```
#include <FL/platform.H> // for Fl_Mac_App_Menu class
#include <FL/Fl_Sys_Menu_Bar.H> // for Fl_Menu_Item
:
void prefs_cb(Fl_Widget *w, void *data) {
    // ..Open your preferences dialog here..
}
:
int main(..) {
    :
    // Items to add to the application menu
    static Fl_Menu_Item appitems[] = {
        { "Preferences", 0, prefs_cb, 0, 0 },
        { 0 }, { 0 }
    };
    Fl_Mac_App_Menu::custom_application_menu_items(appitems); // adds it
}
```

..the result being:

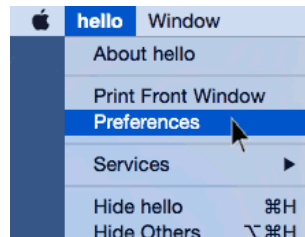


Figure 11.46 Mac Application submenu

11.138.2 Member Enumeration Documentation

11.138.2.1 window_menu_style_enum

enum [Fl_Sys_Menu_Bar::window_menu_style_enum](#)

Possible styles of the Window menu in the system menu bar.

Enumerator

<code>no_window_menu</code>	No Window menu in the system menu bar.
<code>tabbing_mode_none</code>	No tabbed windows, but the system menu bar contains a Window menu.
<code>tabbing_mode_automatic</code>	Windows are created by themselves but can be tabbed later.
<code>tabbing_mode_preferred</code>	Windows are tabbed when created.

11.138.3 Constructor & Destructor Documentation

11.138.3.1 Fl_Sys_Menu_Bar()

```
Fl_Sys_Menu_Bar::Fl_Sys_Menu_Bar (
    int x,
    int y,
    int w,
    int h,
    const char * l = 0)
```

The constructor.

On Mac OS X, all arguments are unused. On other platforms they are used as by [Fl_Menu_Bar::Fl_Menu_Bar\(\)](#).

11.138.4 Member Function Documentation

11.138.4.1 about()

```
void Fl_Sys_Menu_Bar::about (
    Fl\_Callback * cb,
    void * data) [static]
```

Attaches a callback to the "About myprog" item of the system application menu.

This cross-platform function is effective only under the MacOS platform.

Parameters

<i>cb</i>	a callback that will be called by "About myprog" menu item with NULL 1st argument.
<i>data</i>	a pointer transmitted as 2nd argument to the callback.

11.138.4.2 add() [1/3]

```
int Fl_Sys_Menu_Bar::add (
    const char * label,
```



```
const char * shortcut,
Fl_Callback * cb,
void * user_data = 0,
int flags = 0) [inline]
```

Adds a new menu item.

See also

[Fl_Menu_::add](#)(const char* [label](#), int [shortcut](#), [Fl_Callback*](#), void *[user_data](#)=0, int [flags](#)=0)

11.138.4.3 add() [2/3]

```
int Fl_Sys_Menu_Bar::add (
    const char * label,
    int shortcut,
    Fl_Callback * cb,
    void * user_data = 0,
    int flags = 0)
```

Add a new menu item to the system menu bar.

Add to the system menu bar a new menu item, with a title string, shortcut int, callback, argument to the callback, and flags.

Parameters

<i>label</i>	- new menu item's label
<i>shortcut</i>	- new menu item's integer shortcut (can be 0 for none, or e.g. <code>FL_ALT+'x'</code>)
<i>cb</i>	- callback to be invoked when item selected (can be 0 for none, in which case the menubar's callback() can be used instead)
<i>user_data</i>	- argument to the callback
<i>flags</i>	- item's flags, e.g. FL_MENU_TOGGLE , etc.

Returns

the index into the [menu\(\)](#) array, where the entry was added

See also

[Fl_Menu_::add](#)(const char* [label](#), int [shortcut](#), [Fl_Callback](#) *[cb](#), void *[user_data](#), int [flags](#))

11.138.4.4 add() [3/3]

```
int Fl_Sys_Menu_Bar::add (
    const char * str)
```

Forms-compatible procedure to add items to the system menu bar.

Returns

the index into the [menu\(\)](#) array, where the entry was added

See also

[Fl_Menu_::add](#)(const char* [str](#))

11.138.4.5 clear()

```
void Fl_Sys_Menu_Bar::clear ()
```

Set the [Fl_Menu_Item](#) array pointer to null, indicating a zero-length menu.

See also

[Fl_Menu_::clear](#)()

11.138.4.6 clear_submenu()

```
int Fl_Sys_Menu_Bar::clear_submenu (
    int index)
```

Clears the specified submenu pointed to by index of all menu items.

See also

[Fl_Menu_::clear_submenu\(int index\)](#)

11.138.4.7 create_window_menu()

```
void Fl_Sys_Menu_Bar::create_window_menu () [static]
```

Adds a Window menu, to the end of the system menu bar.

FLTK apps typically don't need to call this function which is automatically called by the library the first time a window is shown. The default system menu bar contains a Window menu with a "Merge All Windows" item. Other Window menu styles can be obtained calling [Fl_Sys_Menu_Bar::window_menu_style\(window_menu_style_enum\)](#) before the first [Fl_Window::show\(\)](#). Alternatively, an app can call [create_window_menu\(\)](#) after having populated the system menu bar, for example with [menu\(const Fl_Menu_Item *\)](#), and before the first [Fl_Window::show\(\)](#).

This function does nothing on non MacOS platforms.

Version

1.4

11.138.4.8 draw()

```
void Fl_Sys_Menu_Bar::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Reimplemented from [Fl_Menu_Bar](#).

11.138.4.9 insert() [1/2]

```
int Fl_Sys_Menu_Bar::insert (
    int index,
    const char * label,
    const char * shortcut,
    Fl_Callback * cb,
    void * user_data = 0,
    int flags = 0) [inline]
```

Insert a new menu item.

See also

[Fl_Menu_::insert\(int index, const char* label, const char* shortcut, Fl_Callback *cb, void *user_data=0, int flags=0\)](#)

11.138.4.10 insert() [2/2]

```
int Fl_Sys_Menu_Bar::insert (
    int index,
    const char * label,
    int shortcut,
    Fl_Callback * cb,
```

```
void * user_data = 0,
int flags = 0)
```

insert in the system menu bar a new menu item

Insert in the system menu bar a new menu item, with a title string, shortcut int, callback, argument to the callback, and flags.

Returns

the index into the [menu\(\)](#) array, where the entry was inserted

See also

[Fl_Menu_::insert\(int index, const char* label, int shortcut, Fl_Callback *cb, void *user_data, int flags\)](#)

11.138.4.11 menu()

```
void Fl_Sys_Menu_Bar::menu (
    const Fl_Menu_Item * m)
```

create a system menu bar using the given list of menu structs

Author

Matthias Melcher

Parameters

<i>m</i>	Zero-ending list of Fl_Menu_Item 's

11.138.4.12 mode()

```
void Fl_Sys_Menu_Bar::mode (
    int i,
    int fl)
```

Sets the flags of item i.

See also

[Fl_Menu_::mode\(int i, int fl\)](#)

11.138.4.13 play_menu()

```
void Fl_Sys_Menu_Bar::play_menu (
    const Fl_Menu_Item * item) [virtual]
```

Opens the 1st level submenu of the menubar corresponding to item.

Since

1.4.0

Reimplemented from [Fl_Menu_Bar](#).

11.138.4.14 remove()

```
void Fl_Sys_Menu_Bar::remove (
    int index)
```

remove an item from the system menu bar

Parameters

<i>index</i>	the index of the item to remove

11.138.4.15 replace()

```
void Fl_Sys_Menu_Bar::replace (
    int index,
    const char * name)
```

rename an item from the system menu bar

Parameters

<i>index</i>	the index of the item to rename
<i>name</i>	the new item name as a UTF8 string

11.138.4.16 update()

```
void Fl_Sys_Menu_Bar::update () [virtual]
```

Updates the menu bar after any change to its items.

This is useful when the menu bar can be an [Fl_Sys_Menu_Bar](#) object.

Reimplemented from [Fl_Menu_Bar](#).

11.138.4.17 window_menu_style()

```
void Fl_Sys_Menu_Bar::window_menu_style (
    Fl_Sys_Menu_Bar::window_menu_style_enum style) [static]
```

Set the desired style of the Window menu in the system menu bar.

This function, to be called before the first call to [Fl_Window::show\(\)](#), allows to control whether the system menu bar should contain a Window menu, and if yes, whether new windows should be displayed in tabbed form. These are the effects of various values for *style* :

- `no_window_menu` : don't add a Window menu to the system menu bar
- `tabbing_mode_none` : add a simple Window menu to the system menu bar
- `tabbing_mode_automatic` : the window menu also contains "Merge All Windows" to group all windows in a single tabbed display mode. This is the **default** Window menu style for FLTK apps.
- `tabbing_mode_preferred` : new windows are displayed in tabbed mode when first created

The Window menu, if present, is entirely created and controlled by the FLTK library. Mac OS version 10.12 or later must be running for windows to be displayed in tabbed form. Under non MacOS platforms, this function does nothing.

Version

1.4

The documentation for this class was generated from the following files:

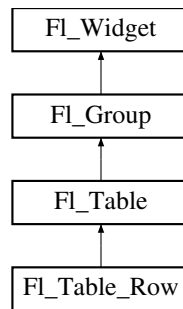
- [Fl_Sys_Menu_Bar.H](#)
- [Fl_Sys_Menu_Bar.cxx](#)

11.139 FI_Table Class Reference

A table of widgets or other content.

```
#include <Fl_Table.H>
```

Inheritance diagram for [Fl_Table](#):



Public Types

- enum `TableContext` {
`CONTEXT_NONE` = 0 , `CONTEXT_STARTPAGE` = 0x01 , `CONTEXT_ENDPAGE` = 0x02 , `CONTEXT_ROW_HEADER`
= 0x04 ,
`CONTEXT_COL_HEADER` = 0x08 , `CONTEXT_CELL` = 0x10 , `CONTEXT_TABLE` = 0x20 , `CONTEXT_RC_RESIZE`
= 0x40 }

The context bit flags for `Fl_Table` related callbacks.

Public Member Functions

- void **add** (`Fl_Widget` &wgt)
The specified widget is removed from its current group (if any) and added to the end of `Fl_Table`'s group.
- void **add** (`Fl_Widget` *wgt)
The specified widget is removed from its current group (if any) and added to the end of `Fl_Table`'s group.
- `Fl_Widget` *const * **array** ()
Returns a pointer to the array of children.
- void **begin** ()
- void **callback** (`Fl_Widget` *, void *)
Callbacks will be called depending on the setting of `Fl_Widget::when()`.
- int **callback_col** ()
Returns the current column the event occurred on.
- `TableContext` **callback_context** ()
Returns the current 'table context'.
- int **callback_row** ()
Returns the current row the event occurred on.
- `Fl_Widget` * **child** (int n) const
Returns the child widget by an index.
- int **children** () const
Returns the number of children in the table.
- virtual void **clear** ()
Clears the table to zero rows (`rows(0)`), zero columns (`cols(0)`), and clears any widgets (`table->clear()`) that were added with `begin()/end()` or `add()/insert()/etc.`
- int **col_header** ()
Returns if column headers are enabled or not.
- void **col_header** (int flag)
Enable or disable column headers.
- `Fl_Color` **col_header_color** ()
Gets the color for column headers.
- void **col_header_color** (`Fl_Color` val)
Sets the color for column headers and redraws the table.
- int **col_header_height** ()

- Gets the column header height.*
- void **col_header_height** (int height)
 - Sets the height in pixels for column headers and redraws the table.*
- int **col_position** ()
 - Returns the current column scroll position as a column number.*
- void **col_position** (int col)
 - Sets the horizontal scroll position so 'col' is at the left, and causes the screen to redraw.*
- int **col_resize** ()
 - Returns if column resizing by the user is allowed.*
- void **col_resize** (int flag)
 - Allows/disallows column resizing by the user.*
- int **col_resize_min** ()
 - Returns the current column minimum resize value.*
- void **col_resize_min** (int val)
 - Sets the current column minimum resize value.*
- int **col_width** (int col)
 - Returns the current width of the specified column in pixels.*
- void **col_width** (int col, int width)
 - Sets the width of the specified column in pixels, and the table is redrawn.*
- void **col_width_all** (int width)
 - Convenience method to set the width of all columns to the same value, in pixels.*
- int **cols** ()
 - Get the number of columns in the table.*
- virtual void **cols** (int val)
 - Set the number of columns in the table and redraw.*
- void **do_callback** (TableContext context, int row, int col)
 - Calls the widget callback.*
- void **end** ()
- int **find** (const FL_Widget &wgt) const
- int **find** (const FL_Widget *wgt) const
- **FL_Table** (int X, int Y, int W, int H, const char *l=0)
 - The constructor for FL_Table.*
- void **get_selection** (int &row_top, int &col_left, int &row_bot, int &col_right)
 - Gets the region of cells selected (highlighted).*
- void **init_sizes** ()
 - Resets the internal array of widget sizes and positions.*
- void **insert** (FL_Widget &wgt, FL_Widget *w2)
 - The specified widget is removed from its current group (if any) and inserted into FL_Table's group before widget 'w2'.*
- void **insert** (FL_Widget &wgt, int n)
 - The specified widget is removed from its current group (if any) and inserted into the FL_Table's group at position 'n'.*
- int **is_interactive_resize** ()
 - Returns 1 if someone is interactively resizing a row or column.*
- int **is_selected** (int r, int c)
 - See if the cell at row *r* and column *c* is selected.*
- int **move_cursor** (int R, int C)
 - Same as move_cursor(R,C,1);.*
- int **move_cursor** (int R, int C, int shiftselect)
 - Moves the selection cursor a relative number of rows/columns specified by R/C.*
- void **remove** (FL_Widget &wgt)
 - The specified widget is removed from FL_Table's group.*
- void **resize** (int X, int Y, int W, int H) **FL_OVERRIDE**

- Handle resize events if user resizes parent window.*

 - int **row_header** ()

Returns if row headers are enabled or not.
 - void **row_header** (int flag)

Enables/disables showing the row headers.
 - **FL_Color** **row_header_color** ()

Returns the current row header color.
 - void **row_header_color** (**FL_Color** val)

Sets the row header color and causes the screen to redraw.
 - int **row_header_width** ()

Returns the current row header width (in pixels).
 - void **row_header_width** (int width)

Sets the row header width to n and causes the screen to redraw.
 - int **row_height** (int row)

Returns the current height of the specified row as a value in pixels.
 - void **row_height** (int row, int height)

Sets the height of the specified row in pixels, and the table is redrawn.
 - void **row_height_all** (int height)

Convenience method to set the height of all rows to the same value, in pixels.
 - int **row_position** ()

Returns the current row scroll position as a row number.
 - void **row_position** (int row)

Sets the vertical scroll position so 'row' is at the top, and causes the screen to redraw.
 - int **row_resize** ()

Returns if row resizing by the user is allowed.
 - void **row_resize** (int flag)

Allows/disallows row resizing by the user.
 - int **row_resize_min** ()

Returns the current row minimum resize value.
 - void **row_resize_min** (int val)

Sets the current row minimum resize value.
 - int **rows** ()

Returns the number of rows in the table.
 - virtual void **rows** (int val)

Sets the number of rows in the table, and the table is redrawn.
 - int **scrollbar_size** () const

Gets the current size of the scrollbars' troughs, in pixels.
 - void **scrollbar_size** (int newSize)

Sets the pixel size of the scrollbars' troughs to newSize, in pixels.
 - void **set_selection** (int row_top, int col_left, int row_bot, int col_right)

Sets the region of cells to be selected (highlighted).
 - int **tab_cell_nav** () const

Get state of table's 'Tab' key cell navigation flag.
 - void **tab_cell_nav** (int val)

Flag to control if Tab navigates table cells or not.
 - void **table_box** (**FL_Boxtype** val)

Sets the kind of box drawn around the data table, the default being FL_NO_BOX.
 - **FL_Boxtype** **table_box** (void)

Returns the current box type used for the data table.
 - int **top_row** ()

Returns the current top row shown in the table.

- void **top_row** (int row)
Sets which row should be at the top of the table, scrolling as necessary, and the table is redrawn.
- void **visible_cells** (int &r1, int &r2, int &c1, int &c2)
Returns the range of row and column numbers for all visible and partially visible cells in the table.
- void **when** (FL_When flags)
The FL_Widget::when() function is used to set a group of flags, determining when the widget callback is called:
- ~FL_Table ()
The destructor for FL_Table.

Public Member Functions inherited from FL_Group

- FL_Widget *& **_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** (FL_Widget &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** (FL_Widget *o)
See void FL_Group::add(FL_Widget &w)
- void **add_resizable** (FL_Widget &o)
Adds a widget to the group and makes it the resizable widget.
- FL_Widget *const * **array** () const
Returns a pointer to the internal array of children.
- FL_Group const * **as_group** () const FL_OVERRIDE
- FL_Group * **as_group** () FL_OVERRIDE
Returns an FL_Group pointer if this widget is an FL_Group.
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- FL_Widget * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
Removes the widget at index from the group and deletes it.
- void **end** ()
Exactly the same as current(this->parent()).
- int **find** (const FL_Widget &o) const
*See int FL_Group::find(const FL_Widget *w) const.*
- int **find** (const FL_Widget *) const
Searches the children for the widget and returns the index.
- FL_Group (int, int, int, int, const char *==0)
Creates a new FL_Group widget using the given position, size, and label string.
- void **focus** (FL_Widget *W)
- void **forms_end** ()
This is for forms compatibility only.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.

- void `insert` (`FI_Widget` &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void `insert` (`FI_Widget` &o, `FI_Widget` *before)
This does insert(w, find(before)).
- void `remove` (`FI_Widget` &)
Removes a widget from the group but does not delete it.
- void `remove` (`FI_Widget` *o)
Removes the widget o from the group.
- void `remove` (int index)
Removes the widget at index from the group but does not delete it.
- `FI_Widget` * `resizable` () const
Returns the group's resizable widget.
- void `resizable` (`FI_Widget` &o)
Sets the group's resizable widget.
- void `resizable` (`FI_Widget` *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual `~FI_Group` ()
The destructor also deletes all the children.

Public Member Functions inherited from `FI_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
Activates the widget.
- unsigned int `active` () const
Returns whether the widget is active.
- int `active_r` () const
Returns whether the widget and all of its parents are active.
- `FI_Align` align () const
Gets the label alignment.
- void `align` (`FI_Align` alignment)
Sets the label alignment.
- long `argument` () const
Gets the current user data (long) argument that is passed to the callback function.
- void `argument` (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class `FI_Gl_Window` * `as_gl_window` ()
Returns an FI_Gl_Window pointer if this widget is an FI_Gl_Window.
- virtual class `FI_Gl_Window` const * `as_gl_window` () const
- virtual `FI_Window` * `as_window` ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual `FI_Window` const * `as_window` () const
- void `bind_deimage` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- void `bind_deimage` (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void `bind_image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- void `bind_image` (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.

- [FI_Boxtype box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) ([uchar](#) c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color color](#) () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)

- Sets the damage bits for an area inside the widget.*

 - int **damage_resize** (int, int, int, int)

Internal use only.
- void **deactivate** ()

Deactivates the widget.
- **FL_Image * deimage** ()

Gets the image that is used as part of the widget label when in the inactive state.
- const **FL_Image * deimage** () const

Gets the image that is used as part of the widget label when in the inactive state.
- void **deimage** (**FL_Image** &img)

Sets the image to use as part of the widget label when in the inactive state.
- void **deimage** (**FL_Image** *img)

Sets the image to use as part of the widget label when in the inactive state.
- int **deimage_bound** () const

Returns whether the inactive image is managed by the widget.
- void **do_callback** (**FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)

Calls the widget callback function with default arguments.
- void **do_callback** (**FL_Widget** *widget, long arg, **FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)

Calls the widget callback function with arbitrary arguments.
- void **do_callback** (**FL_Widget** *widget, void *arg=0, **FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)

Calls the widget callback function with arbitrary arguments.
- void **draw_label** (int, int, int, int, **FL_Align**) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int **h** () const

Gets the widget height.
- virtual void **hide** ()

Makes a widget invisible.
- int **horizontal_label_margin** ()

Get the spacing between the label and the horizontal edge of the widget.
- void **horizontal_label_margin** (int px)

Set the spacing between the label and the horizontal edge of the widget.
- **FL_Image * image** ()

Gets the image that is used as part of the widget label when in the active state.
- const **FL_Image * image** () const

Gets the image that is used as part of the widget label when in the active state.
- void **image** (**FL_Image** &img)

Sets the image to use as part of the widget label when in the active state.
- void **image** (**FL_Image** *img)

Sets the image to use as part of the widget label when in the active state.
- int **image_bound** () const

Returns whether the image is managed by the widget.
- int **inside** (const **FL_Widget** *wgt) const

Checks if this widget is a child of wgt.
- int **is_label_copied** () const

Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * **label** () const

Gets the current label text.
- void **label** (const char *text)

Sets the current label pointer.
- void **label** (**FL_Labeltype** a, const char *b)

Shortcut to set the label text and type in one call.

- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FI_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FI_Color c)`
Sets the label color.
- `FI_Font labelfont () const`
Gets the font to use.
- `void labelfont (FI_Font f)`
Sets the font to use.
- `FI_Fonsize labelsz () const`
Gets the font size in pixels.
- `void labelsz (FI_Fonsize pix)`
Sets the font size in pixels.
- `FI_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (FI_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`
Sets width ww and height hh accordingly with the label size.
- `bool needs_keyboard () const`
Returns whether this widget needs a keyboard.
- `void needs_keyboard (bool needs)`
Sets whether this widget needs a keyboard.
- `unsigned int output () const`
Returns if a widget is used for output only.
- `FI_Group * parent () const`
Returns a pointer to the parent widget.
- `void parent (FI_Group *p)`
Internal use only - "for hacks only".
- `void position (int X, int Y)`
Repositions the window or widget.
- `void redraw ()`
Schedules the drawing of the widget.
- `void redraw_label ()`
Schedules the drawing of the label.
- `FI_Color selection_color () const`
Gets the selection color.
- `void selection_color (FI_Color a)`
Sets the selection color.
- `void set_active ()`
Marks the widget as active without sending events or changing focus.
- `void set_changed ()`
Marks the value of the widget as changed.
- `void set_output ()`
Sets a widget to output only.
- `void set_visible ()`
Makes the widget visible.
- `void set_visible_focus ()`

- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

Gets the current tooltip text.
- void [tooltip](#) (const char *text)

Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const

Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const

Gets the widget type.
- void [type](#) ([uchar](#) t)

Sets the widget type.
- int [use_accents_menu](#) ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const

Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)

Sets the user data for this widget.
- void [user_data](#) (void *v)

Sets the user data for this widget.
- int [vertical_label_margin](#) ()

Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const

Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const

Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)

Modifies keyboard focus navigation.
- int [visible_r](#) () const

Returns whether a widget and all its parents are visible.
- int [w](#) () const

Gets the widget width.
- [Fl_When](#) [when](#) () const

Returns the conditions under which the callback is called.

- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **Fl_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual ~**Fl_Widget** ()
Destroys the widget.

Protected Types

- enum **ResizeFlag** {
 RESIZE_NONE = 0 , **RESIZE_COL_LEFT** = 1 , **RESIZE_COL_RIGHT** = 2 , **RESIZE_ROW_ABOVE** = 3 ,
 RESIZE_ROW_BELOW = 4 }

Protected Types inherited from **Fl_Widget**

- enum {
 INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
 FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
 ,
 OVERRIDE = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
 ,
 MENU_WINDOW = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
 ,
 GROUP_RELATIVE = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
 = 1<<19 ,
 NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
 AUTO_DELETE_USER_DATA = 1<<23 ,
 MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
 USERFLAG1 = 1<<31 }
flags possible values enumeration.

Protected Member Functions

- void **change_cursor** (**Fl_Cursor** newcursor)
Change mouse cursor to different type.
- long **col_scroll_position** (int col)
Returns the scroll position (in pixels) of the specified column 'col'.
- **TableContext** **cursor2rowcol** (int &R, int &C, **ResizeFlag** &resizeflag)
Find row/col for the recent mouse event.
- void **damage_zone** (int r1, int c1, int r2, int c2, int r3=0, int c3=0)
Sets the damage zone to the specified row/col values.
- void **draw** () **FL_OVERRIDE**
*Draws the entire **Fl_Table**.*
- virtual void **draw_cell** (**TableContext** context, int R=0, int C=0, int X=0, int Y=0, int W=0, int H=0)
Subclass should override this method to handle drawing the cells.
- int **find_cell** (**TableContext** context, int R, int C, int &X, int &Y, int &W, int &H)
Find a cell's X/Y/W/H region for the specified cell in row 'R', column 'C'.
- void **get_bounds** (**TableContext** context, int &X, int &Y, int &W, int &H)
Returns the (X,Y,W,H) bounding region for the specified 'context'.
- int **handle** (int e) **FL_OVERRIDE**

- Handle FLTK events.*

 - int **is_fltk_container** ()

Does the table contain any child fltk widgets?
 - void **recalc_dimensions** ()
- Recalculate the dimensions of the table, and affect any children.*
- void **redraw_range** (int topRow, int botRow, int leftCol, int rightCol)
- Define region of cells to be redrawn by specified range of rows/cols, and then sets damage(DAMAGE_CHILD).*
- int **row_col_clamp** (TableContext context, int &R, int &C)
- Return specified row/col values R and C to within the table's current row/col limits.*
- long **row_scroll_position** (int row)
- Returns the scroll position (in pixels) of the specified 'row'.*
- void **table_resized** ()
- Call this if table was resized, to recalculate internal data.*
- void **table_scrolled** ()
- Recalculate internals after a scroll.*

Protected Member Functions inherited from FI_Group

- **FI_Rect * bounds** ()
- Returns the internal array of widget sizes and positions.*
- void **draw_child** (FI_Widget &widget) const
- Forces a child to redraw.*
- void **draw_children** ()
- Draws all children of the group.*
- void **draw_outside_label** (const FI_Widget &widget) const
- Parents normally call this to draw outside labels of child widgets.*
- virtual int **on_insert** (FI_Widget *, int)
- Allow derived groups to act when a widget is added as a child.*
- virtual int **on_move** (int, int)
- Allow derived groups to act when a widget is moved within the group.*
- virtual void **on_remove** (int)
- Allow derived groups to act when a child widget is removed from the group.*
- int * **sizes** ()
- Returns the internal array of widget sizes and positions.*
- void **update_child** (FI_Widget &widget) const
- Draws a child only if it needs it.*

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
- Clears a flag in the flags mask.*
- void **draw_backdrop** () const
- If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.*
- void **draw_box** () const
- Draws the widget box according its box style.*
- void **draw_box** (FI_Boxtype t, FI_Color c) const
- Draws a box of type t, of color c at the widget's position and size.*
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
- Draws a box of type t, of color c at the position X,Y and size W,H.*
- void **draw_focus** () const
- Draws a focus rectangle around the widget.*
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const

- Draws a focus rectangle around the widget.*
- void **draw_focus** (**FI_Boxtype** t, int x, int y, int w, int h, **FI_Color** bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **FI_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Static Protected Member Functions

- static void **scroll_cb** (**FI_Widget** *, void *)
Callback for when someone moves a scrollbar.

Protected Attributes

- int **botrow**
bottom row# of currently visible table on screen
- int **current_col**
selection cursor's current column (-1 if none)
- int **current_row**
selection cursor's current row (-1 if none)
- **FI_Scrollbar** * **hscrollbar**
child horizontal scrollbar widget
- int **leftcol**
left column# of currently visible table on screen
- int **leftcol_scrollpos**
precomputed scroll position for left column
- int **rightcol**
right column# of currently visible table on screen
- int **select_col**
extended selection column (-1 if none)
- int **select_row**
extended selection row (-1 if none)
- **FI_Scroll** * **table**
child FI_Scroll widget container for child fltk widgets (if any)
- int **table_h**
table's virtual height (in pixels)
- int **table_w**

- table's virtual width (in pixels)*
- int **tih**
Data table's inner h dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **tiw**
Data table's inner w dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **tix**
Data table's inner x dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **tiy**
Data table's inner y dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **toh**
Data table's outer h dimension, outside bounding box. See [Table Dimension Diagram](#).
- int **toprow**
top row# of currently visible table on screen
- int **toprow_scrollpos**
precomputed scroll position for top row
- int **tow**
Data table's outer w dimension, outside bounding box. See [Table Dimension Diagram](#).
- int **tox**
Data table's outer x dimension, outside bounding box. See [Table Dimension Diagram](#).
- int **toy**
Data table's outer y dimension, outside bounding box. See [Table Dimension Diagram](#).
- [FI_Scrollbar](#) * **vscrollbar**
child vertical scrollbar widget
- int **wih**
Table widget's inner h dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **wiw**
Table widget's inner w dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **wix**
Table widget's inner x dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **wiy**
Table widget's inner y dimension, inside bounding box. See [Table Dimension Diagram](#).

Additional Inherited Members

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * **current** ()
Returns the currently active group.
- static void **current** ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

11.139.1 Detailed Description

A table of widgets or other content.

This is the base class for table widgets.

To be useful it must be subclassed and several virtual functions defined. Normally applications use widgets derived from this widget, and do not use this widget directly; this widget is usually too low level to be used directly by applications.

This widget does *not* handle the data in the table. The `draw_cell()` method must be overridden by a subclass to manage drawing the contents of the cells.

This widget can be used in several ways:

- As a custom widget; see `examples/table-simple.cxx` and `test/table.cxx`. Very optimal for even extremely large tables.
- As a table made up of a single FLTK widget instanced all over the table, simulating a numeric spreadsheet. See `examples/table-spreadsheet.cxx` and `examples/table-spreadsheet-with-keyboard-nav.cxx`. Optimal for large tables.
- As a regular container of FLTK widgets, one widget per cell. See `examples/table-as-container.cxx`. *Not* recommended for large tables.

	A	B	C	D	E
000:	1000	1001	1002	1003	1004
001:	2000	2001	2002	2003	2004
002:	3000	3001	3002	3003	3004
003:	4000	4001	4002	4003	4004
004:	5000	5001	5002	5003	5004
005:	6000	6001	6002	6003	6004
006:	7000	7001	7002	7003	7004
007:	8000	8001	8002	8003	8004
008:	9000	9001	9002	9003	9004
009:	10000	10001	10002	10003	10004

Figure 11.47 table-simple example

	Column 0	Column 1	Column 2	Column 3	Column 4	Column 5
Row 0	0/0	0.1	0/2	0.3	0/4	0.5
Row 1	1/0	1.1	1/2	1.3	1/4	1.5
Row 2	2/0	2.1	2/2	2.3	2/4	2.5
Row 3	3/0	3.1	3/2	3.3	3/4	3.5
Row 4	4/0	4.1	4/2	4.3	4/4	4.5
Row 5	5/0	5.1	5/2	5.3	5/4	5.5
Row 6	6/0	6.1	6/2	6.3	6/4	6.5
Row 7	7/0	7.1	7/2	7.3	7/4	7.5

Figure 11.48 table-as-container example

When acting as part of a custom widget, events on the cells and/or headings generate callbacks when they are clicked by the user. You control when events are generated based on the setting for `Fl_Table::when()`.

When acting as a container for FLTK widgets, the FLTK widgets maintain themselves. Although the `draw_cell()` method must be overridden, its contents can be very simple. See the `draw_cell()` code in `examples/table-simple.cxx`.

The following variables are available to classes deriving from [Fl_Table](#):

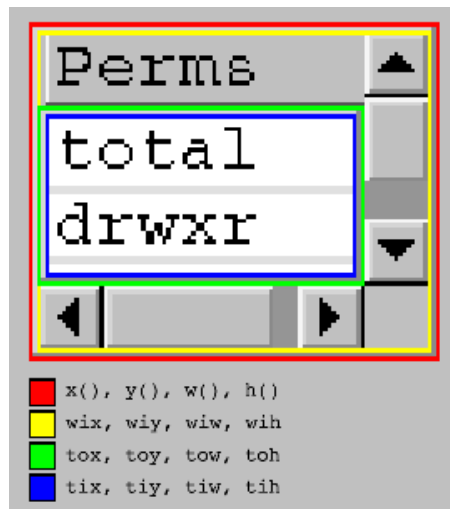


Figure 11.49 Fl_Table Dimensions

<code>x()/y()/w()/h()</code>	Fl_Table widget's outer dimension. The outer edge of the border of the Fl_Table . (Red in the diagram above)
<code>wix/wiy/wiw/wih</code>	Fl_Table widget's inner dimension. The inner edge of the border of the Fl_Table . eg. if the Fl_Table 's <code>box()</code> is <code>FL_NO_BOX</code> , these values are the same as <code>x()/y()/w()/h()</code> . (Yellow in the diagram above)
<code>tox/toy/tow/toh</code>	The table's outer dimension. The outer edge of the border around the cells, but inside the row/col headings and scrollbars. (Green in the diagram above)
<code>tix/tiy/tiw/tih</code>	The table's inner dimension. The inner edge of the border around the cells, but inside the row/col headings and scrollbars. AKA the table's clip region. eg. if the <code>table_box()</code> is <code>FL_↔ NO_BOX</code> , these values are the same as <code>tox/toy/tow/toh</code> . (Blue in the diagram above)

CORE DEVELOPERS

- Greg Ercolano : 12/16/2002 - initial implementation 12/16/02. [Fl_Table](#), [Fl_Table_Row](#), docs.
- Jean-Marc Lienher : 02/22/2004 - added keyboard nav + mouse selection, and ported [Fl_Table](#) into fltk-utf8-1.1.4

OTHER CONTRIBUTORS

- Inspired by the Feb 2000 version of FLVW's `Flvw_Table` widget. Mucho thanks to those folks.
- Mister Satan : 04/07/2003 - MinGW porting mods, and `singleinput.cxx`; a cool [Fl_Input](#) oriented spreadsheet example
- Marek Paliwoda : 01/08/2003 - Porting mods for Borland
- Ori Berger : 03/16/2006 - Optimizations for >500k rows/cols

LICENSE

Greg kindly gave his permission to integrate [Fl_Table](#) and [Fl_Table_Row](#) into FLTK, allowing FLTK license to apply while his widgets are part of the library. [updated by Greg, 04/26/17]

11.139.2 Member Enumeration Documentation

11.139.2.1 TableContext

enum `Fl_Table::TableContext`

The context bit flags for `Fl_Table` related callbacks.

Should be used in `draw_cell()` to determine what's being drawn, or in a `callback()` to determine where a recent event occurred.

Enumerator

CONTEXT_NONE	no known context
CONTEXT_STARTPAGE	before the table is redrawn
CONTEXT_ENDPAGE	after the table is redrawn
CONTEXT_ROW_HEADER	drawing or event occurred in the row header
CONTEXT_COL_HEADER	drawing or event occurred in the col header
CONTEXT_CELL	drawing or event occurred in a cell
CONTEXT_TABLE	drawing or event occurred in a dead zone of table
CONTEXT_RC_RESIZE	column or row is being resized

11.139.3 Constructor & Destructor Documentation

11.139.3.1 Fl_Table()

```
Fl_Table::Fl_Table (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

The constructor for `Fl_Table`.

This creates an empty table with no rows or columns, with headers and row/column resize behavior disabled.

11.139.3.2 ~Fl_Table()

```
Fl_Table::~~Fl_Table ()
```

The destructor for `Fl_Table`.

Destroys the table and its associated widgets.

11.139.4 Member Function Documentation

11.139.4.1 array()

```
Fl_Widget *const * Fl_Table::array () [inline]
```

Returns a pointer to the array of children.

This pointer is only valid until the next time a child is added or removed.

11.139.4.2 callback()

```
void Fl_Table::callback (
    Fl_Widget * ,
    void * )
```

Callbacks will be called depending on the setting of `Fl_Widget::when()`.

Callback functions should use the following functions to determine the context/row/column:

- `Fl_Table::callback_row()` returns current row
- `Fl_Table::callback_col()` returns current column

- `Fl_Table::callback_context()` returns current table context

`callback_row()` and `callback_col()` will be set to the row and column number the event occurred on. If someone clicked on a row header, `col` will be 0. If someone clicked on a column header, `row` will be 0.

`callback_context()` will return one of the following:

<code>Fl_Table::CONTEXT_ROW_HEADER</code>	Someone clicked on a row header. Excludes resizing.
<code>Fl_Table::CONTEXT_COL_HEADER</code>	Someone clicked on a column header. Excludes resizing.
<code>Fl_Table::CONTEXT_CELL</code>	Someone clicked on a cell. To receive callbacks for FL_RELEASE events, you must set when(FL_WHEN_RELEASE).
<code>Fl_Table::CONTEXT_RC_RESIZE</code>	Someone is resizing rows/columns either interactively, or via the <code>col_width()</code> or <code>row_height()</code> API. Use <code>is_interactive_resize()</code> to determine interactive resizing. If resizing a column, R=0 and C=column being resized. If resizing a row, C=0 and R=row being resized. NOTE: To receive resize events, you must set when(FL_WHEN_↵_CHANGED).

```
class MyTable : public Fl_Table {
[...]
```

private:

```
// Handle events that happen on the table
void event_callback2() {
    int R = callback_row(),           // row where event occurred
        C = callback_col();          // column where event occurred
    TableContext context = callback_context(); // which part of table
    fprintf(stderr, "callback: Row=%d Col=%d Context=%d Event=%d\n",
        R, C, (int)context, (int)Fl::event());
}

// Actual static callback
static void event_callback(Fl_Widget*, void* data) {
    MyTable *o = (MyTable*)data;
    o->event_callback2();
}

public:
// Constructor
MyTable() {
    [...]
    table.callback(&event_callback, (void*)this); // setup callback
    table.when(FL_WHEN_CHANGED|FL_WHEN_RELEASE); // when to call it
}
};
```

11.139.4.3 callback_col()

```
int Fl_Table::callback_col () [inline]
```

Returns the current column the event occurred on.

This function should only be used from within the user's callback function.

11.139.4.4 callback_context()

```
TableContext Fl_Table::callback_context () [inline]
```

Returns the current 'table context'.

This function should only be used from within the user's callback function.

11.139.4.5 callback_row()

```
int Fl_Table::callback_row () [inline]
```

Returns the current row the event occurred on.

This function should only be used from within the user's callback function.

11.139.4.6 child()

```
Fl_Widget * Fl_Table::child (
    int n) const [inline]
```

Returns the child widget by an index.

When using the [Fl_Table](#) as a container for FLTK widgets, this method returns the widget pointer from the internal array of widgets in the container.

Typically used in loops, eg:

```
for ( int i=0; i<children(); i++ ) {
    Fl_Widget *w = child(i);
    [...]
}
```

11.139.4.7 children()

```
int Fl_Table::children () const [inline]
```

Returns the number of children in the table.

When using the [Fl_Table](#) as a container for FLTK widgets, this method returns how many child widgets the table has.

See also

[child\(int\)](#)

11.139.4.8 clear()

```
virtual void Fl_Table::clear () [inline], [virtual]
```

Clears the table to zero rows ([rows\(0\)](#)), zero columns ([cols\(0\)](#)), and clears any widgets ([table->clear\(\)](#)) that were added with [begin\(\)/end\(\)](#) or [add\(\)/insert\(\)/etc.](#)

See also

[rows\(int\)](#), [cols\(int\)](#)

Reimplemented in [Fl_Table_Row](#).

11.139.4.9 col_header()

```
void Fl_Table::col_header (
    int flag) [inline]
```

Enable or disable column headers.

If changed, the table is redrawn.

11.139.4.10 col_resize()

```
void Fl_Table::col_resize (
    int flag) [inline]
```

Allows/disallows column resizing by the user.

1=allow interactive resizing, 0=disallow interactive resizing. Since interactive resizing is done via the column headers, [col_header\(\)](#) must also be enabled to allow resizing.

11.139.4.11 col_resize_min()

```
void Fl_Table::col_resize_min (
    int val) [inline]
```

Sets the current column minimum resize value.

This is used to prevent the user from interactively resizing any column to be smaller than 'pixels'. Must be a value ≥ 1 .

11.139.4.12 col_width()

```
void Fl_Table::col_width (
    int col,
    int width)
```

Sets the width of the specified column in pixels, and the table is redrawn.

[callback\(\)](#) will be invoked with `CONTEXT_RC_RESIZE` if the column's width was actually changed, and [when\(\)](#) is `FL_WHEN_CHANGED`.

11.139.4.13 col_width_all()

```
void Fl_Table::col_width_all (
    int width) [inline]
```

Convenience method to set the width of all columns to the same value, in pixels.
The screen is redrawn.

11.139.4.14 cursor2rowcol()

```
Fl_Table::TableContext Fl_Table::cursor2rowcol (
    int & R,
    int & C,
    ResizeFlag & resizeflag) [protected]
```

Find row/col for the recent mouse event.

Returns the context, and the row/column values in R/C. Also returns 'resizeflag' if mouse is hovered over a resize boundary.

11.139.4.15 damage_zone()

```
void Fl_Table::damage_zone (
    int r1,
    int c1,
    int r2,
    int c2,
    int r3 = 0,
    int c3 = 0) [protected]
```

Sets the damage zone to the specified row/col values.

Calls [redraw_range\(\)](#).

11.139.4.16 do_callback()

```
void Fl_Table::do_callback (
    TableContext context,
    int row,
    int col) [inline]
```

Calls the widget callback.

Saves the specified 'context', 'row', and 'col' values, so that the user's callback can then access them with the member functions [callback_context\(\)](#), [callback_row\(\)](#) and [callback_col\(\)](#).

11.139.4.17 draw()

```
void Fl_Table::draw (
    void ) [protected], [virtual]
```

Draws the entire [Fl_Table](#).

Lets fltk widgets draw themselves first, followed by the cells via calls to [draw_cell\(\)](#).

Reimplemented from [Fl_Group](#).

11.139.4.18 draw_cell()

```
virtual void Fl_Table::draw_cell (
    TableContext context,
    int R = 0,
    int C = 0,
    int X = 0,
    int Y = 0,
    int W = 0,
    int H = 0) [inline], [protected], [virtual]
```

Subclass should override this method to handle drawing the cells.

This method will be called whenever the table is redrawn, once per cell.

Only cells that are completely (or partially) visible will be told to draw.
context will be one of the following:

<code>Fl_Table::CONTEXT_STARTPAGE</code>	When table, or parts of the table, are about to be redrawn. Use to initialize static data, such as font selections. R/C will be zero, X/Y/W/H will be the dimensions of the table's entire data area. (Useful for locking a database before accessing; see also visible_cells())
<code>Fl_Table::CONTEXT_ENDPAGE</code>	When table has completed being redrawn. R/C will be zero, X/Y/W/H dimensions of table's data area. (Useful for unlocking a database after accessing)
<code>Fl_Table::CONTEXT_ROW_HEADER</code>	Whenever a row header cell needs to be drawn. R will be the row number of the header being redrawn, C will be zero, X/Y/W/H will be the fltk drawing area of the row header in the window
<code>Fl_Table::CONTEXT_COL_HEADER</code>	Whenever a column header cell needs to be drawn. R will be zero, C will be the column number of the header being redrawn, X/Y/W/H will be the fltk drawing area of the column header in the window
<code>Fl_Table::CONTEXT_CELL</code>	Whenever a data cell in the table needs to be drawn. R/C will be the row/column of the cell to be drawn, X/Y/W/H will be the fltk drawing area of the cell in the window
<code>Fl_Table::CONTEXT_RC_RESIZE</code>	Whenever table or row/column is resized or scrolled, either interactively or via col_width() or row_height() . R/C/X/Y/W/H will all be zero. Useful for fltk containers that need to resize or move the child fltk widgets.

R and C will be set to the row and column number of the cell being drawn. In the case of row headers, C will be 0. In the case of column headers, R will be 0.

X/Y/W/H will be the position and dimensions of where the cell should be drawn.

In the case of custom widgets, a minimal [draw_cell\(\)](#) override might look like the following. With custom widgets it is up to the caller to handle drawing everything within the dimensions of the cell, including handling the selection color. Note all clipping must be handled as well; this allows drawing outside the dimensions of the cell if so desired for 'custom effects'.

```
// This is called whenever Fl_Table wants you to draw a cell
void MyTable::draw_cell(TableContext context, int R=0, int C=0, int X=0, int Y=0, int W=0, int H=0) {
    static char s[40];
    sprintf(s, "%d/%d", R, C);           // text for each cell
    switch ( context ) {
        case CONTEXT_STARTPAGE:         // Fl_Table telling us it's starting to draw page
            fl_font(FL_HELVETICA, 16);
            return;

        case CONTEXT_ROW_HEADER:         // Fl_Table telling us to draw row/col headers
        case CONTEXT_COL_HEADER:
            fl_push_clip(X, Y, W, H);
            {
                fl_draw_box(FL_THIN_UP_BOX, X, Y, W, H, color());
                fl_color(FL_BLACK);
                fl_draw(s, X, Y, W, H, FL_ALIGN_CENTER);
            }
            fl_pop_clip();
            return;

        case CONTEXT_CELL:               // Fl_Table telling us to draw cells
            fl_push_clip(X, Y, W, H);
            {
                // BG COLOR
                fl_color(is_selected(R, C) ? selection_color() : FL_WHITE);
                fl_rectf(X, Y, W, H);

                // TEXT
                fl_color(FL_BLACK);
            }
            return;
    }
}
```



```

        fl_draw(s, X, Y, W, H, FL_ALIGN_CENTER);

        // BORDER
        fl_color(FL_LIGHT2);
        fl_rect(X, Y, W, H);
    }
    fl_pop_clip();
    return;

default:
    return;
}
//NOTREACHED
}

```

11.139.4.19 find_cell()

```

int Fl_Table::find_cell (
    TableContext context,
    int R,
    int C,
    int & X,
    int & Y,
    int & W,
    int & H) [protected]

```

Find a cell's X/Y/W/H region for the specified cell in row 'R', column 'C'.

Returns

- 0 – on success, XYWH returns the region of the specified cell.
- -1 – if R or C are out of range, and X/Y/W/H will be set to zero.

11.139.4.20 get_selection()

```

void Fl_Table::get_selection (
    int & row_top,
    int & col_left,
    int & row_bot,
    int & col_right)

```

Gets the region of cells selected (highlighted).

Parameters

in	<i>row_top</i>	Returns the top row of selection area
in	<i>col_left</i>	Returns the left column of selection area
in	<i>row_bot</i>	Returns the bottom row of selection area
in	<i>col_right</i>	Returns the right column of selection area

11.139.4.21 handle()

```

int Fl_Table::handle (
    int e) [protected], [virtual]

```

Handle FLTK events.

Reimplemented from [Fl_Group](#).

Reimplemented in [Fl_Table_Row](#).

11.139.4.22 init_sizes()

```

void Fl_Table::init_sizes () [inline]

```

Resets the internal array of widget sizes and positions.

See also

[Fl_Group::init_sizes\(\)](#)

11.139.4.23 insert()

```
void Fl_Table::insert (
    Fl_Widget & wgt,
    Fl_Widget * w2) [inline]
```

The specified widget is removed from its current group (if any) and inserted into [Fl_Table](#)'s group before widget 'w2'. This will append if 'w2' is not in [Fl_Table](#)'s group.

11.139.4.24 is_interactive_resize()

```
int Fl_Table::is_interactive_resize () [inline]
```

Returns 1 if someone is interactively resizing a row or column.

You can currently call this only from within your [callback\(\)](#).

11.139.4.25 is_selected()

```
int Fl_Table::is_selected (
    int r,
    int c)
```

See if the cell at row *r* and column *c* is selected.

Returns

1 if the cell is selected, 0 if not.

11.139.4.26 move_cursor()

```
int Fl_Table::move_cursor (
    int R,
    int C,
    int shiftselect)
```

Moves the selection cursor a relative number of rows/columns specified by R/C.

R/C can be positive or negative, depending on the direction to move. A value of 0 for R or C prevents cursor movement on that axis.

If shiftselect is set, the selection range is extended to the new cursor position. If clear, the cursor is simply moved, and any previous selection is cancelled.

Used mainly by keyboard events (e.g. [Fl_Right](#), [FL_Home](#), [FL_End](#)..) to let the user keyboard navigate the selection cursor around.

The scroll positions may be modified if the selection cursor traverses into cells off the screen's edge.

Internal variables `select_row/select_col` and `current_row/current_col` are modified, among others.

Examples:

```
R=1, C=0 -- moves the selection cursor one row downward.
R=5, C=0 -- moves the selection cursor 5 rows downward.
R=-5, C=0 -- moves the cursor 5 rows upward.
R=2, C=2 -- moves the cursor 2 rows down and 2 columns to the right.
```

11.139.4.27 recalc_dimensions()

```
void Fl_Table::recalc_dimensions () [protected]
```

Recalculate the dimensions of the table, and affect any children.

Internally, [Fl_Group::resize\(\)](#) and [init_sizes\(\)](#) are called.

11.139.4.28 redraw_range()

```
void Fl_Table::redraw_range (
    int topRow,
    int botRow,
```

```
int leftCol,
int rightCol) [inline], [protected]
```

Define region of cells to be redrawn by specified range of rows/cols, and then sets damage(DAMAGE_CHILD).
Extends any previously defined range to redraw.

11.139.4.29 `resize()`

```
void Fl_Table::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Handle resize events if user resizes parent window.
This changes the size of [Fl_Table](#), causing it to redraw.
Reimplemented from [Fl_Group](#).

11.139.4.30 `row_col_clamp()`

```
int Fl_Table::row_col_clamp (
    TableContext context,
    int & R,
    int & C) [protected]
```

Return specified row/col values R and C to within the table's current row/col limits.

Returns

0 if no changes were made, or 1 if they were.

11.139.4.31 `row_header()`

```
void Fl_Table::row_header (
    int flag) [inline]
```

Enables/disables showing the row headers.
1=enabled, 0=disabled. If changed, the table is redrawn.

11.139.4.32 `row_height()`

```
void Fl_Table::row_height (
    int row,
    int height)
```

Sets the height of the specified row in pixels, and the table is redrawn.
[callback\(\)](#) will be invoked with CONTEXT_RC_RESIZE if the row's height was actually changed, and [when\(\)](#) is FL_WHEN_CHANGED.

11.139.4.33 `row_height_all()`

```
void Fl_Table::row_height_all (
    int height) [inline]
```

Convenience method to set the height of all rows to the same value, in pixels.
The screen is redrawn.

11.139.4.34 `row_resize()`

```
void Fl_Table::row_resize (
    int flag) [inline]
```

Allows/disallows row resizing by the user.
1=allow interactive resizing, 0=disallow interactive resizing. Since interactive resizing is done via the row headers, [row_header\(\)](#) must also be enabled to allow resizing.

11.139.4.35 row_resize_min()

```
void Fl_Table::row_resize_min (
    int val) [inline]
```

Sets the current row minimum resize value.

This is used to prevent the user from interactively resizing any row to be smaller than 'pixels'. Must be a value ≥ 1 .

11.139.4.36 rows()

```
void Fl_Table::rows (
    int val) [virtual]
```

Sets the number of rows in the table, and the table is redrawn.

Reimplemented in [Fl_Table_Row](#).

11.139.4.37 scrollbar_size() [1/2]

```
int Fl_Table::scrollbar_size () const [inline]
```

Gets the current size of the scrollbars' troughs, in pixels.

If this value is zero (default), this widget will use the [Fl::scrollbar_size\(\)](#) value as the scrollbar's width.

Returns

Scrollbar size in pixels, or 0 if the global [Fl::scrollbar_size\(\)](#) is being used.

See also

[Fl::scrollbar_size\(int\)](#)

11.139.4.38 scrollbar_size() [2/2]

```
void Fl_Table::scrollbar_size (
    int newSize) [inline]
```

Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.

Normally you should not need this method, and should use [Fl::scrollbar_size\(int\)](#) instead to manage the size of ALL your widgets' scrollbars. This ensures your application has a consistent UI, is the default behavior, and is normally what you want.

Only use THIS method if you really need to override the global scrollbar size. The need for this should be rare.

Setting `newSize` to the special value of 0 causes the widget to track the global [Fl::scrollbar_size\(\)](#), which is the default.

Parameters

in	<i>newSize</i>	Sets the scrollbar size in pixels. If 0 (default), scrollbar size tracks the global Fl::scrollbar_size()

See also

[Fl::scrollbar_size\(\)](#)

11.139.4.39 set_selection()

```
void Fl_Table::set_selection (
    int row_top,
    int col_left,
    int row_bot,
    int col_right)
```

Sets the region of cells to be selected (highlighted).

So for instance, `set_selection(0,0,0,0)` selects the top/left cell in the table. And `set_selection(0,0,1,1)` selects the four cells in rows 0 and 1, column 0 and 1.

To deselect all cells, use `set_selection(-1,-1,-1,-1)`;

Parameters

in	<i>row_top</i>	Top row of selection area
in	<i>col_left</i>	Left column of selection area
in	<i>row_bot</i>	Bottom row of selection area
in	<i>col_right</i>	Right column of selection area

11.139.4.40 tab_cell_nav() [1/2]

```
int Fl_Table::tab_cell_nav () const [inline]
```

Get state of table's 'Tab' key cell navigation flag.

Returns

1 if Tab configured to navigate cells in table
0 to navigate widget focus (default)

See also

[tab_cell_nav\(int\)](#)

11.139.4.41 tab_cell_nav() [2/2]

```
void Fl_Table::tab_cell_nav (
    int val) [inline]
```

Flag to control if Tab navigates table cells or not.

If on, Tab key navigates table cells. If off, Tab key navigates fltk widget focus. (default)

As of fltk 1.3, the default behavior of the Tab key is to navigate focus off of the current widget, and on to the next one. But in some applications, it's useful for Tab to be used to navigate cells in the [Fl_Table](#).

Parameters

in	<i>val</i>	If <i>val</i> is 1, Tab key navigates cells in table, not fltk widgets. If <i>val</i> is 0, Tab key will advance focus to the next fltk widget (default), and does not navigate cells in table.

11.139.4.42 table_box()

```
void Fl_Table::table_box (
    Fl_Boxtype val) [inline]
```

Sets the kind of box drawn around the data table, the default being FL_NO_BOX.

Changing this value will cause the table to redraw.

11.139.4.43 table_resized()

```
void Fl_Table::table_resized () [protected]
```

Call this if table was resized, to recalculate internal data.

Calls `recall_dimensions()`, and recalculates scrollbar sizes.

11.139.4.44 table_scrolled()

```
void Fl_Table::table_scrolled () [protected]
```

Recalculate internals after a scroll.

Call this if table has been scrolled or resized. Does not handle [redraw\(\)](#). TODO: Assumes `ti[xywh]` has already been recalculated.

11.139.4.45 top_row() [1/2]

```
int Fl_Table::top_row () [inline]
```

Returns the current top row shown in the table.

This row may be partially obscured.

11.139.4.46 top_row() [2/2]

```
void Fl_Table::top_row (
    int row) [inline]
```

Sets which row should be at the top of the table, scrolling as necessary, and the table is redrawn.

If the table cannot be scrolled that far, it is scrolled as far as possible.

11.139.4.47 visible_cells()

```
void Fl_Table::visible_cells (
    int & r1,
    int & r2,
    int & c1,
    int & c2) [inline]
```

Returns the range of row and column numbers for all visible and partially visible cells in the table.

These values can be used e.g. by your [draw_cell\(\)](#) routine during CONTEXT_STARTPAGE to figure out what cells are about to be redrawn for the purposes of locking the data from a database before it's drawn.

```

leftcol      rightcol
:            :
toprow .. .-----:
|            |
|  V I S I B L E  |
|            |
|    T A B L E    |
|            |
botrow .. .-----`

```

e.g. in a table where the visible rows are 5-20, and the visible columns are 100-120, then those variables would be:

- toprow = 5
- botrow = 20
- leftcol = 100
- rightcol = 120

11.139.4.48 when()

```
void Fl_Table::when (
    Fl_When flags)
```

The [Fl_Widget::when\(\)](#) function is used to set a group of flags, determining when the widget callback is called:

FL_WHEN_CHANGED	callback() will be called when rows or columns are resized (interactively or via col_width() or row_height()), passing CONTEXT_RC_RESIZE via callback_context() .
FL_WHEN_RELEASE	callback() will be called during FL_RELEASE events, such as when someone releases a mouse button somewhere on the table.

The [callback\(\)](#) routine is sent a TableContext that indicates the context the event occurred in, such as in a cell, in a header, or elsewhere on the table. When an event occurs in a cell or header, [callback_row\(\)](#) and [callback_col\(\)](#) can be used to determine the row and column. The callback can also look at the regular fltk event values (ie. [Fl::event\(\)](#) and [Fl::event_button\(\)](#)) to determine what kind of event is occurring.

The documentation for this class was generated from the following files:

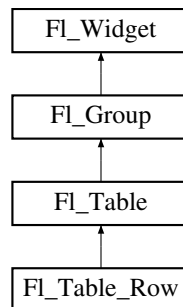
- Fl_Table.H
- Fl_Table.cxx

11.140 Fl_Table_Row Class Reference

A table with row selection capabilities.

```
#include <Fl_Table_Row.H>
```

Inheritance diagram for Fl_Table_Row:



Public Types

- enum **TableRowSelectMode** { **SELECT_NONE** , **SELECT_SINGLE** , **SELECT_MULTI** }

Public Types inherited from Fl_Table

- enum **TableContext** {
CONTEXT_NONE = 0 , **CONTEXT_STARTPAGE** = 0x01 , **CONTEXT_ENDPAGE** = 0x02 , **CONTEXT_ROW_HEADER**
 = 0x04 ,
CONTEXT_COL_HEADER = 0x08 , **CONTEXT_CELL** = 0x10 , **CONTEXT_TABLE** = 0x20 , **CONTEXT_RC_RESIZE**
 = 0x40 }

The context bit flags for [Fl_Table](#) related callbacks.

Public Member Functions

- void **clear** () **FL_OVERRIDE**
Clears the table to zero rows (rows(0)), zero columns (cols(0)), and clears any widgets (table->clear()) that were added with begin()/end() or add()/insert()/etc.
- Fl_Table_Row** (int X, int Y, int W, int H, const char *l=0)
The constructor for the [Fl_Table_Row](#).
- int **row_selected** (int row)
Checks to see if 'row' is selected.
- int **rows** ()
- void **rows** (int val) **FL_OVERRIDE**
Sets the number of rows in the table, and the table is redrawn.
- void **select_all_rows** (int flag=1)
This convenience function changes the selection state for all rows based on 'flag'.
- int **select_row** (int row, int flag=1)
Changes the selection state for 'row', depending on the value of 'flag'.
- TableRowSelectMode **type** () const
- void **type** (TableRowSelectMode val)
Sets the table selection mode.
- ~Fl_Table_Row** ()
The destructor for the [Fl_Table_Row](#).

Public Member Functions inherited from [FI_Table](#)

- void **add** ([FI_Widget](#) &wgt)
The specified widget is removed from its current group (if any) and added to the end of [FI_Table](#)'s group.
- void **add** ([FI_Widget](#) *wgt)
The specified widget is removed from its current group (if any) and added to the end of [FI_Table](#)'s group.
- [FI_Widget](#) *const * **array** ()
Returns a pointer to the array of children.
- void **begin** ()
- void **callback** ([FI_Widget](#) *, void *)
Callbacks will be called depending on the setting of [FI_Widget::when\(\)](#).
- int **callback_col** ()
Returns the current column the event occurred on.
- [TableContext](#) **callback_context** ()
Returns the current 'table context'.
- int **callback_row** ()
Returns the current row the event occurred on.
- [FI_Widget](#) * **child** (int n) const
Returns the child widget by an index.
- int **children** () const
Returns the number of children in the table.
- int **col_header** ()
Returns if column headers are enabled or not.
- void **col_header** (int flag)
Enable or disable column headers.
- [FI_Color](#) **col_header_color** ()
Gets the color for column headers.
- void **col_header_color** ([FI_Color](#) val)
Sets the color for column headers and redraws the table.
- int **col_header_height** ()
Gets the column header height.
- void **col_header_height** (int height)
Sets the height in pixels for column headers and redraws the table.
- int **col_position** ()
Returns the current column scroll position as a column number.
- void **col_position** (int col)
Sets the horizontal scroll position so 'col' is at the left, and causes the screen to redraw.
- int **col_resize** ()
Returns if column resizing by the user is allowed.
- void **col_resize** (int flag)
Allows/disallows column resizing by the user.
- int **col_resize_min** ()
Returns the current column minimum resize value.
- void **col_resize_min** (int val)
Sets the current column minimum resize value.
- int **col_width** (int col)
Returns the current width of the specified column in pixels.
- void **col_width** (int col, int width)
Sets the width of the specified column in pixels, and the table is redrawn.
- void **col_width_all** (int width)
Convenience method to set the width of all columns to the same value, in pixels.

- **int cols ()**
Get the number of columns in the table.
- **virtual void cols (int val)**
Set the number of columns in the table and redraw.
- **void do_callback (TableContext context, int row, int col)**
Calls the widget callback.
- **void end ()**
- **int find (const FL_Widget &wgt) const**
- **int find (const FL_Widget *wgt) const**
- **FL_Table (int X, int Y, int W, int H, const char *l=0)**
The constructor for FL_Table.
- **void get_selection (int &row_top, int &col_left, int &row_bot, int &col_right)**
Gets the region of cells selected (highlighted).
- **void init_sizes ()**
Resets the internal array of widget sizes and positions.
- **void insert (FL_Widget &wgt, FL_Widget *w2)**
The specified widget is removed from its current group (if any) and inserted into FL_Table's group before widget 'w2'.
- **void insert (FL_Widget &wgt, int n)**
The specified widget is removed from its current group (if any) and inserted into the FL_Table's group at position 'n'.
- **int is_interactive_resize ()**
Returns 1 if someone is interactively resizing a row or column.
- **int is_selected (int r, int c)**
See if the cell at row r and column c is selected.
- **int move_cursor (int R, int C)**
Same as move_cursor(R,C,1);.
- **int move_cursor (int R, int C, int shiftselect)**
Moves the selection cursor a relative number of rows/columns specified by R/C.
- **void remove (FL_Widget &wgt)**
The specified widget is removed from FL_Table's group.
- **void resize (int X, int Y, int W, int H) FL_OVERRIDE**
Handle resize events if user resizes parent window.
- **int row_header ()**
Returns if row headers are enabled or not.
- **void row_header (int flag)**
Enables/disables showing the row headers.
- **FL_Color row_header_color ()**
Returns the current row header color.
- **void row_header_color (FL_Color val)**
Sets the row header color and causes the screen to redraw.
- **int row_header_width ()**
Returns the current row header width (in pixels).
- **void row_header_width (int width)**
Sets the row header width to n and causes the screen to redraw.
- **int row_height (int row)**
Returns the current height of the specified row as a value in pixels.
- **void row_height (int row, int height)**
Sets the height of the specified row in pixels, and the table is redrawn.
- **void row_height_all (int height)**
Convenience method to set the height of all rows to the same value, in pixels.
- **int row_position ()**
Returns the current row scroll position as a row number.

- void **row_position** (int row)
Sets the vertical scroll position so 'row' is at the top, and causes the screen to redraw.
- int **row_resize** ()
Returns if row resizing by the user is allowed.
- void **row_resize** (int flag)
Allows/disallows row resizing by the user.
- int **row_resize_min** ()
Returns the current row minimum resize value.
- void **row_resize_min** (int val)
Sets the current row minimum resize value.
- int **rows** ()
Returns the number of rows in the table.
- int **scrollbar_size** () const
Gets the current size of the scrollbars' troughs, in pixels.
- void **scrollbar_size** (int newSize)
Sets the pixel size of the scrollbars' troughs to newSize, in pixels.
- void **set_selection** (int row_top, int col_left, int row_bot, int col_right)
Sets the region of cells to be selected (highlighted).
- int **tab_cell_nav** () const
Get state of table's 'Tab' key cell navigation flag.
- void **tab_cell_nav** (int val)
Flag to control if Tab navigates table cells or not.
- void **table_box** (FL_Boxtype val)
Sets the kind of box drawn around the data table, the default being FL_NO_BOX.
- **FL_Boxtype table_box** (void)
Returns the current box type used for the data table.
- int **top_row** ()
Returns the current top row shown in the table.
- void **top_row** (int row)
Sets which row should be at the top of the table, scrolling as necessary, and the table is redrawn.
- void **visible_cells** (int &r1, int &r2, int &c1, int &c2)
Returns the range of row and column numbers for all visible and partially visible cells in the table.
- void **when** (FL_When flags)
The FL_Widget::when() function is used to set a group of flags, determining when the widget callback is called:
- **~FL_Table** ()
The destructor for FL_Table.

Public Member Functions inherited from **FL_Group**

- **FL_Widget *& _ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** (FL_Widget &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** (FL_Widget *o)
See void FL_Group::add(FL_Widget &w)
- void **add_resizable** (FL_Widget &o)
Adds a widget to the group and makes it the resizable widget.
- **FL_Widget *const * array** () const
Returns a pointer to the internal array of children.
- **FL_Group const * as_group** () const **FL_OVERRIDE**
- **FL_Group * as_group** () **FL_OVERRIDE**

- Returns an *FL_Group* pointer if this widget is an *FL_Group*.

 - void **begin** ()

Sets the current group so you can build the widget tree by just constructing the widgets.
- *FL_Widget* * **child** (int n) const

Returns the *n*'th child.
- int **children** () const

Returns how many child widgets the group has.
- void **clear** ()

Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()

Returns the current clipping mode.
- void **clip_children** (int c)

Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)

Removes the widget at *index* from the group and deletes it.
- void **end** ()

Exactly the same as *current(this->parent())*.
- int **find** (const *FL_Widget* &o) const

See int *FL_Group::find(const FL_Widget *w) const*.
- int **find** (const *FL_Widget* *) const

Searches the children for the widget and returns the index.
- *FL_Group* (int, int, int, int, const char *s)

Creates a new *FL_Group* widget using the given position, size, and label string.
- void **focus** (*FL_Widget* *w)
- void **forms_end** ()

This is for forms compatibility only.
- void **init_sizes** ()

Resets the internal array of widget sizes and positions.
- void **insert** (*FL_Widget* &, int i)

The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** (*FL_Widget* &o, *FL_Widget* *before)

This does *insert(w, find(before))*.
- void **remove** (*FL_Widget* &)

Removes a widget from the group but does not delete it.
- void **remove** (*FL_Widget* *o)

Removes the widget *o* from the group.
- void **remove** (int index)

Removes the widget at *index* from the group but does not delete it.
- *FL_Widget* * **resizable** () const

Returns the group's resizable widget.
- void **resizable** (*FL_Widget* &o)

Sets the group's resizable widget.
- void **resizable** (*FL_Widget* *o)

The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual ~*FL_Group* ()

The destructor also deletes all the children.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

- Marks the widget as inactive without sending events or changing focus.*

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*

 - void `clear_damage` (uchar c=0)
- Clears or sets the damage flags.*

 - void `clear_output` ()
- Sets a widget to accept input.*

 - void `clear_visible` ()
- Hides the widget.*

 - void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*

 - `FL_Color` `color` () const
- Gets the background color of the widget.*

 - void `color` (FL_Color bg)
- Sets the background color of the widget.*

 - void `color` (FL_Color bg, FL_Color sel)
- Sets the background and selection color of the widget.*

 - `FL_Color` `color2` () const
- For back compatibility only.*

 - void `color2` (unsigned a)
- For back compatibility only.*

 - int `contains` (const FL_Widget *w) const
- Checks if w is a child of this widget.*

 - void `copy_label` (const char *new_label)
- Sets the current label.*

 - void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*

 - uchar `damage` () const
- Returns non-zero if `draw()` needs to be called.*

 - void `damage` (uchar c)
- Sets the damage bits for the widget.*

 - void `damage` (uchar c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int `damage_resize` (int, int, int, int)
- Internal use only.*

 - void `deactivate` ()
- Deactivates the widget.*

 - `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*

 - const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*

 - void `deimage` (FL_Image &img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - void `deimage` (FL_Image *img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*

 - void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)
- Calls the widget callback function with default arguments.*

 - void `do_callback` (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)
- Calls the widget callback function with arbitrary arguments.*

- void `do_callback` (`FL_Widget *widget`, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image * image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image &img`)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image *img`)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget *wgt`) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize labelsz` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype labeltype` () const
Gets the label type.
- void `labeltype` (`FL_Labeltype` a)

- Sets the label type.*

 - void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `Fl_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`Fl_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- `Fl_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`Fl_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.
- void `set_visible` ()

Makes the widget visible.
- void `set_visible_focus` ()

Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()

Makes a widget visible.
- void `size` (int W, int H)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const

Gets the current tooltip text.
- void `tooltip` (const char *text)

Sets the current tooltip text.

- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual ~[Fl_Widget](#) ()
Destroys the widget.

Protected Member Functions

- int [find_cell](#) ([TableContext](#) context, int R, int C, int &X, int &Y, int &W, int &H)
- int [handle](#) (int event) [FL_OVERRIDE](#)
Handle FLTK events.

Protected Member Functions inherited from FI_Table

- void **change_cursor** (FI_Cursor newcursor)
Change mouse cursor to different type.
- long **col_scroll_position** (int col)
Returns the scroll position (in pixels) of the specified column 'col'.
- **TableContext** **cursor2rowcol** (int &R, int &C, ResizeFlag &resizeflag)
Find row/col for the recent mouse event.
- void **damage_zone** (int r1, int c1, int r2, int c2, int r3=0, int c3=0)
Sets the damage zone to the specified row/col values.
- void **draw** () **FL_OVERRIDE**
Draws the entire FI_Table.
- virtual void **draw_cell** (TableContext context, int R=0, int C=0, int X=0, int Y=0, int W=0, int H=0)
Subclass should override this method to handle drawing the cells.
- int **find_cell** (TableContext context, int R, int C, int &X, int &Y, int &W, int &H)
Find a cell's X/Y/W/H region for the specified cell in row 'R', column 'C'.
- void **get_bounds** (TableContext context, int &X, int &Y, int &W, int &H)
Returns the (X,Y,W,H) bounding region for the specified 'context'.
- int **is_fltk_container** ()
Does the table contain any child fltk widgets?
- void **recalc_dimensions** ()
Recalculate the dimensions of the table, and affect any children.
- void **redraw_range** (int topRow, int botRow, int leftCol, int rightCol)
Define region of cells to be redrawn by specified range of rows/cols, and then sets damage(DAMAGE_CHILD).
- int **row_col_clamp** (TableContext context, int &R, int &C)
Return specified row/col values R and C to within the table's current row/col limits.
- long **row_scroll_position** (int row)
Returns the scroll position (in pixels) of the specified 'row'.
- void **table_resized** ()
Call this if table was resized, to recalculate internal data.
- void **table_scrolled** ()
Recalculate internals after a scroll.

Protected Member Functions inherited from FI_Group

- FI_Rect * **bounds** ()
Returns the internal array of widget sizes and positions.
- void **draw_child** (FI_Widget &widget) const
Forces a child to redraw.
- void **draw_children** ()
Draws all children of the group.
- void **draw_outside_label** (const FI_Widget &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int **on_insert** (FI_Widget *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int **on_move** (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void **on_remove** (int)
Allow derived groups to act when a child widget is removed from the group.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** (FI_Widget &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * **current** ()
Returns the currently active group.
- static void **current** ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Table](#)

- enum **ResizeFlag** {
RESIZE_NONE = 0 , **RESIZE_COL_LEFT** = 1 , **RESIZE_COL_RIGHT** = 2 , **RESIZE_ROW_ABOVE** = 3 ,
RESIZE_ROW_BELOW = 4 }

Protected Types inherited from [Fl_Widget](#)

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }

flags possible values enumeration.

Static Protected Member Functions inherited from [Fl_Table](#)

- static void **scroll_cb** ([Fl_Widget](#) *, void *)
Callback for when someone moves a scrollbar.

Protected Attributes inherited from [Fl_Table](#)

- int **botrow**
bottom row# of currently visible table on screen
- int **current_col**
selection cursor's current column (-1 if none)
- int **current_row**
selection cursor's current row (-1 if none)
- [Fl_Scrollbar](#) * **hscrollbar**
child horizontal scrollbar widget
- int **leftcol**
left column# of currently visible table on screen
- int **leftcol_scrollpos**
precomputed scroll position for left column
- int **rightcol**
right column# of currently visible table on screen
- int **select_col**
extended selection column (-1 if none)
- int **select_row**
extended selection row (-1 if none)
- [Fl_Scroll](#) * **table**
child [Fl_Scroll](#) widget container for child fltk widgets (if any)
- int **table_h**
table's virtual height (in pixels)
- int **table_w**
table's virtual width (in pixels)

- int **tih**
Data table's inner h dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **tiw**
Data table's inner w dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **tix**
Data table's inner x dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **tiy**
Data table's inner y dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **toh**
Data table's outer h dimension, outside bounding box. See [Table Dimension Diagram](#).
- int **toprow**
top row# of currently visible table on screen
- int **toprow_scrollpos**
precomputed scroll position for top row
- int **tow**
Data table's outer w dimension, outside bounding box. See [Table Dimension Diagram](#).
- int **tox**
Data table's outer x dimension, outside bounding box. See [Table Dimension Diagram](#).
- int **toy**
Data table's outer y dimension, outside bounding box. See [Table Dimension Diagram](#).
- [Fl_Scrollbar](#) * **vscrollbar**
child vertical scrollbar widget
- int **wih**
Table widget's inner h dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **wiw**
Table widget's inner w dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **wix**
Table widget's inner x dimension, inside bounding box. See [Table Dimension Diagram](#).
- int **wiy**
Table widget's inner y dimension, inside bounding box. See [Table Dimension Diagram](#).

11.140.1 Detailed Description

A table with row selection capabilities.

This class implements a simple table with the ability to select rows. This widget is similar to an [Fl_Browser](#) with columns. Most methods of importance will be found in the [Fl_Table](#) widget, such as [Fl_Table::rows\(\)](#) and [Fl_Table::cols\(\)](#).

To be useful it must be subclassed and at minimum the [draw_cell\(\)](#) method must be overridden to provide the content of the cells. This widget does *not* manage the cell's data content; it is up to the parent class's [draw_cell\(\)](#) method override to provide this.

Events on the cells and/or headings generate callbacks when they are clicked by the user. You control when events are generated based on the values you supply for [Fl_Table::when\(\)](#).

11.140.2 Constructor & Destructor Documentation

11.140.2.1 [Fl_Table_Row\(\)](#)

```
Fl_Table_Row::Fl_Table_Row (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0) [inline]
```

The constructor for the [Fl_Table_Row](#).

This creates an empty table with no rows or columns, with headers and row/column resize behavior disabled.

11.140.2.2 ~Fl_Table_Row()

```
Fl_Table_Row::~Fl_Table_Row () [inline]
```

The destructor for the [Fl_Table_Row](#).

Destroys the table and its associated widgets.

11.140.3 Member Function Documentation

11.140.3.1 clear()

```
void Fl_Table_Row::clear () [inline], [virtual]
```

Clears the table to zero rows (rows(0)), zero columns (cols(0)), and clears any widgets (table->[clear\(\)](#)) that were added with begin()/end() or [add\(\)](#)/insert()/etc.

See also

[rows\(int\)](#), [cols\(int\)](#)

Reimplemented from [Fl_Table](#).

11.140.3.2 handle()

```
int Fl_Table_Row::handle (
    int e) [protected], [virtual]
```

Handle FLTK events.

Reimplemented from [Fl_Table](#).

11.140.3.3 row_selected()

```
int Fl_Table_Row::row_selected (
    int row)
```

Checks to see if 'row' is selected.

Returns 1 if selected, 0 if not. You can change the selection of a row by clicking on it, or by using [select_row\(row, flag\)](#)

`row` **should** be a valid row. If the row is out of range the return value is 0 (zero).

Parameters

<code>in</code>	<code>row</code>	row to be checked

Returns

whether given row is selected

Return values

<code>1</code>	row is selected
<code>0</code>	row is not selected or <code>row</code> is out of range

11.140.3.4 rows()

```
void Fl_Table_Row::rows (
    int val) [virtual]
```

Sets the number of rows in the table, and the table is redrawn.

Reimplemented from [Fl_Table](#).

11.140.3.5 select_all_rows()

```
void Fl_Table_Row::select_all_rows (
    int flag = 1)
```

This convenience function changes the selection state for *all* rows based on 'flag'.
0=deselect, 1=select, 2=toggle existing state.

11.140.3.6 select_row()

```
int Fl_Table_Row::select_row (
    int row,
    int flag = 1)
```

Changes the selection state for '*row*', depending on the value of '*flag*'.
The optional *flag* can be:

- 0: clear selection
- 1: set selection (default)
- 2: toggle selection

Parameters

in	<i>row</i>	row to be selected, deselected, or toggled
in	<i>flag</i>	change mode, see description

Returns

result of modification, see description

Return values

0	selection state did not change
1	selection state changed
-1	row out of range or incorrect selection mode (<i>flag</i>)

11.140.3.7 type()

```
void Fl_Table_Row::type (
    TableRowSelectMode val)
```

Sets the table selection mode.

- `Fl_Table_Row::SELECT_NONE` - No selection allowed
- `Fl_Table_Row::SELECT_SINGLE` - Only single rows can be selected
- `Fl_Table_Row::SELECT_MULTI` - Multiple rows can be selected

The documentation for this class was generated from the following files:

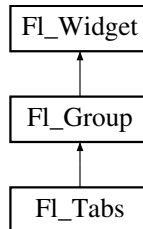
- `Fl_Table_Row.H`
- `Fl_Table_Row.cxx`

11.141 Fl_Tabs Class Reference

The [Fl_Tabs](#) widget is a container widget that displays a set of tabs, with each tab representing a different child widget.

```
#include <Fl_Tabs.H>
```

Inheritance diagram for Fl_Tabs:



Public Types

- enum { [OVERFLOW_COMPRESS](#) = 0 , [OVERFLOW_CLIP](#) , [OVERFLOW_PULLDOWN](#) , [OVERFLOW_DRAG](#) }

Public Member Functions

- void [client_area](#) (int &rx, int &ry, int &rw, int &rh, int tabh=0)
Returns the position and size available to be used by its children.
- [Fl_Tabs](#) (int X, int Y, int W, int H, const char *L=0)
Creates a new [Fl_Tabs](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handle all events in the tabs area and forward the rest to the selected child.
- void [handle_overflow](#) (int ov)
Set a method to handle an overflowing tab bar.
- [Fl_Widget](#) * [push](#) () const
Returns the tab group for the tab the user has currently down-clicked on and remains over until [FL_RELEASE](#).
- int [push](#) ([Fl_Widget](#) *)
This is called by the tab widget's [handle\(\)](#) method to set the tab group widget the user last [FL_PUSH](#)'ed on.
- void [resize](#) (int, int, int, int) [FL_OVERRIDE](#)
Make sure that we redraw all tabs when the widget size changes.
- void [show](#) () [FL_OVERRIDE](#)
Ensure proper placement of selected tab.
- [Fl_Align](#) [tab_align](#) () const
Gets the tab label alignment.
- void [tab_align](#) ([Fl_Align](#) a)
Sets the tab label alignment.
- [Fl_Widget](#) * [value](#) ()
Gets the currently visible widget/tab.
- int [value](#) ([Fl_Widget](#) *)
Sets the widget to become the currently visible widget/tab.
- virtual [Fl_Widget](#) * [which](#) (int event_x, int event_y)
Return a pointer to the child widget with a tab at the given coordinates.
- ~[Fl_Tabs](#) () [FL_OVERRIDE](#)
Delete allocated resources and destroy all children.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *[_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void [add](#) ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void [add](#) ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void [add_resizable](#) ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * [array](#) () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * [as_group](#) () const [FL_OVERRIDE](#)
- [FL_Group](#) * [as_group](#) () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void [begin](#) ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * [child](#) (int n) const
Returns the n'th child.
- int [children](#) () const
Returns how many child widgets the group has.
- void [clear](#) ()
Deletes all child widgets from memory recursively.
- unsigned int [clip_children](#) ()
Returns the current clipping mode.
- void [clip_children](#) (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int [delete_child](#) (int n)
*Removes the widget at *index* from the group and deletes it.*
- void [end](#) ()
*Exactly the same as *current(this->parent())*.*
- int [find](#) (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\) const](#).*
- int [find](#) (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char *s=0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void [focus](#) ([FL_Widget](#) *W)
- void [forms_end](#) ()
This is for forms compatibility only.
- void [init_sizes](#) ()
Resets the internal array of widget sizes and positions.
- void [insert](#) ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void [insert](#) ([FL_Widget](#) &o, [FL_Widget](#) *before)
*This does *insert(w, find(before))*.*
- void [remove](#) ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void [remove](#) ([FL_Widget](#) *o)
Removes the widget o from the group.
- void [remove](#) (int index)

- Removes the widget at `index` from the group but does not delete it.*
- `FL_Widget * resizable () const`
 - Returns the group's resizable widget.*
- `void resizable (FL_Widget &o)`
 - Sets the group's resizable widget.*
- `void resizable (FL_Widget *o)`
 - The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
- `virtual ~FL_Group ()`
 - The destructor also deletes all the children.*

Public Member Functions inherited from FL_Widget

- `void _clear_fullscreen ()`
- `void _set_fullscreen ()`
- `void activate ()`
 - Activates the widget.*
- `unsigned int active () const`
 - Returns whether the widget is active.*
- `int active_r () const`
 - Returns whether the widget and all of its parents are active.*
- `FL_Align align () const`
 - Gets the label alignment.*
- `void align (FL_Align alignment)`
 - Sets the label alignment.*
- `long argument () const`
 - Gets the current user data (long) argument that is passed to the callback function.*
- `void argument (long v)`
 - Sets the current user data (long) argument that is passed to the callback function.*
- `virtual class FL_Gl_Window * as_gl_window ()`
 - Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.*
- `virtual class FL_Gl_Window const * as_gl_window () const`
- `virtual FL_Window * as_window ()`
 - Returns an `FL_Window` pointer if this widget is an `FL_Window`.*
- `virtual FL_Window const * as_window () const`
- `void bind_deimage (FL_Image *img)`
 - Sets the image to use as part of the widget label when in the inactive state.*
- `void bind_deimage (int f)`
 - Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- `void bind_image (FL_Image *img)`
 - Sets the image to use as part of the widget label when in the active state.*
- `void bind_image (int f)`
 - Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- `FL_Boxtype box () const`
 - Gets the box type of the widget.*
- `void box (FL_Boxtype new_box)`
 - Sets the box type for the widget.*
- `FL_Callback_p callback () const`
 - Gets the current callback function for the widget.*
- `void callback (FL_Callback *cb)`
 - Sets the current callback function for the widget.*
- `void callback (FL_Callback *cb, FL_Callback_User_Data *p, bool auto_free)`

- Sets the current callback function and managed user data for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.

 - void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.

 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.

 - unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.

 - void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.

 - void [clear_changed](#) ()

Marks the value of the widget as unchanged.

 - void [clear_damage](#) ([uchar](#) c=0)

Clears or sets the damage flags.

 - void [clear_output](#) ()

Sets a widget to accept input.

 - void [clear_visible](#) ()

Hides the widget.

 - void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.

 - [FI_Color](#) [color](#) () const

Gets the background color of the widget.

 - void [color](#) ([FI_Color](#) bg)

Sets the background color of the widget.

 - void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)

Sets the background and selection color of the widget.

 - [FI_Color](#) [color2](#) () const

For back compatibility only.

 - void [color2](#) (unsigned a)

For back compatibility only.

 - int [contains](#) (const [FI_Widget](#) *w) const

Checks if w is a child of this widget.

 - void [copy_label](#) (const char *new_label)

Sets the current label.

 - void [copy_tooltip](#) (const char *text)

Sets the current tooltip text.

 - [uchar](#) [damage](#) () const

Returns non-zero if [draw\(\)](#) needs to be called.

 - void [damage](#) ([uchar](#) c)

Sets the damage bits for the widget.

 - void [damage](#) ([uchar](#) c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.

 - int [damage_resize](#) (int, int, int, int)

Internal use only.

 - void [deactivate](#) ()

Deactivates the widget.

 - [FI_Image](#) * [deimage](#) ()

Gets the image that is used as part of the widget label when in the inactive state.

 - const [FI_Image](#) * [deimage](#) () const

Gets the image that is used as part of the widget label when in the inactive state.

- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const

- Gets the font to use.*

 - void [labelfont](#) ([FI_Font](#) f)

Sets the font to use.
- [FI_Fontsize](#) [labelsize](#) () const

Gets the font size in pixels.
- void [labelsize](#) ([FI_Fontsize](#) pix)

Sets the font size in pixels.
- [FI_Labeltype](#) [labeltype](#) () const

Gets the label type.
- void [labeltype](#) ([FI_Labeltype](#) a)

Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FI_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FI_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- [FI_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.

- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual [~Fl_Widget](#) ()
Destroys the widget.

Protected Member Functions

- void **check_overflow_menu** ()
Check if the tabs overflow and sets the has_overflow_menu flag accordingly.
- virtual void **clear_tab_positions** ()
Clear internal array of tab positions and widths.
- void **draw** () **FL_OVERRIDE**
Draw the tabs area, the optional pulldown button, and all children.
- void **draw_overflow_menu_button** ()
Draw square button-like graphics with a down arrow in the top or bottom right corner.
- virtual void **draw_tab** (int x1, int x2, int W, int H, **FI_Widget** *o, int flags, int sel)
Draw a tab in the top or bottom tabs area.
- void **handle_overflow_menu** ()
This is called when the user clicks the overflow pulldown menu button.
- virtual int **hit_close** (**FI_Widget** *o, int event_x, int event_y)
Check whether the coordinates fall within the "close" button area of the tab.
- virtual int **hit_overflow_menu** (int event_x, int event_y)
Determine if the coordinates are in the area of the overflow menu button.
- virtual int **hit_tabs_area** (int event_x, int event_y)
Determine if the coordinates are within the tabs area.
- int **maybe_do_callback** (**FI_Widget** *o)
Set tab o as selected and call callbacks if needed.
- int **on_insert** (**FI_Widget** *, int) **FL_OVERRIDE**
Make sure that we redraw all tabs when new children are added.
- int **on_move** (int, int) **FL_OVERRIDE**
Make sure that we redraw all tabs when children are moved.
- void **on_remove** (int) **FL_OVERRIDE**
Make sure that we redraw all tabs when new children are removed.
- virtual void **redraw_tabs** ()
Redraw all tabs (and only the tabs).
- virtual int **tab_height** ()
Return space (height) in pixels usable for tabs.
- virtual int **tab_positions** ()
Calculate tab positions and widths.
- void **take_focus** (**FI_Widget** *o)
Take keyboard focus if o is not NULL.

Protected Member Functions inherited from **FI_Group**

- **FI_Rect** * **bounds** ()
Returns the internal array of widget sizes and positions.
- void **draw_child** (**FI_Widget** &widget) const
Forces a child to redraw.
- void **draw_children** ()
Draws all children of the group.
- void **draw_outside_label** (const **FI_Widget** &widget) const
Parents normally call this to draw outside labels of child widgets.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** (**FI_Widget** &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from Fl_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (Fl_Boxtype t, Fl_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **Fl_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Protected Attributes

- int **has_overflow_menu**
set in OVERFLOW_PULLDOWN mode if tabs overflow. The actual menu array is created only on demand
- int **overflow_type**
- **Fl_Align** **tab_align_**
tab label alignment
- int **tab_count**
Array size of tab positions etc.
- int * **tab_flags**
Array of tab flag of tabs per child.
- int **tab_offset**
for pulldown and drag overflow, this is the horizontal offset when the tabs bar is dragged by the user
- int * **tab_pos**
Array of x-offsets of tabs per child + 1.
- int * **tab_width**
Array of widths of tabs per child.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * [current](#) ()
Returns the currently active group.
- static void [current](#) ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [FI_Widget](#)

- static void [default_callback](#) ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.141.1 Detailed Description

The [FI_Tabs](#) widget is a container widget that displays a set of tabs, with each tab representing a different child widget.

The user can select a tab by clicking on it, and the corresponding child widget will be displayed. The [FI_Tabs](#) widget is useful for organizing a large number of controls or other widgets into a compact space, allowing the user to switch between different sets of controls as needed.

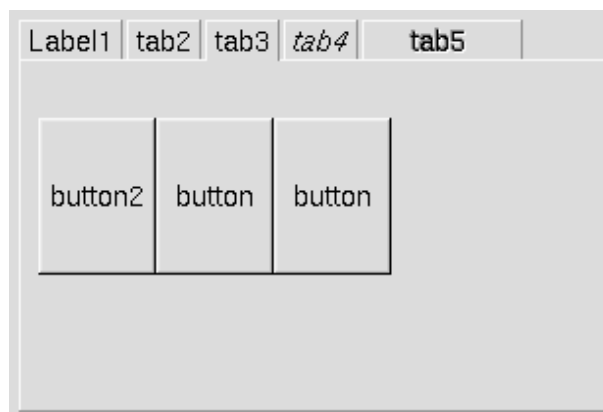


Figure 11.50 [FI_Tabs](#)

Clicking the tab makes a child `visible()` by calling `show()` on it, and all other children are made invisible by calling `hide()` on them. Usually the children are `Fl_Group` widgets containing several widgets themselves.

Each child makes a card, and its `label()` is printed on the card tab, including the label font and style. The selection color of that child is used to color the tab, while the color of the child determines the background color of the pane. '&' in labels are used to prefix a shortcut that is drawn underlined and that activates the corresponding tab; repeated '&&' avoids that.

The size of the tabs is controlled by the bounding box of the children (there should be some space between the children and the edge of the `Fl_Tabs`), and the tabs may be placed "inverted" on the bottom - this is determined by which gap is larger. It is easiest to lay this out in FLUID, using the FLUID browser to select each child group and resize them until the tabs look the way you want them to.

Note: The widgets contained in each child should leave some clear space (five pixels as of FLTK 1.4.x) at the top or bottom of the group (where the tabs are displayed). Otherwise drawing the children may interfere with the selection border between the tabs and the children. This is particularly important if the child group is an `Fl_Scroll` widget: either the `Fl_Scroll` widget must be inset by five pixels relative to other children or it can be wrapped inside another `Fl_Group` and inset by five pixels within this group so the contents of the `Fl_Scroll` widget are kept away from the tabs by this amount.

The background area behind and to the right of the tabs is "transparent", exposing the background detail of the parent. The value of `Fl_Tabs::box()` does not affect this area. So if `Fl_Tabs` is resized by itself without the parent, force the appropriate parent (visible behind the tabs) to `redraw()` to prevent artifacts.

See "Resizing Caveats" below on how to keep tab heights constant. See "Callback's Use Of `when()`" on how to control the details of how clicks invoke the `callback()`.

A typical use of the `Fl_Tabs` widget:

```
// Typical use of Fl_Tabs
Fl_Tabs *tabs = new Fl_Tabs(10,10,300,200);
{
    Fl_Group *grp1 = new Fl_Group(20,30,280,170,"Tab1");
    {
        ..widgets that go in tab#1..
    }
    grp1->end();
    Fl_Group *grp2 = new Fl_Group(20,30,280,170,"Tab2");
    {
        ..widgets that go in tab#2..
    }
    grp2->end();
}
tabs->end();
```

Default Appearance

The appearance of each "tab" is taken from the `label()` and `color()` of the child group corresponding to that "tab" and panel. Where the "tabs" appear depends on the position and size of the child groups that make up the panels within the `Fl_Tabs` widget, i.e. whether there is more space above or below them. The height of the "tabs" depends on how much free space is available.

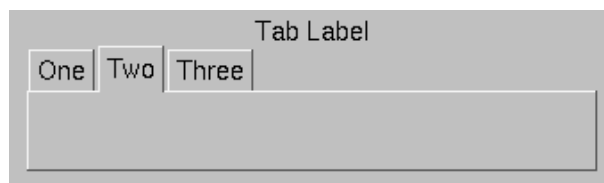


Figure 11.51 `Fl_Tabs` Default Appearance

Highlighting The Selected Tab

The selected "tab" can be highlighted further by setting the `selection_color()` of the `Fl_Tab` itself, e.g.

```

..
tabs = new Fl_Tabs(..);
tabs->selection_color(FL_DARK3);
..

```

The result of the above looks like:

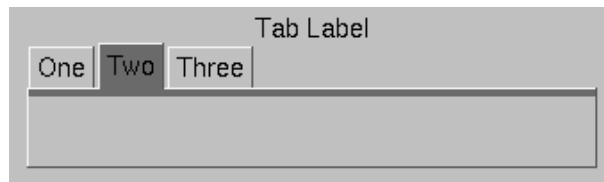


Figure 11.52 Highlighting the selected tab

Uniform Tab and Panel Appearance

In order to have uniform tab and panel appearance, not only must the `color()` and `selection_color()` for each child group be set, but also the `selection_color()` of the `Fl_Tab` itself any time a new "tab" is selected. This can be achieved within the `Fl_Tab` callback, e.g.

```

void MyTabCallback(Fl_Widget *w, void*) {
    Fl_Tabs *tabs = (Fl_Tabs*)w;
    // When tab changed, make sure it has same color as its group
    tabs->selection_color( (tabs->value())->color() );
}
..
int main(..) {
    // Define tabs widget
    tabs = new Fl_Tabs(..);
    tabs->callback(MyTabCallback);

    // Create three tabs each colored differently
    grp1 = new Fl_Group(.. "One");
    grp1->color(9);
    grp1->selection_color(9);
    grp1->end();

    grp2 = new Fl_Group(.. "Two");
    grp2->color(10);
    grp2->selection_color(10);
    grp2->end();

    grp3 = new Fl_Group(.. "Three");
    grp3->color(14);
    grp3->selection_color(14);
    grp3->end();
    ..
    // Make sure default tab has same color as its group
    tabs->selection_color( (tab->value())->color() );
    ..
    return Fl::run();
}

```

The result of the above looks like:



Figure 11.53 `Fl_Tabs` with uniform colors

If `Fl_Tabs` has no children, the widget will be drawn as a flat rectangle in the background color set by `color()`.

Close Button on Tabs

The `Fl_Tabs` widget allows you to specify that a child widget should display a close button in its tab. If the `FL_WHEN_CLOSED` flag is set for the child widget, an "X" symbol will be displayed to the left of the label text in the tab. When the close button is clicked, the child widget's callback function will be called with the `FL_REASON_CLOSED` reason. It is then the responsibility of the child widget to remove itself from the `Fl_Tabs` container.

Tabs that are in a compressed state will not display a close button until they are fully expanded.

Overflowing Tabs

When the combined width of the tabs exceeds that of the `Fl_Tabs` widget, the tabs will overflow. `Fl_Tabs` provides four options for managing tabs overflow:

- `Fl_Tabs::OVERFLOW_COMPRESS`: proportionally compress the tabs to the left and right of the selected tab until they all fit within the widget.
- `Fl_Tabs::OVERFLOW_CLIP`: clip any tabs that extend beyond the right edge of the `Fl_Tabs` widget, making some tabs unreachable.
- `Fl_Tabs::OVERFLOW_PULLDOWN`: don't compress the tabs but instead generate a pulldown menu at the right end of the tabs area, displaying all available tabs.
- `Fl_Tabs::OVERFLOW_DRAG`: maintain the tabs' original sizes, allowing horizontal dragging of the tabs area using the mouse, a horizontal mouse wheel, or the horizontal scrolling gesture on touchpads.

Resizing Caveats

When `Fl_Tabs` is resized vertically, the default behavior scales the tab's height as well as its children. To keep the tab height constant during resizing, set the tab widget's `resizable()` to one of the tab's child groups, i.e.

```
tabs = new Fl_Tabs(..);
grp1 = new Fl_Group(..);
..
grp2 = new Fl_Group(..);
..
tabs->end();
tabs->resizable(grp1);    // keeps tab height constant
```

Callback's Use Of `when()`

As of FLTK 1.3.3, `Fl_Tabs()` supports the following flags for `when()`:

- `FL_WHEN_NEVER` – callback never invoked (all flags off)
- `FL_WHEN_CHANGED` – if flag set, invokes callback when a tab has been changed (on click or keyboard navigation)
- `FL_WHEN_NOT_CHANGED` – if flag set, invokes callback when the tabs remain unchanged (on click or keyboard navigation)
- `FL_WHEN_RELEASE` – if flag set, invokes callback on RELEASE of mouse button or keyboard navigation

Notes:

1. The above flags can be logically OR-ed (`|`) or added (`+`) to combine behaviors.
2. The default value for `when()` is `FL_WHEN_RELEASE` (inherited from `Fl_Widget`).
3. If `FL_WHEN_RELEASE` is the *only* flag specified, the behavior will be as if (`FL_WHEN_RELEASE|FL_WHEN_CHANGED`) was specified.
4. The value of `changed()` will be valid during the callback.
5. If both `FL_WHEN_CHANGED` and `FL_WHEN_NOT_CHANGED` are specified, the callback is invoked whether the tab has been changed or not. The `changed()` method can be used to determine the cause.
6. `FL_WHEN_NOT_CHANGED` can happen if someone clicks on an already selected tab, or if a keyboard navigation attempt results in no change to the tabs, such as using the arrow keys while at the left or right end of the tabs.
7. `Fl::callback_reason()` returns `FL_REASON_SELECTED` or `FL_REASON_RESELECTED`

11.141.2 Member Enumeration Documentation

11.141.2.1 anonymous enum

```
anonymous enum
```

Enumerator

OVERFLOW_COMPRESS	Tabs will be compressed and overlaid on top of each other.
OVERFLOW_CLIP	Only the first tabs that fit will be displayed.
OVERFLOW_PULLDOWN	Tabs that do not fit will be placed in a pull-down menu.
OVERFLOW_DRAG	The tab bar can be dragged horizontally to reveal additional tabs.

11.141.3 Constructor & Destructor Documentation

11.141.3.1 Fl_Tabs()

```
Fl_Tabs::Fl_Tabs (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new [Fl_Tabs](#) widget using the given position, size, and label string.

The default boxtype is FL_THIN_UP_BOX.

Use [add\(Fl_Widget*\)](#) to add each child, which are usually [Fl_Group](#) widgets. The children should be sized to stay away from the top or bottom edge of the [Fl_Tabs](#) widget, which is where the tabs will be drawn.

All children of [Fl_Tabs](#) should have the same size and exactly fit on top of each other. They should only leave space above or below where the tabs will go, but not on the sides. If the first child of [Fl_Tabs](#) is set to "resizable()", the riders will not resize when the tabs are resized.

The destructor *also deletes all the children*. This allows a whole tree to be deleted at once, without having to keep a pointer to all the children in the user code. A kludge has been done so the [Fl_Tabs](#) and all of its children can be automatic (local) variables, but you must declare the [Fl_Tabs](#) widget *first* so that it is destroyed last.

11.141.4 Member Function Documentation

11.141.4.1 clear_tab_positions()

```
void Fl_Tabs::clear_tab_positions () [protected], [virtual]
```

Clear internal array of tab positions and widths.

See also

[tab_positions\(\)](#).

11.141.4.2 client_area()

```
void Fl_Tabs::client_area (
    int & rx,
    int & ry,
    int & rw,
    int & rh,
    int tabh = 0)
```

Returns the position and size available to be used by its children.

If there isn't any child yet the `tabh` parameter will be used to calculate the return values. This assumes that the children's labelsize is the same as the [Fl_Tabs](#)' labelsize and adds a small border.

If there are already children, the values of `child(0)` are returned, and `tabh` is ignored.

Note

Children should always use the same positions and sizes.

This function requires access to the display drivers to determine the selected font height. If `client_area` is invoked before the first window is displayed, ensure that [fl_open_display\(\)](#) is called beforehand.

`tabh` can be one of

- 0: calculate label size, tabs on top
- -1: calculate label size, tabs on bottom
- > 0: use given `tabh` value, tabs on top (height = `tabh`)
- < -1: use given `tabh` value, tabs on bottom (height = `-tabh`)

Parameters

in	<i>tabh</i>	position and optional height of tabs (see above)
out	<i>rx,ry,rw,rh</i>	(x,y,w,h) of client area for children

Since

FLTK 1.3.0

11.141.4.3 draw()

```
void Fl_Tabs::draw (
    void ) [protected], [virtual]
```

Draw the tabs area, the optional pulldown button, and all children.

Reimplemented from [Fl_Group](#).

11.141.4.4 draw_tab()

```
void Fl_Tabs::draw_tab (
    int x1,
    int x2,
    int W,
    int H,
    Fl_Widget * o,
    int flags,
    int what) [protected], [virtual]
```

Draw a tab in the top or bottom tabs area.

Tabs can be selected, or on the left or right side of the selected tab. If overlapping, left tabs are drawn bottom to top using clipping. The selected tab is then the topmost, followed by the right side tabs drawn top to bottom.

Tabs with the `FL_WHEN_CLOSE` bit set will draw a cross on their left side only if they are not compressed/overlapping.

Parameters

in	<i>x1</i>	horizontal position of the left visible edge of the tab
in	<i>x2</i>	horizontal position of the following tab
in	<i>W,H</i>	width and height of the tab
in	<i>o</i>	the child widget that corresponds to this tab
in	<i>flags</i>	if bit 1 is set, this tab is overlapped by another tab
in	<i>what</i>	can be LEFT, SELECTED, or RIGHT to indicate if the tab is to the left side or the right side of the selected tab, or the selected tab itself

11.141.4.5 handle()

```
int Fl_Tabs::handle (
    int event) [virtual]
```

Handle all events in the tabs area and forward the rest to the selected child.

Parameters

in	<i>event</i>	handle this event

Returns

1 if the event was handled

Reimplemented from [Fl_Group](#).

11.141.4.6 handle_overflow()

```
void Fl_Tabs::handle_overflow (
    int ov)
```

Set a method to handle an overflowing tab bar.

The [Fl_Tabs](#) widget allows you to specify how to handle the situation where there are more tabs than can be displayed at once. The available options are:

- `OVERFLOW_COMPRESS`: Tabs will be compressed and overlaid on top of each other.
- `OVERFLOW_CLIP`: Only the first tabs that fit will be displayed.
- `OVERFLOW_PULLDOWN`: Tabs that do not fit will be placed in a pull-down menu.
- `OVERFLOW_DRAG`: The tab bar can be dragged horizontally to reveal additional tabs.

You can set the desired behavior using the `overflow()` method.

Parameters

ov	overflow type

See also

[OVERFLOW_COMPRESS](#), [OVERFLOW_CLIP](#), [OVERFLOW_PULLDOWN](#), [OVERFLOW_DRAG](#)

11.141.4.7 handle_overflow_menu()

```
void Fl_Tabs::handle_overflow_menu () [protected]
```

This is called when the user clicks the overflow pulldown menu button.

This method creates a menu item array that contains the titles of all tabs in the [Fl_Tabs](#) group. Visible and invisible tabs are separated by dividers to indicate their state.

The menu is then presented until the user selects an item or cancels. The chosen tab is then selected and made visible.

The menu item array is then deleted.

11.141.4.8 hit_close()

```
int Fl_Tabs::hit_close (
    Fl_Widget * o,
    int event_x,
    int event_y) [protected], [virtual]
```

Check whether the coordinates fall within the "close" button area of the tab.

The [Fl_Tabs::hit_close\(\)](#) method checks whether the given event coordinates fall within the area of the "close" button on the tab of the specified child widget. This method should be called after the [Fl_Tabs::which\(\)](#) method, which updates a lookup table used to determine the width of each tab.

Parameters

<i>o</i>	check the tab of this widget

<i>event_x, event_y</i>	event coordinates

Returns

1 if we hit the close button, and 0 otherwise

11.141.4.9 hit_overflow_menu()

```
int Fl_Tabs::hit_overflow_menu (  
    int event_x,  
    int event_y) [protected], [virtual]
```

Determine if the coordinates are in the area of the overflow menu button.

Parameters

<i>event_x, event_y</i>	event coordinates

Returns

1 if we hit the overflow menu button, and 0 otherwise

11.141.4.10 hit_tabs_area()

```
int Fl_Tabs::hit_tabs_area (  
    int event_x,  
    int event_y) [protected], [virtual]
```

Determine if the coordinates are within the tabs area.

Parameters

<i>event_x, event_y</i>	event coordinates

Returns

1 if we hit the tabs area, and 0 otherwise

11.141.4.11 maybe_do_callback()

```
int Fl_Tabs::maybe_do_callback (  
    Fl_Widget * o) [protected]
```

Set tab o as selected and call callbacks if needed.

Parameters

in	o	the newly selected tab

Returns

0 if o is invalid or was deleted by the callback and must no longer be used

11.141.4.12 on_insert()

```
int Fl_Tabs::on_insert (
    Fl_Widget * candidate,
    int index) [protected], [virtual]
```

Make sure that we redraw all tabs when new children are added.

Reimplemented from [Fl_Group](#).

11.141.4.13 on_move()

```
int Fl_Tabs::on_move (
    int a,
    int b) [protected], [virtual]
```

Make sure that we redraw all tabs when children are moved.

Reimplemented from [Fl_Group](#).

11.141.4.14 on_remove()

```
void Fl_Tabs::on_remove (
    int index) [protected], [virtual]
```

Make sure that we redraw all tabs when new children are removed.

Reimplemented from [Fl_Group](#).

11.141.4.15 push() [1/2]

```
Fl_Widget * Fl_Tabs::push () const [inline]
```

Returns the tab group for the tab the user has currently down-clicked on and remains over until FL_RELEASE.

Otherwise, returns NULL.

While the user is down-clicked on a tab, the return value is the tab group for that tab. But as soon as the user releases, or drags off the tab with the button still down, the return value will be NULL.

See also

[push\(Fl_Widget*\)](#).

11.141.4.16 push() [2/2]

```
int Fl_Tabs::push (
    Fl_Widget * o)
```

This is called by the tab widget's [handle\(\)](#) method to set the tab group widget the user last FL_PUSH'ed on.

Set back to zero on FL_RELEASE.

As of this writing, the value is mainly used by [draw_tab\(\)](#) to determine whether or not to draw a 'down' box for the tab when it's clicked, and to turn it off if the user drags off it.

See also

[push\(\)](#).

11.141.4.17 redraw_tabs()

```
void Fl_Tabs::redraw_tabs () [protected], [virtual]
```

Redraw all tabs (and only the tabs).

This method sets the Fl_Tab's damage flags so the tab area is redrawn.

11.141.4.18 resize()

```
void Fl_Tabs::resize (
    int X,
    int Y,
```



```
int W,
int H) [virtual]
```

Make sure that we redraw all tabs when the widget size changes.
Reimplemented from [Fl_Group](#).

11.141.4.19 show()

```
void Fl_Tabs::show () [virtual]
```

Ensure proper placement of selected tab.
Reimplemented from [Fl_Widget](#).

11.141.4.20 tab_align() [1/2]

```
Fl_Align Fl_Tabs::tab_align () const [inline]
```

Gets the tab label alignment.

See also

[tab_align\(Fl_Align\)](#)

11.141.4.21 tab_align() [2/2]

```
void Fl_Tabs::tab_align (
    Fl_Align a) [inline]
```

Sets the tab label alignment.

The default is `FL_ALIGN_CENTER` so tab labels are centered, but since the label space is measured (per label) to fit the labels, there wouldn't be any difference if labels were aligned left or right.

If you want to show an image (icon) next to the group's label you can set a different label alignment. `FL_ALIGN_IMAGE_NEXT_TO_TEXT` is the recommended alignment to show the icon left of the text.

11.141.4.22 tab_height()

```
int Fl_Tabs::tab_height () [protected], [virtual]
```

Return space (height) in pixels usable for tabs.

The calculated height is the largest space between all children and the upper and lower widget boundaries, respectively. If the space at the bottom is larger than at the top, the value will be negative and the tabs should be placed at the bottom.

Returns

Vertical space that can be used for the tabs.

Return values

<code>>0</code>	To put the tabs at the top of the widget.
<code><0</code>	To put the tabs on the bottom.
<i>Full</i>	height, if <code>children() == 0</code> .

11.141.4.23 tab_positions()

```
int Fl_Tabs::tab_positions () [protected], [virtual]
```

Calculate tab positions and widths.

This protected method calculates the horizontal display positions and widths of all tabs. If the number of children 'nc' (see below) is `> 0` three internal arrays are allocated, otherwise the arrays are free'd and the pointers are set to NULL. Note that the first array is larger (nc+1).

- `tab_pos[nc+1]` : The left edges of each tab plus a fake left edge for a tab past the right-hand one.
- `tab_width[nc]` : The width of each tab

- `tab_flags[nc]` : Flags, bit 0 is set if the tab is compressed

If needed, these arrays are (re)allocated.

These positions are actually of the left edge of the slope. They are either separated by the correct distance or by EXTRASPACE or by zero.

In OVERFLOW_COMPRESS mode, tab positions and widths are compressed to make the entire tabs bar fit into the width of [Fl_Tabs](#) while keeping the selected tab fully visible.

In other overflow modes, the tabs area may be dragged horizontally using [tab_offset](#). The `tab_pos` array is not adjusted to the horizontal offset, but starts at this->x() plus the box's left margin.

The protected variable `tab_count` is set to the currently allocated size, i.e. the number of children (`nc`).

Returns

Index of the selected item

Return values

-1	If the number of children is 0 (zero).

Note

Return values in 1.3 were not documented. Return values before Sep 2023 were documented as 1 based index and 0 if there were no children. This was actually never the case. It always returned a 0 based index and the (useless) value of also 0 if there were no children. The current version returns -1 if there are no children.

For this method to work, only a single child should be selected. Calling the method [value\(\)](#) before calling [tab_positions\(\)](#) will ensure that exactly one child is selected and return a pointer to that child.

See also

[clear_tab_positions\(\)](#)

11.141.4.24 take_focus()

```
void Fl_Tabs::take_focus (
    Fl_Widget * o) [protected]
```

Take keyboard focus if `o` is not NULL.

Parameters

<code>in</code>	<code>o</code>	selected tab

11.141.4.25 value() [1/2]

```
Fl_Widget * Fl_Tabs::value ()
```

Gets the currently visible widget/tab.

The [Fl_Tabs::value\(\)](#) method returns a pointer to the currently visible child widget of the [Fl_Tabs](#) container. The visible child is the first child that is currently being displayed, or the last child if none of the children are being displayed.

If child widgets have been added, moved, or deleted, this method ensures that only one tab is visible at a time.

Returns

a pointer to the currently visible child

11.141.4.26 value() [2/2]

```
int Fl_Tabs::value (  
    Fl_Widget * newvalue)
```

Sets the widget to become the currently visible widget/tab.

The `Fl_Tabs::value()` method allows you to set a particular child widget of the `Fl_Tabs` container to be the currently visible widget. If the specified widget is a child of the `Fl_Tabs` container, it will be made visible and all other children will be hidden. The method returns 1 if the value was changed, and 0 if the specified value was already set.

Parameters

<code>in</code>	<code>newvalue</code>	a pointer to a child widget

Returns

- 1 if a different tab was chosen
- 0 if there was no change (new value already set)

11.141.4.27 which()

```
Fl_Widget * Fl_Tabs::which (
    int event_x,
    int event_y) [virtual]
```

Return a pointer to the child widget with a tab at the given coordinates.

The `Fl_Tabs::which()` method returns a pointer to the child widget of the `Fl_Tabs` container that corresponds to the tab at the given event coordinates. If the event coordinates are outside the area of the tabs or if the `Fl_Tabs` container has no children, the method returns NULL.

Parameters

<code>event_x, event_y</code>	event coordinates

Returns

- pointer to the selected child widget, or NULL

11.141.5 Member Data Documentation**11.141.5.1 overflow_type**

```
int Fl_Tabs::overflow_type [protected]
```

See also

[OVERFLOW_COMPRESS](#), [OVERFLOW_CLIP](#), etc.

11.141.5.2 tab_count

```
int Fl_Tabs::tab_count [protected]
```

Array size of tab positions etc.

See also

[tab_positions\(\)](#)

11.141.5.3 tab_flags

```
int* Fl_Tabs::tab_flags [protected]
```

Array of tab flag of tabs per child.

See also

[tab_positions\(\)](#)

11.141.5.4 tab_pos

`int* Fl_Tabs::tab_pos [protected]`
 Array of x-offsets of tabs per child.

See also

[tab_positions\(\)](#)

11.141.5.5 tab_width

`int* Fl_Tabs::tab_width [protected]`
 Array of widths of tabs per child.

See also

[tab_positions\(\)](#)

The documentation for this class was generated from the following files:

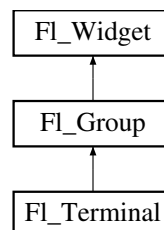
- `Fl_Tabs.H`
- `Fl_Tabs.cxx`

11.142 Fl_Terminal Class Reference

Terminal widget supporting Unicode/utf-8, ANSI/xterm escape codes with full RGB color control.

`#include <Fl_Terminal.H>`

Inheritance diagram for `Fl_Terminal`:



Classes

- class [CharStyle](#)
- class [Cursor](#)
- class [EscapeSeq](#)
- class [Margin](#)
- class [PartialUtf8Buf](#)
- class [RingBuffer](#)
- class [Selection](#)
- class [Utf8Char](#)

Public Types

- enum [Attrib](#) {
`NORMAL` = 0x00 , `BOLD` = 0x01 , `DIM` = 0x02 , `ITALIC` = 0x04 ,
`UNDERLINE` = 0x08 , `_RESERVED_1` = 0x10 , `INVERSE` = 0x20 , `_RESERVED_2` = 0x40 ,
`STRIKEOUT` = 0x80 }

Bits for the per-character attributes, which control text features such as italic, bold, underlined text, etc.

- enum [CharFlags](#) {
`FG_XTERM` = 0x01 , `BG_XTERM` = 0x02 , `EOL` = 0x04 , `RESV_A` = 0x08 ,
`RESV_B` = 0x10 , `RESV_C` = 0x20 , `RESV_D` = 0x40 , `RESV_E` = 0x80 ,
`COLORMASK` = (FG_XTERM | BG_XTERM) }

Per-character 8 bit flags (uchar) used to manage special states for characters.

- enum `OutFlags` { `OFF` = 0x00 , `CR_TO_LF` = 0x01 , `LF_TO_CR` = 0x02 , `LF_TO_CRLF` = 0x04 }

Output translation flags for special control character translations.

- enum `RedrawStyle` { `NO_REDRAW` =0 , `RATE_LIMITED` , `PER_WRITE` }

Determines when `FI_Terminal` calls `redraw()` if new text is added.

- enum `ScrollbarStyle` { `SCROLLBAR_OFF` = 0x00 , `SCROLLBAR_AUTO` = 0x01 , `SCROLLBAR_ON` = 0x02 }

Behavior of scrollbars.

Public Member Functions

- void `ansi` (bool val)
Enable/disable the ANSI mode flag.
- bool `ansi` (void) const
Return the state of the ANSI flag.
- void `append` (const char *s, int len=-1)
Appends string s to the terminal at the current cursor position using the current text color/attributes.
- void `append_ascii` (const char *s)
Append NULL terminated ASCII string to terminal, slightly more efficient than `append_utf8()`.
- void `append_utf8` (const char *buf, int len=-1)
Append NULL terminated UTF-8 string to terminal.
- void `box` (`FI_Boxtype` val)
Sets the box type, updates terminal margins et al.
- `FI_Boxtype` `box` (void) const
Returns the current box type.
- void `clear` (`FI_Color` val)
Clears the screen to a specific color val and homes the cursor.
- void `clear` (void)
Clears the screen to the current `textbgcolor()`, and homes the cursor.
- void `clear_history` (void)
Clears the scroll history buffer and adjusts scrollbar, forcing it to `redraw()`.
- void `clear_screen` (bool scroll_to_hist=true)
Clear the terminal screen only; does not affect the cursor position.
- void `clear_screen_home` (bool scroll_to_hist=true)
Clear the terminal screen and home the cursor.
- void `color` (`FI_Color` val)
Sets the background color for the terminal's `FI_Group::box()`.
- `FI_Color` `color` (void) const
Return base widget `FI_Group`'s `box()` color()
- int `cursor_col` (void) const
Return the cursor's current column position on the screen.
- void `cursor_home` (void)
Move cursor to the home position (top/left).
- int `cursor_row` (void) const
Return the cursor's current row position on the screen.
- void `cursorbgcolor` (`FI_Color` val)
Set the cursor's background color used for the cursor itself.
- `FI_Color` `cursorbgcolor` (void) const
Get the cursor's background color used for the cursor itself.
- void `cursorfgcolor` (`FI_Color` val)
Set the cursor's foreground color used for text under the cursor.
- `FI_Color` `cursorfgcolor` (void) const

- Get the cursor's foreground color used for text under the cursor.*

 - void [display_columns](#) (int val)

Set terminal's display width in columns of text characters.

 - int [display_columns](#) (void) const

Return terminal's display width in columns of text characters.

 - void [display_rows](#) (int val)

Set terminal's display height in lines of text (rows).

 - int [display_rows](#) (void) const

Return terminal's display height in lines of text (rows).

 - void [draw](#) (void) [FL_OVERRIDE](#)

Draws the entire [FL_Terminal](#).

 - void [error_char](#) (const char *val)

Sets the "error character" utf8 string shown for invalid utf8 or bad ANSI sequences if [show_unknown\(\)](#) is true.

 - const char * [error_char](#) (void) const

Returns the "error character" utf8 string, which is shown for invalid utf8 or bad ANSI sequences if [show_unknown\(\)](#) is true.

 - [FL_Terminal](#) (int X, int Y, int W, int H, const char *L, int rows, int cols, int hist)

Same as the default FLTK constructor, but lets the user force the rows, columns and history to specific sizes on creation.

 - [FL_Terminal](#) (int X, int Y, int W, int H, const char *L=0)

The constructor for [FL_Terminal](#).

 - int [handle](#) (int e) [FL_OVERRIDE](#)

Handle FLTK events.

 - void [history_lines](#) (int val)

Set the number of lines of screen history.

 - int [history_lines](#) (void) const

Return the number of lines of screen history.

 - void [history_rows](#) (int val)

Set terminal's scrollbar history buffer size in lines of text (rows).

 - int [history_rows](#) (void) const

Return terminal's scrollbar history buffer size in lines of text (rows).

 - int [history_use](#) (void) const

Returns how many lines are "in use" by the screen history buffer.

 - void [hscrollbar_style](#) ([ScrollbarStyle](#) val)

Set the horizontal scrollbar behavior style.

 - [ScrollbarStyle](#) [hscrollbar_style](#) (void) const

Get the horizontal scrollbar behavior style.

 - void [margin_bottom](#) (int val)

Set the bottom margin; see [Margins](#).

 - int [margin_bottom](#) (void) const

Return the bottom margin; see [Margins](#).

 - void [margin_left](#) (int val)

Set the left margin; see [Margins](#).

 - int [margin_left](#) (void) const

Return the left margin; see [Margins](#).

 - void [margin_right](#) (int val)

Set the right margin; see [Margins](#).

 - int [margin_right](#) (void) const

Return the right margin; see [Margins](#).

 - void [margin_top](#) (int val)

Set the top margin; see [Margins](#).

- int **margin_top** (void) const
Return the top margin; see [Margins](#).
- void **output_translate** (FL_Terminal::OutFlags val)
Sets the combined output translation flags to val.
- FL_Terminal::OutFlags **output_translate** (void) const
Return the current combined output translation flags.
- void **plot_char** (char c, int drow, int dcol)
Plot the ASCII character c at the terminal's display position (drow,dcol).
- void **plot_char** (const char *text, int len, int drow, int dcol)
Plot the UTF-8 character text of length len at display position (drow,dcol).
- void **print_char** (char c)
Prints single ASCII char c at current cursor position, and advances the cursor.
- void **print_char** (const char *text, int len=-1)
Prints single UTF-8 char text of optional byte length len at current cursor position, and advances the cursor if the character is printable.
- void **printf** (const char *fmt,...)
Appends printf formatted messages to the terminal.
- void **redraw_rate** (float val)
Set the maximum rate redraw speed in floating point seconds if [redraw_style\(\)](#) is set to RATE_LIMITED.
- float **redraw_rate** (void) const
Get max rate redraw speed in floating point seconds.
- void **redraw_style** (RedrawStyle val)
Set how FL_Terminal manages screen redrawing.
- RedrawStyle **redraw_style** (void) const
Get the redraw style.
- void **reset_terminal** (void)
Resets terminal to default colors, clears screen, history and mouse selection, homes cursor, resets tabstops.
- void **resize** (int X, int Y, int W, int H) FL_OVERRIDE
Handle widget resizing, such as if user resizes parent window.
- int **scrollbar_actual_size** (void) const
Returns the scrollbar's actual "trough size", which is the width of FL_VERTICAL scrollbars, or height of FL_↔ HORIZONTAL scrollbars.
- void **scrollbar_size** (int val)
Set the pixel size of both horizontal and vertical scrollbar's "trough" to val.
- int **scrollbar_size** (void) const
Get current pixel size of all the scrollbar's troughs for this widget, or zero if the global [FL::scrollbar_size\(\)](#) is being used (default).
- const char * **selection_text** (void) const
Return text selection (for copy()/paste() operations)
- int **selection_text_len** (void) const
Return byte length of all UTF-8 chars in selection, or 0 if no selection.
- void **selectionbgcolor** (FL_Color val)
Set mouse selection background color.
- FL_Color **selectionbgcolor** (void) const
Get mouse selection background color.
- void **selectionfgcolor** (FL_Color val)
Set mouse selection foreground color.
- FL_Color **selectionfgcolor** (void) const
Get mouse selection foreground color.
- void **show_unknown** (bool val)
Set the "show unknown" flag.

- bool [show_unknown](#) (void) const
Return the "show unknown" flag.
- const char * [text](#) (bool lines_below_cursor=false) const
Return a string copy of all lines in the terminal (including history).
- [uchar textattrib](#) () const
Get text attribute bits (underline, inverse, etc).
- void [textattrib](#) ([uchar](#) val)
Set text attribute bits (underline, inverse, etc).
- void [textbgcolor](#) ([FI_Color](#) val)
Set text background color to fltk color `val` used by any new text added.
- [FI_Color textbgcolor](#) (void) const
Return text's current background color.
- void [textbgcolor_default](#) ([FI_Color](#) val)
Set the default text background color used by any new text added after a reset (`<ESC>c`, `<ESC>[0m`, or `reset_terminal()`).
- [FI_Color textbgcolor_default](#) (void) const
Return text's default background color.
- void [textbgcolor_xterm](#) ([uchar](#) val)
Sets the background text color as one of the 8 'xterm color' values.
- void [textcolor](#) ([FI_Color](#) val)
Set the text color for the terminal.
- [FI_Color textcolor](#) (void) const
Return `textcolor()`. This is a convenience method that returns `textfgcolor_default()`
- void [textfgcolor](#) ([FI_Color](#) val)
Set text foreground drawing color to fltk color `val` used by any new text added.
- [FI_Color textfgcolor](#) (void) const
Return text's current foreground color.
- void [textfgcolor_default](#) ([FI_Color](#) val)
Set the default text foreground color used by `<ESC>c`, `<ESC>[0m`, and `reset_terminal()`.
- [FI_Color textfgcolor_default](#) (void) const
Return text's default foreground color.
- void [textfgcolor_xterm](#) ([uchar](#) val)
Sets the foreground text color as one of the 8 'xterm color' values.
- void [textfont](#) ([FI_Font](#) val)
Sets the font used for all text displayed in the terminal.
- [FI_Font textfont](#) (void) const
Return text font used to draw all text in the terminal.
- void [textsize](#) ([FI_Fonsize](#) val)
Sets the font size used for all text displayed in the terminal.
- [FI_Fonsize textsize](#) (void) const
Return text font size used to draw all text in the terminal.
- void [vprintf](#) (const char *fmt, va_list ap)
Appends printf formatted messages to the terminal.
- [~FI_Terminal](#) (void)
The destructor for `FI_Terminal`.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *[_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void [add](#) ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void [add](#) ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void [add_resizable](#) ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * [array](#) () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * [as_group](#) () const [FL_OVERRIDE](#)
- [FL_Group](#) * [as_group](#) () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void [begin](#) ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * [child](#) (int n) const
Returns the n'th child.
- int [children](#) () const
Returns how many child widgets the group has.
- void [clear](#) ()
Deletes all child widgets from memory recursively.
- unsigned int [clip_children](#) ()
Returns the current clipping mode.
- void [clip_children](#) (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int [delete_child](#) (int n)
*Removes the widget at *index* from the group and deletes it.*
- void [end](#) ()
*Exactly the same as *current(this->parent())*.*
- int [find](#) (const [FL_Widget](#) &o) const
*See int [FL_Group::find\(const FL_Widget *w\)](#) const.*
- int [find](#) (const [FL_Widget](#) *) const
Searches the children for the widget and returns the index.
- [FL_Group](#) (int, int, int, int, const char *==0)
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- void [focus](#) ([FL_Widget](#) *W)
- void [forms_end](#) ()
This is for forms compatibility only.
- void [init_sizes](#) ()
Resets the internal array of widget sizes and positions.
- void [insert](#) ([FL_Widget](#) &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void [insert](#) ([FL_Widget](#) &o, [FL_Widget](#) *before)
*This does *insert(w, find(before))*.*
- void [remove](#) ([FL_Widget](#) &)
Removes a widget from the group but does not delete it.
- void [remove](#) ([FL_Widget](#) *o)
Removes the widget o from the group.
- void [remove](#) (int index)

- Removes the widget at `index` from the group but does not delete it.*

 - `FL_Widget * resizable () const`
 - Returns the group's resizable widget.*
 - `void resizable (FL_Widget &o)`
 - Sets the group's resizable widget.*
 - `void resizable (FL_Widget *o)`
 - The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
 - `virtual ~FL_Group ()`
 - The destructor also deletes all the children.*

Public Member Functions inherited from FL_Widget

- `void _clear_fullscreen ()`
- `void _set_fullscreen ()`
- `void activate ()`
 - Activates the widget.*
- `unsigned int active () const`
 - Returns whether the widget is active.*
- `int active_r () const`
 - Returns whether the widget and all of its parents are active.*
- `FL_Align align () const`
 - Gets the label alignment.*
- `void align (FL_Align alignment)`
 - Sets the label alignment.*
- `long argument () const`
 - Gets the current user data (long) argument that is passed to the callback function.*
- `void argument (long v)`
 - Sets the current user data (long) argument that is passed to the callback function.*
- `virtual class FL_Gl_Window * as_gl_window ()`
 - Returns an `FL_Gl_Window` pointer if this widget is an `FL_Gl_Window`.*
- `virtual class FL_Gl_Window const * as_gl_window () const`
- `virtual FL_Window * as_window ()`
 - Returns an `FL_Window` pointer if this widget is an `FL_Window`.*
- `virtual FL_Window const * as_window () const`
- `void bind_deimage (FL_Image *img)`
 - Sets the image to use as part of the widget label when in the inactive state.*
- `void bind_deimage (int f)`
 - Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- `void bind_image (FL_Image *img)`
 - Sets the image to use as part of the widget label when in the active state.*
- `void bind_image (int f)`
 - Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- `FL_Boxtype box () const`
 - Gets the box type of the widget.*
- `void box (FL_Boxtype new_box)`
 - Sets the box type for the widget.*
- `FL_Callback_p callback () const`
 - Gets the current callback function for the widget.*
- `void callback (FL_Callback *cb)`
 - Sets the current callback function for the widget.*
- `void callback (FL_Callback *cb, FL_Callback_User_Data *p, bool auto_free)`

- Sets the current callback function and managed user data for the widget.*

 - void [callback](#) ([FI_Callback](#) *cb, void *p)

Sets the current callback function and data for the widget.

 - void [callback](#) ([FI_Callback0](#) *cb)

Sets the current callback function for the widget.

 - void [callback](#) ([FI_Callback1](#) *cb, long p=0)

Sets the current callback function for the widget.

 - unsigned int [changed](#) () const

Checks if the widget value changed since the last callback.

 - void [clear_active](#) ()

Marks the widget as inactive without sending events or changing focus.

 - void [clear_changed](#) ()

Marks the value of the widget as unchanged.

 - void [clear_damage](#) (uchar c=0)

Clears or sets the damage flags.

 - void [clear_output](#) ()

Sets a widget to accept input.

 - void [clear_visible](#) ()

Hides the widget.

 - void [clear_visible_focus](#) ()

Disables keyboard focus navigation with this widget.

 - [FI_Color](#) [color](#) () const

Gets the background color of the widget.

 - void [color](#) ([FI_Color](#) bg)

Sets the background color of the widget.

 - void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)

Sets the background and selection color of the widget.

 - [FI_Color](#) [color2](#) () const

For back compatibility only.

 - void [color2](#) (unsigned a)

For back compatibility only.

 - int [contains](#) (const [FI_Widget](#) *w) const

Checks if w is a child of this widget.

 - void [copy_label](#) (const char *new_label)

Sets the current label.

 - void [copy_tooltip](#) (const char *text)

Sets the current tooltip text.

 - [uchar](#) [damage](#) () const

Returns non-zero if [draw\(\)](#) needs to be called.

 - void [damage](#) ([uchar](#) c)

Sets the damage bits for the widget.

 - void [damage](#) ([uchar](#) c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.

 - int [damage_resize](#) (int, int, int, int)

Internal use only.

 - void [deactivate](#) ()

Deactivates the widget.

 - [FI_Image](#) * [deimage](#) ()

Gets the image that is used as part of the widget label when in the inactive state.

 - const [FI_Image](#) * [deimage](#) () const

Gets the image that is used as part of the widget label when in the inactive state.

- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const

- Gets the font to use.*

 - void [labelfont](#) ([FI_Font](#) f)

Sets the font to use.
- [FI_Fontsize](#) [labelsize](#) () const

Gets the font size in pixels.
- void [labelsize](#) ([FI_Fontsize](#) pix)

Sets the font size in pixels.
- [FI_Labeltype](#) [labeltype](#) () const

Gets the label type.
- void [labeltype](#) ([FI_Labeltype](#) a)

Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FI_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FI_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- [FI_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.

- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- uchar [type](#) () const
Gets the widget type.
- void [type](#) (uchar t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) (uchar i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual ~[Fl_Widget](#) ()
Destroys the widget.

Public Attributes

- [Fl_Scrollbar](#) * [hscrollbar](#)
Horizontal scrollbar.
- [Fl_Scrollbar](#) * [scrollbar](#)
Vertical scrollbar.

Protected Member Functions

- void **clear_eod** (void)
Clear from cursor to End Of Display (EOD), like "<ESC> [J<ESC> [0J".
- void **clear_eol** (void)
Clear from cursor to End Of Line (EOL), like "<ESC> [K".
- void **clear_line** (int row)
Clear entire line for specified row.
- void **clear_line** (void)
Clear entire line cursor is currently on.
- void **clear_mouse_selection** (void)
Clear any current mouse selection.
- void **clear_sod** (void)
Clear from cursor to Start Of Display (EOD), like "<ESC> [1J".
- void **clear_sol** (void)
Clear from cursor to Start Of Line (SOL), like "<ESC> [1K".
- void **current_style** (const [CharStyle](#) &sty)
Set current style for rendering text.
- [CharStyle](#) & **current_style** (void) const
Return reference to internal current style for rendering text.
- void **cursor_col** (int col)
*Move cursor to the specified column *col*.*
- void **cursor_cr** (void)
Move cursor as if a CR (\r) was received.
- void **cursor_crlf** (int count=1)
Move cursor as if a CR/LF pair (\r\n) was received.
- void **cursor_down** (int count=1, bool do_scroll=false)
*Moves cursor down *count* lines.*
- void **cursor_eol** (void)
Move cursor to the last column (at the far right) on the current line.
- void **cursor_left** (int count=1)
*Moves cursor left *count* columns, and cursor stops (does not wrap) if it hits screen edge.*
- void **cursor_right** (int count=1, bool do_scroll=false)
*Moves cursor right *count* columns.*
- void **cursor_row** (int row)
*Move cursor to the specified row *row*.*
- void **cursor_sol** (void)
Move cursor to the first column (at the far left) on the current line.
- void **cursor_tab_left** (int count=1)
Tab left, do not wrap beyond left edge.
- void **cursor_tab_right** (int count=1)
Tab right, do not wrap beyond right edge.
- void **cursor_up** (int count=1, bool do_scroll=false)
*Moves cursor up *count* lines.*
- void **delete_chars** (int drow, int dcol, int rep)

- Delete char(s) at (drow,dcol) for 'rep' times.*

 - void **delete_chars** (int rep)
- Delete char(s) at cursor position for 'rep' times.*

 - void **delete_rows** (int count)
- Delete (count) rows at cursor position.*

 - int **disp_cols** (void) const
- Return the number of columns in the display area.*

 - int **disp_erow** (void) const
- Return the ending row# in the display area.*

 - int **disp_rows** (void) const
- Return the number of rows in the display area.*

 - int **disp_srow** (void) const
- Return the starting row# in the display area.*

 - void **draw_buff** (int Y) const
- Draws the buffer position we are scrolled to onto the FLTK screen starting at pixel position Y.*

 - void **draw_row** (int grow, int Y) const
- Draw the specified global row, which is the row in ring_chars[].*

 - void **draw_row_bg** (int grow, int X, int Y) const
- Draw the background for the specified ring_chars[] global row grow starting at FLTK coords X and Y.*

 - bool **get_selection** (int &srow, int &scol, int &erow, int &ecol) const
- Return mouse selection's start/end position in the ring buffer, if any.*

 - int **h_to_row** (int H) const
- Given a height in pixels, return number of rows that "fits" into that area.*

 - int **handle_unknown_char** (int drow, int dcol)
- Handle an unknown char by either emitting an error symbol to the tty, or do nothing, depending on the user configurable value of show_unknown().*

 - int **handle_unknown_char** (void)
- Handle an unknown char by either emitting an error symbol to the tty, or do nothing, depending on the user configurable value of show_unknown().*

 - int **hist_cols** (void) const
- Return the number of columns in the scrollbar history.*

 - int **hist_erow** (void) const
- Return the ending row# of the scrollbar history.*

 - int **hist_rows** (void) const
- Return the number of rows in the scrollbar history.*

 - int **hist_srow** (void) const
- Return the starting row# of the scrollbar history.*

 - int **hist_use** (void) const
- Return number of rows in use by the scrollbar history.*

 - int **hist_use_srow** (void) const
- Return the starting row of the "in use" scrollbar history.*

 - void **insert_char** (char c, int rep)
- Insert char 'c' at the current cursor position for 'rep' times.*

 - void **insert_char_eol** (char c, int drow, int dcol, int rep)
- Insert char 'c' for 'rep' times at display row 'drow' and column 'dcol'.*

 - void **insert_rows** (int count)
- Insert (count) rows at current cursor position.*

 - bool **is_inside_selection** (int row, int col) const
- Is global row/column inside the current mouse selection?*

 - bool **is_selection** (void) const
- Returns true if there's a mouse selection.*

- int **offset** (void) const
Returns the current offset into the ring buffer.
- void **restore_cursor** (void)
Restore previously saved cursor position, if any. Used by ESC [u.
- int **ring_cols** (void) const
Return the number of columns in the ring buffer.
- int **ring_erow** (void) const
Return the ending row# in the ring buffer (Always [ring_rows\(\)](#)-1)
- int **ring_rows** (void) const
Return the number of rows in the ring buffer.
- int **ring_srow** (void) const
Return the starting row# in the ring buffer. (Always 0)
- void **save_cursor** (void)
Save current cursor position. Used by ESC [s.
- void **scroll** (int rows)
Scroll the display up(+) or down(-) the specified rows.
- void **select_line** (int grow)
Select the entire row.
- void **select_word** (int grow, int gcol)
Select the word around the given row and column.
- bool **selection_extend** (int X, int Y)
Extend selection to FLTK coords X,Y.
- [Utf8Char](#) * **u8c_cursor** (void)
Return the [Utf8Char](#) for character under cursor.*
- [Utf8Char](#) * **u8c_disp_row** (int drow)
Return pointer to the first [u8c](#) character in row drow of the display.
- const [Utf8Char](#) * **u8c_disp_row** (int drow) const
See docs for non-const version of [u8c_disp_row\(int\)](#)
- [Utf8Char](#) * **u8c_hist_row** (int hrow)
Return [u8c](#) for beginning of a row inside the scrollbar history.
- const [Utf8Char](#) * **u8c_hist_row** (int hrow) const
See docs for non-const version of [u8c_hist_row\(int\)](#)
- const [Utf8Char](#) * **u8c_hist_use_row** (int hrow) const
See docs for non-const version of [u8c_hist_use_row\(int\)](#)
- [Utf8Char](#) * **u8c_hist_use_row** (int hurow)
Return [u8c](#) for beginning of row hurow inside the 'in use' part of the scrollbar history.
- [Utf8Char](#) * **u8c_ring_row** (int grow)
Return UTF-8 char for row grow in the ring buffer.
- const [Utf8Char](#) * **u8c_ring_row** (int grow) const
See docs for non-const version of [u8c_ring_row\(int\)](#)
- int **w_to_col** (int W) const
Given a width in pixels, return number of columns that "fits" into that area.
- const [Utf8Char](#) * **walk_selection** (const [Utf8Char](#) *u8c, int &row, int &col) const
Walk the mouse selection one character at a time from beginning to end, returning a [Utf8Char](#) to the next character in the selection, or NULL if the end was reached, or if there's no selection.*

Protected Member Functions inherited from FL_Group

- `FL_Rect * bounds ()`
Returns the internal array of widget sizes and positions.
- `void draw_child (FL_Widget &widget) const`
Forces a child to redraw.
- `void draw_children ()`
Draws all children of the group.
- `void draw_outside_label (const FL_Widget &widget) const`
Parents normally call this to draw outside labels of child widgets.
- `virtual int on_insert (FL_Widget *, int)`
Allow derived groups to act when a widget is added as a child.
- `virtual int on_move (int, int)`
Allow derived groups to act when a widget is moved within the group.
- `virtual void on_remove (int)`
Allow derived groups to act when a child widget is removed from the group.
- `int * sizes ()`
Returns the internal array of widget sizes and positions.
- `void update_child (FL_Widget &widget) const`
Draws a child only if it needs it.

Protected Member Functions inherited from FL_Widget

- `void clear_flag (unsigned int c)`
Clears a flag in the flags mask.
- `void draw_backdrop () const`
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- `void draw_box () const`
Draws the widget box according its box style.
- `void draw_box (FL_Boxtype t, FL_Color c) const`
Draws a box of type t, of color c at the widget's position and size.
- `void draw_box (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const`
Draws a box of type t, of color c at the position X,Y and size W,H.
- `void draw_focus () const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int X, int Y, int W, int H) const`
Draws a focus rectangle around the widget.
- `void draw_focus (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const`
Draws a focus box for the widget at the given position and size.
- `void draw_label () const`
Draws the widget's label at the defined label position.
- `void draw_label (int, int, int, int) const`
Draws the label in an arbitrary bounding box.
- `FL_Widget (int x, int y, int w, int h, const char *label=0L)`
Creates a widget at the given position and size.
- `unsigned int flags () const`
Gets the widget flags mask.
- `void h (int v)`
Internal use only.
- `void set_flag (unsigned int c)`
Sets a flag in the flags mask.
- `void w (int v)`

Internal use only.

- void `x` (int v)

Internal use only.

- void `y` (int v)

Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from `Fl_Group`

- static `Fl_Group * current` ()
Returns the currently active group.
- static void `current` (`Fl_Group *g`)
Sets the current group.

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget *widget`, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from `Fl_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
 ,
`OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
 ,
`MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
 ,
`GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
 = 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

11.142.1 Detailed Description

Terminal widget supporting Unicode/utf-8, ANSI/xterm escape codes with full RGB color control.

11.142.2 FI_Terminal

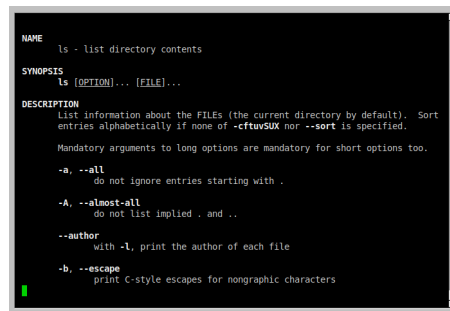


Figure 11.54 FI_Terminal widget showing a linux manual page

[FI_Terminal](#) is an output-only text widget supporting ASCII and UTF-8/Unicode. It supports most terminal text features, such as most VT100/xterm style escape sequences (see [The Escape Codes FI_Terminal Supports](#)), text colors/attributes, scrollback history, mouse selection, etc.

It is recommended that accessing features such as setting text colors and cursor positioning is best done with ANSI/XTERM escape sequences. But if one sets `ansi(false)` then this is not possible, so the public API can be used for common operations, e.g.

Public API	ESC code equivalent	Description
clear_screen_home()	ESC [H ESC [2 J	Clear screen, home cursor
cursor_home()	ESC [H	Home the cursor
clear_history()	ESC [3 J	Clear scrollback history
reset_terminal()	ESC [c	Reset terminal

To access more advanced API calls, one can derive a class from [FI_Terminal](#) to access protected methods manipulate the terminal more directly, e.g.

Protected API	ESC code equiv.	Description
current_style()	ESC [# m	Set text attributes
clear_eod()	ESC [0 J	Clear from cursor to end of display
clear_sod()	ESC [1 J	Clear from cursor to start of display
clear_eol()	ESC [0 K	Clear from cursor to end of line
clear_sol()	ESC [1 K	Clear from cursor to start of line
clear_line()	ESC [2 K	Clear line cursor is on
scroll(int) // >0 for up	ESC [1 S	Scroll up one line
scroll(int) // <0 for down	ESC [1 T	Scroll down one line
cursor_left()	ESC [1 D	Move cursor left (no wrap)
cursor_right()	ESC [1 C	Move cursor right (no wrap)
cursor_up()	ESC [1 B	Move cursor up (no scroll or wrap)
cursor_down()	ESC [1 A	Move cursor down (no scroll or wrap)
cursor_row() cursor_col()	ESC [# ; # H	Move cursor to row# / column#
insert_char()	ESC [# @	Insert a char at cursor position
delete_chars()	ESC [# P	Delete chars at cursor position
insert_rows()	ESC [# L	Insert rows at cursor position
delete_rows()	ESC [# M	Delete rows at cursor position
etc...	etc...	etc...

Many commonly used API functions are public, such as `textfgcolor()` for setting text colors. Others, such as `cursor_up()` are protected to prevent common misuse, and are available only to subclasses.

Some common operations:

- Set the terminal's background color, see `color(Fl_Color)`
- Set the terminal's default text color, see `textfgcolor_default(Fl_Color)`
- Printing text to the terminal, see `Fl_Terminal::printf()` and `Fl_Terminal::append()`
- Clearing the screen, see `clear()`
- Getting the terminal's buffer contents, see `text()`
- Getting single utf8 characters by row/col from the terminal display, see `utf8_char_at_disp()`
- Getting the text from a text selection, see `get_selection()`

For applications that need input support, the widget can be subclassed to provide keyboard input, and advanced features like pseudo ttys, termio, serial port I/O, etc., as such features are beyond the scope of FLTK.

11.142.2.1 Examples

```
//
// Fl_Terminal: Simple Use
//
Fl_Terminal *tty = new Fl_Terminal(...);
:
tty->append("Hello world.\n"); // simple strings
tty->append("\033[31mThis text is red.\033[0m\n"); // colored text
tty->append("\033[32mThis text is green.\033[0m\n");
tty->printf("The value of x is %.02f\n", x); // printf() formatting
```

There are also public methods for doing what most "\033[" escape codes do, so that if `ansi(bool)` is set to "false", one can still change text colors or clear the screen via application control, e.g.

```
tty->home(); // home the cursor
tty->clear_screen(); // clear the screen
tty->textfgcolor(0xff000000); // change the text color to RED
tty->textbgcolor(0x0000ff00); // change the background color to BLUE
//
// R G B
```

When creating the widget, the width/height determine the default column and row count for the terminal's display based on the current font size. The column width determines where text will wrap.

You can specify wider column sizes than the screen using `display_columns(colwidth)`. When this value is larger than the widget's width, text will wrap off-screen, and can be revealed by resizing the widget wider.

11.142.2.2 Writing To Terminal From Applications

An application needing terminal output as part of its user interface can instance `Fl_Terminal`, and write text strings with:

- `append()` to append strings
- `printf()` to append formatted strings

Single character output can be done with:

- `print_char()` to print a single ASCII/UTF-8 char at the cursor
- `plot_char()` to put single ASCII/UTF-8 char at an x,y position

11.142.2.3 Text Attributes

The terminal's text supports these attributes:

- Italic - italicized text: `\033[3m`
- Bold - brighter/thicker text: `\033[1m`
- Dim - lower brightness text: `\033[2m`
- Underline - text that is underlined: `\033[4m`
- Strikeout - text that has a line through the text: `\033[9m`
- Inverse - text whose background and foreground colors are swapped: `\033[7m`
- Normal - normal text: `\033[0m`

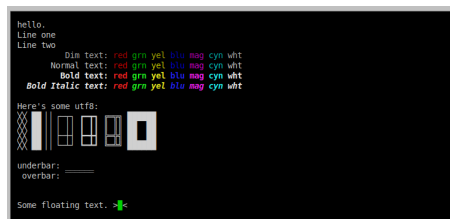


Figure 11.55 FI_Terminal screen

11.142.2.4 Text and Background Colors

There's at least two ways to specify colors for text and background colors:

- 3 bit / 8 Color Values
- Full 24 bit R/G/B colors

Example of 3 bit colors:

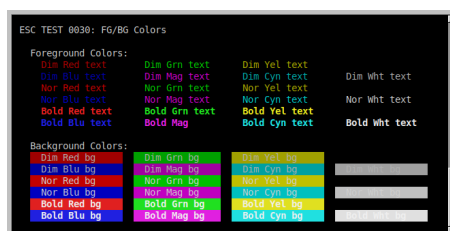


Figure 11.56 FI_Terminal 3 bit colors

Example application source code using 3 bit colors:

```
//
// Text colors
//
tty->append("\033[31m Red text.\033[0m\n"); // Print red text..
tty->append("\033[32m Green text.\033[0m\n");
:
tty->append("\033[36m Cyan text.\033[0m\n");
tty->append("\033[37m White text.\033[0m\n");
//
// Background colors
//
tty->append("\033[41m Red Background.\033[0m\n"); // background will be red
tty->append("\033[42m Green Background.\033[0m\n");
:
tty->append("\033[46m Cyan Background.\033[0m\n");
tty->append("\033[47m White Background.\033[0m\n");
```

Example of 24 bit colors:

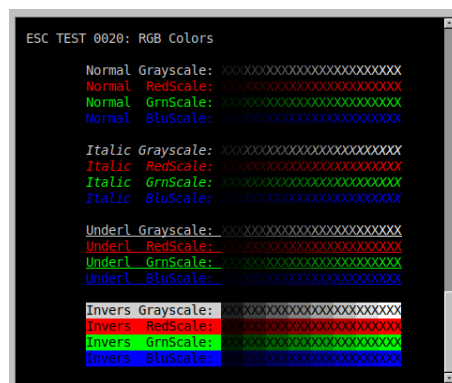


Figure 11.57 FI_Terminal 24 bit colors

Example application source code using 24 bit colors:

```
//
// 24 bit Text Color
//
tty->append("\033[38;2;0;0;255m Text is BLUE.\033[0m\n"); // RGB: R=0, G=0, B=255
tty->append("\033[38;2;255;0;0m Text is RED.\033[0m\n"); // RGB: R=255, G=0, B=0
tty->append("\033[38;2;127;64;0m Text is DARK ORANGE.\033[0m\n"); // RGB: R=127, G=64, B=0
//
// 24 bit Background Color
//
tty->append("\033[48;2;0;0;255m Background is BLUE.\033[0m\n"); // RGB: R=0, G=0, B=255
tty->append("\033[48;2;255;0;0m Background is RED.\033[0m\n"); // RGB: R=255, G=0, B=0
tty->append("\033[48;2;127;64;0m Background is DARK ORANGE.\033[0m\n"); // RGB: R=127, G=64, B=0
```

For more on the ANSI escape codes, see [The Escape Codes FI_Terminal Supports](#).

11.142.2.5 Features

Most standard terminal behaviors are supported, e.g.

- ASCII + UTF-8/Unicode
- scrollback history management
- mouse selection + copy/paste (^C, ^A)
- autoscroll during selection

Most popular ANSI/DEC VT100/Xterm escape sequences are supported (see [The Escape Codes Fl_Terminal Supports](#)), including:

- per-character colors for text and background
- per-character text attributes: bold/dim, underline, strikeouts
- scrolling up/down
- character insert/delete for characters/rows/screen
- clearing characters/rows/screen

Does not (yet) support:

- programmable regions (scroll regions and attribute blocks)
- dynamic line wrap (where resizing display dynamically re-wraps long lines)

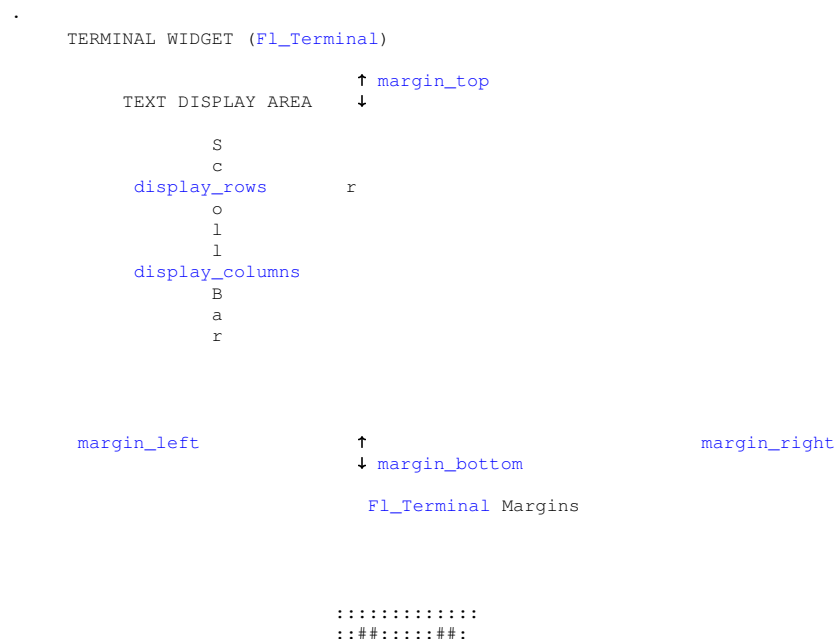
Will likely never implement as part of this widget:

- pty/termio management (such features should be *subclassed*)
- Different per-character font family + sizes (font family/size is global only)
- variable width fonts

Regarding the font family+size; the way the terminal is currently designed, the font family and size must not vary within text; rows have to be consistent height. Varying widths are tricky too, esp. when it comes to moving the cursor up/down within a column; varying *widths* are supported (due to Unicode characters sometimes being "wide", but not heights).

11.142.2.6 Margins

The margins define the amount of space (in pixels) around the outside of the text display area, the space between the widget's inner edge (inside the [box\(\)](#)) and the text display area's outer edge. The margins can be inspected and changed with the [margin_left\(\)](#), [margin_right\(\)](#), [margin_top\(\)](#) and [margin_bottom\(\)](#) methods.



```

::###::###:
::#####: Fl_Terminal::color()
::#####:
::#####:
::###::###:
::###::###:
::#####:
::#####:
::#####:
Fl_Terminal::textbgcolor()

Fl_Terminal::textfgcolor()

```

11.142.2.7 Caveats

- This widget is not a full terminal emulator; it does not do stdio redirection, pseudo ttys/termios/character cooking, keyboard input processing, full curses(3) support. However, such features CAN be implemented with subclassing.
- The `printf()` and `vprintf()` functions are limited to strings no longer than 1024 characters (including NULL). For printing longer strings, use `append()` which has no string limits.
- For background colors `textbgcolor()` and `textbgcolor_default()`, the special color value 0xffffffff lets the widget's `box()` `color()` show through behind the text.

11.142.3 Member Enumeration Documentation

11.142.3.1 Attrib

enum `Fl_Terminal::Attrib`

Bits for the per-character attributes, which control text features such as italic, bold, underlined text, etc.
NOTE: enum names with a leading underbar must not be used, and are reserved for future use.

Enumerator

NORMAL	all attributes off
BOLD	bold text: uses bold font, color brighter than normal
DIM	dim text; color slightly darker than normal
ITALIC	italic font text
UNDERLINE	underlined text
<code>_RESERVED↵ _1</code>	<i>(reserved for internal future use)</i>
INVERSE	inverse text; fg/bg color are swapped
<code>_RESERVED↵ _2</code>	<i>(reserved for internal future use)</i>
STRIKEOUT	strikeout text

11.142.3.2 CharFlags

enum `Fl_Terminal::CharFlags`

Per-character 8 bit flags (uchar) used to manage special states for characters.

Enumerator

FG_XTERM	this char's fg color is an XTERM color; can be affected by Dim+Bold
BG_XTERM	this char's bg color is an XTERM color; can be affected by Dim+Bold
EOL	TODO: char at EOL, used for line re-wrap during screen resizing.

11.142.3.3 OutFlags

enum [Fl_Terminal::OutFlags](#)

Output translation flags for special control character translations.

Enumerator

OFF	no output translation
CR_TO_LF	carriage return generates a vertical line-feed (<code>\r</code> -> <code>\n</code>)
LF_TO_CR	line-feed generates a carriage return (<code>\n</code> -> <code>\r</code>)
LF_TO_CRLF	line-feed generates a carriage return line-feed (<code>\n</code> -> <code>\r\n</code>)

11.142.3.4 RedrawStyle

enum [Fl_Terminal::RedrawStyle](#)

Determines when [Fl_Terminal](#) calls [redraw\(\)](#) if new text is added.

RATE_LIMITED is the recommended setting, using [redraw_rate\(float\)](#) to determine the maximum rate of redraws.

See also

[redraw_style\(\)](#), [redraw_rate\(\)](#)

Enumerator

NO_REDRAW	app must call redraw() as needed to update text to screen
RATE_LIMITED	timer controlled redraws. (DEFAULT)
PER_WRITE	redraw triggered after <i>every</i> append() / printf() / etc. operation

11.142.3.5 ScrollbarStyle

enum [Fl_Terminal::ScrollbarStyle](#)

Behavior of scrollbars.

Enumerator

SCROLLBAR_OFF	scrollbar always invisible
SCROLLBAR_AUTO	scrollbar visible if widget resized in a way that hides columns (default)
SCROLLBAR_ON	scrollbar always visible

11.142.4 Constructor & Destructor Documentation

11.142.4.1 Fl_Terminal() [1/2]

```
Fl_Terminal::Fl_Terminal (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

The constructor for [Fl_Terminal](#).

This creates an empty terminal with defaults:

- white on black text; see [textfgcolor\(Fl_Color\)](#), [textbgcolor\(Fl_Color\)](#)
- rows/cols based on the W and H values, see [display_rows\(\)](#), [display_columns\(\)](#)

- scrollbar history of 100 lines, see [history_rows\(\)](#)
- [redraw_style\(\)](#) set to RATE_LIMITED, [redraw_rate\(\)](#) set to 0.10 seconds

Note: While [Fl_Terminal](#) derives from [Fl_Group](#), it's not intended for user code to use it as a parent for other widgets, so [end\(\)](#) is called.

Parameters

in	<i>X,Y,W,H</i>	position and size.
in	<i>L</i>	label string (optional), may be NULL.

11.142.4.2 Fl_Terminal() [2/2]

```
Fl_Terminal::Fl_Terminal (
    int X,
    int Y,
    int W,
    int H,
    const char * L,
    int rows,
    int cols,
    int hist)
```

Same as the default FLTK constructor, but lets the user force the rows, columns and history to specific sizes on creation.

Since the row/cols/hist are specified directly, this prevents the widget from auto-calculating the initial text buffer size based on the widget's pixel width/height, bypassing calls to the font system before the widget is displayed.

Note

fluid uses this constructor internally to avoid font calculations that opens the display, useful for when running in a headless context. (issue 837)

11.142.4.3 ~Fl_Terminal()

```
Fl_Terminal::~~Fl_Terminal (
    void )
```

The destructor for [Fl_Terminal](#).

Destroys the terminal display, scroll history, and associated widgets.

11.142.5 Member Function Documentation

11.142.5.1 ansi() [1/2]

```
void Fl_Terminal::ansi (
    bool val)
```

Enable/disable the ANSI mode flag.

If true, ANSI and VT100/xterm codes will be processed. If false, these codes won't be processed and will either be ignored or print the error character "¿", depending on the value of [show_unknown\(\)](#).

See also

[show_unknown\(\)](#), [The Escape Codes Fl_Terminal Supports](#)

11.142.5.2 ansi() [2/2]

```
bool Fl_Terminal::ansi (
    void ) const
```

Return the state of the ANSI flag.

See also

[ansi\(bool\)](#)

11.142.5.3 append()

```
void Fl_Terminal::append (
    const char * s,
    int len = -1)
```

Appends string *s* to the terminal at the current cursor position using the current text color/attributes.

If *s* is NULL, the UTF-8 character cache is cleared, which is recommended before starting a block reading loop, and again after the block loop has completed.

If *len* is not specified, it's assumed *s* is a NULL terminated string. If *len* IS specified, it can be used for writing strings that aren't NULL terminated, such as block reads on a pipe, network, or other block oriented data source.

Redraws of the terminal widget are by default handled automatically, but can be changed with [redraw_rate\(\)](#) and [redraw_style\(\)](#).

Block I/O

When reading block oriented sources (such as pipes), [append\(\)](#) will handle partial UTF-8 chars straddling the block boundaries. It does this using an internal byte cache, which should be cleared before and after block I/O loops by calling `append(NULL)` as shown in the example below, to prevent the possibilities of partial UTF-8 characters left behind by an interrupted or incomplete block loop.

```
// Example block reading a command pipe in Unix

// Run command and read as a pipe
FILE *fp = popen("ls -la", "r");
if (!fp) { ..error_handling.. }

// Enable non-blocking I/O
int fd = fileno(fp);
fcntl(fd, F_SETFL, O_NONBLOCK);

// Clear UTF-8 character cache before starting block loop
G_tty->append(NULL); // prevents leftover partial UTF-8 bytes

// Block read loop
while (1) {
    Fl::wait(0.05); // give fltk .05 secs of cpu to manage UI
    ssize_t bytes = read(fd, s, sizeof(s)); // read block from pipe
    if (bytes == -1 && errno == EAGAIN) continue; // no data yet? continue
    if (bytes > 0) G_tty->append(s); // append output to terminal
    else break; // end of pipe?
}

// Flush cache again after block loop completes
G_tty->append(NULL);

// Close pipe, done
pclose(fp);
```

Note

- String can contain ASCII or UTF-8 chars
- *len* is optional; if unspecified, expects *s* to be a NULL terminated string
- Handles partial UTF-8 chars split between calls (e.g. block oriented writes)
- If *s* is NULL, this clears the "partial UTF-8" character cache
- Redraws are managed automatically by default; see [redraw_style\(\)](#)

11.142.5.4 append_ascii()

```
void Fl_Terminal::append_ascii (
    const char * s)
```

Append NULL terminated ASCII string to terminal, slightly more efficient than [append_utf8\(\)](#).

- If *s* is NULL, behavior is to do nothing
- Redraws are triggered automatically, depending on [redraw_style\(\)](#)

11.142.5.5 append_utf8()

```
void Fl_Terminal::append_utf8 (
    const char * buf,
    int len = -1)
```

Append NULL terminated UTF-8 string to terminal.

- If buf is NULL, UTF-8 cache buffer is cleared
- If optional len isn't specified or is -1, strlen(text) is used.
- If len is 0 or <-1, no changes are made
- Handles UTF-8 chars split across calls (e.g. block writes from pipes, etc)
- Redraws are triggered automatically, depending on [redraw_style\(\)](#)

11.142.5.6 box()

```
void Fl_Terminal::box (
    Fl_Boxtype val) [inline]
```

Sets the box type, updates terminal margins et al.

Default is FL_DOWN_FRAME.

FL_XXX_FRAME types are handled in a special way by this widget, and guarantee the background is a flat field.

FL_XXX_BOX may draw gradients as inherited by [Fl::scheme\(\)](#).

11.142.5.7 clear() [1/2]

```
void Fl_Terminal::clear (
    Fl_Color val)
```

Clears the screen to a specific color val and homes the cursor.

See also

[clear_screen\(\)](#), [clear_screen_home\(\)](#), [cursor_home\(\)](#)

11.142.5.8 clear() [2/2]

```
void Fl_Terminal::clear (
    void )
```

Clears the screen to the current [textbgcolor\(\)](#), and homes the cursor.

See also

[clear_screen\(\)](#), [clear_screen_home\(\)](#), [cursor_home\(\)](#)

11.142.5.9 clear_screen()

```
void Fl_Terminal::clear_screen (
    bool scroll_to_hist = true)
```

Clear the terminal screen only; does not affect the cursor position.

Also clears the current mouse selection.

If 'scroll_to_hist' is true, the screen is cleared by scrolling the contents into the scrollbar history, where it can be retrieved with the scrollbar. This is the default behavior. If false, the screen is cleared and the scrollbar history is unchanged.

Similar to the escape sequence "<ESC>[2J".

See also

[clear_screen_home\(\)](#)

11.142.5.10 clear_screen_home()

```
void Fl_Terminal::clear_screen_home (
    bool scroll_to_hist = true)
```

Clear the terminal screen and home the cursor.

Also clears the current mouse selection.

If 'scroll_to_hist' is true, the screen is cleared by scrolling the contents into the scrollback history, where it can be retrieved with the scrollbar. This is the default behavior. If false, the screen is cleared and the scrollback history is unchanged.

Similar to the escape sequence "<ESC>[2J<ESC>[H".

See also

[clear_screen\(\)](#)

11.142.5.11 color()

```
void Fl_Terminal::color (
    Fl_Color val)
```

Sets the background color for the terminal's [Fl_Group::box\(\)](#).

If the [textbgcolor\(\)](#) and [textbgcolor_default\(\)](#) are set to the special "see through" color 0xffffffff when any text was added, changing [color\(\)](#) affects the color that shows through behind that existing text.

Otherwise, whatever specific background color was set for existing text will persist after changing [color\(\)](#).

To see the effects of a change to [color\(\)](#), follow up with a call to [redraw\(\)](#).

The default value is 0x0.

11.142.5.12 cursor_col()

```
void Fl_Terminal::cursor_col (
    int col) [protected]
```

Move cursor to the specified column *col*.

This value is clamped to the range (0..[display_columns\(\)](#)-1).

11.142.5.13 cursor_cr()

```
void Fl_Terminal::cursor_cr (
    void ) [protected]
```

Move cursor as if a CR (\r) was received.

Same as [cursor_sol\(\)](#)

11.142.5.14 cursor_down()

```
void Fl_Terminal::cursor_down (
    int count = 1,
    bool do_scroll = false) [protected]
```

Moves cursor down *count* lines.

If cursor hits screen bottom, it either stops (does not wrap) if *do_scroll* is false, or wraps and scrolls up if *do_scroll* is true.

Parameters

<i>count</i>	Number of lines to move cursor down
<i>do_scroll</i>	Enable scrolling if set to true

11.142.5.15 cursor_right()

```
void Fl_Terminal::cursor_right (
    int count = 1,
    bool do_scroll = false) [protected]
```

Moves cursor right `count` columns.

If cursor hits right edge of screen, it either stops (does not wrap) if `do_scroll` is false, or wraps and scrolls up one line if `do_scroll` is true.

11.142.5.16 `cursor_row()`

```
void Fl_Terminal::cursor_row (
    int row) [protected]
```

Move cursor to the specified row `row`.

This value is clamped to the range (0..`display_rows()`-1).

11.142.5.17 `cursor_up()`

```
void Fl_Terminal::cursor_up (
    int count = 1,
    bool do_scroll = false) [protected]
```

Moves cursor up `count` lines.

If cursor hits screen top, it either stops (does not wrap) if `do_scroll` is false, or scrolls down if `do_scroll` is true.

11.142.5.18 `delete_rows()`

```
void Fl_Terminal::delete_rows (
    int count) [protected]
```

Delete (count) rows at cursor position.

Causes rows to scroll up, and empty lines created at bottom of screen. Lines deleted by scroll up are NOT moved into the scroll history.

11.142.5.19 `display_columns()` [1/2]

```
void Fl_Terminal::display_columns (
    int dcols)
```

Set terminal's display width in columns of text characters.

This value is normally managed automatically by `resize()` based on the current font size, and should not be changed. You CAN make the `display_columns()` larger than the width of the widget; text in the terminal will simply run off the screen edge and be clipped; the only way to reveal that text is if the user enlarges the widget, or the font size made smaller.

To change the display width, it is best to use `resize()` instead.

11.142.5.20 `display_columns()` [2/2]

```
int Fl_Terminal::display_columns (
    void ) const
```

Return terminal's display width in columns of text characters.

This value is normally managed automatically by `resize()` based on the current font size.

11.142.5.21 `display_rows()` [1/2]

```
void Fl_Terminal::display_rows (
    int drows)
```

Set terminal's display height in lines of text (rows).

This value is normally managed automatically by `resize()` based on the current font size, and should not be changed.

To change the display height, use `resize()` instead.

11.142.5.22 `display_rows()` [2/2]

```
int Fl_Terminal::display_rows (
    void ) const
```

Return terminal's display height in lines of text (rows).

This value is normally managed automatically by `resize()` based on the current font size.

11.142.5.23 draw()

```
void Fl_Terminal::draw (
    void ) [virtual]
```

Draws the entire [Fl_Terminal](#).

Lets the group draw itself first (scrollbars should be only members), followed by the terminal's screen contents.

Reimplemented from [Fl_Group](#).

11.142.5.24 draw_buff()

```
void Fl_Terminal::draw_buff (
    int Y) const [protected]
```

Draws the buffer position we are scrolled to onto the FLTK screen starting at pixel position Y.

This can be anywhere in the ring buffer, not just the 'active display'; depends on what position the scrollbar is set to.

Handles attributes, colors, text selections, cursor.

Parameters

in	Y	top position of top left character in the window in FLTK coordinates

11.142.5.25 draw_row()

```
void Fl_Terminal::draw_row (
    int grow,
    int Y) const [protected]
```

Draw the specified global row, which is the row in `ring_chars[]`.

The global row includes history + display buffers.

Parameters

in	<i>grow</i>	row number
in	Y	top position of characters in the row in FLTK coordinates

11.142.5.26 draw_row_bg()

```
void Fl_Terminal::draw_row_bg (
    int grow,
    int X,
    int Y) const [protected]
```

Draw the background for the specified `ring_chars[]` global row `grow` starting at FLTK coords X and Y.

Note we may be called to draw display, or even history if we're scrolled back. If there's any change in bg color, we draw the filled rects here.

If the bg color for a character is the special "see through" color 0xffffffff, no pixels are drawn.

Parameters

in	<i>grow</i>	row number
in	X,Y	top left corner of the row in FLTK coordinates

11.142.5.27 error_char() [1/2]

```
void Fl_Terminal::error_char (
    const char * val) [inline]
```

Sets the "error character" utf8 string shown for invalid utf8 or bad ANSI sequences if [show_unknown\(\)](#) is true.

Default: "¿".

See also

[show_unknown\(bool\)](#)

11.142.5.28 error_char() [2/2]

```
const char * Fl_Terminal::error_char (
    void ) const [inline]
```

Returns the "error character" utf8 string, which is shown for invalid utf8 or bad ANSI sequences if [show_unknown\(\)](#) is true.

See also

[show_unknown\(bool\)](#)

11.142.5.29 get_selection()

```
bool Fl_Terminal::get_selection (
    int & srow,
    int & scol,
    int & erow,
    int & ecol) const [protected]
```

Return mouse selection's start/end position in the ring buffer, if any.

Ensures (start < end) to allow walking 'forward' thru selection, left-to-right, top-to-bottom. The row/col values are indexes into the entire ring buffer.

Example: walk the characters of the mouse selection:

```
// Get selection
int srow,scol,erow,ecol;
if (get_selection(srow,scol,erow,ecol)) { // mouse selection exists?
    // Walk entire selection from start to end
    for (int row=srow; row<=erow; row++) { // walk rows of selection
        const Utf8Char *u8c = u8c_ring_row(row); // ptr to first character in row
        int col_start = (row==srow) ? scol : 0; // start row? start at scol
        int col_end = (row==erow) ? ecol : ring_cols(); // end row? end at ecol
        u8c += col_start; // include col offset (if any)
        for (int col=col_start; col<=col_end; col++,u8c++) { // walk columns
            ..do something with each char at *u8c..
        }
    }
}
```

Returns:

- true – valid selection values returned
- false – no selection was made, returned values undefined

See also

[walk_selection\(\)](#), [is_selection\(\)](#)

Parameters

<i>srow</i>	starting row for selection
<i>scol</i>	starting column for selection
<i>erow</i>	ending row for selection
<i>ecol</i>	ending column for selection

11.142.5.30 h_to_row()

```
int Fl_Terminal::h_to_row (
    int H) const [protected]
```

Given a height in pixels, return number of rows that "fits" into that area.

This is used by the constructor to size the row/cols to fit the widget size.

11.142.5.31 handle()

```
int Fl_Terminal::handle (
    int e) [virtual]
```

Handle FLTK events.

Reimplemented from [Fl_Group](#).

11.142.5.32 handle_unknown_char() [1/2]

```
int Fl_Terminal::handle_unknown_char (
    int drow,
    int dcol) [protected]
```

Handle an unknown char by either emitting an error symbol to the tty, or do nothing, depending on the user configurable value of [show_unknown\(\)](#).

This writes the "unknown" character to the display position (drow,dcol) if [show_unknown\(\)](#) is true.

Returns 1 if tty modified, 0 if not.

See also

[show_unknown\(\)](#)

11.142.5.33 handle_unknown_char() [2/2]

```
int Fl_Terminal::handle_unknown_char (
    void ) [protected]
```

Handle an unknown char by either emitting an error symbol to the tty, or do nothing, depending on the user configurable value of [show_unknown\(\)](#).

This writes the "unknown" character to the output stream if [show_unknown\(\)](#) is true.

Returns 1 if tty modified, 0 if not.

See also

[show_unknown\(\)](#)

11.142.5.34 history_lines()

```
void Fl_Terminal::history_lines (
    int val)
```

Set the number of lines of screen history.

Large values can be briefly heavy on cpu and memory usage.

11.142.5.35 history_use()

```
int Fl_Terminal::history_use (
    void ) const
```

Returns how many lines are "in use" by the screen history buffer.

This value will be 0 if history was recently cleared with e.g. [clear_history\(\)](#) or "<ESC>c".

Return value will be in the range 0 .. ([history_lines\(\)](#)-1).

11.142.5.36 hscrollbar_style() [1/2]

```
void Fl_Terminal::hscrollbar_style (
    ScrollbarStyle val)
```

Set the horizontal scrollbar behavior style.

This determines when the scrollbar is visible.

ScrollbarStyle enum	Description
SCROLLBAR_ON	Horizontal scrollbar always displayed.
SCROLLBAR_OFF	Horizontal scrollbar never displayed.
SCROLLBAR_AUTO	Horizontal scrollbar displayed whenever widget width hides columns.

The default style is SCROLLBAR_AUTO.

See also

[ScrollbarStyle](#)

11.142.5.37 hscrollbar_style() [2/2]

```
Fl_Terminal::ScrollbarStyle Fl_Terminal::hscrollbar_style (
    void ) const
```

Get the horizontal scrollbar behavior style.

This determines when the scrollbar is visible.

Value will be one of the [Fl_Terminal::ScrollbarStyle](#) enum values.

See also

[hscrollbar_style\(Fl_Terminal::ScrollbarStyle\)](#)

11.142.5.38 insert_char()

```
void Fl_Terminal::insert_char (
    char c,
    int rep) [protected]
```

Insert char 'c' at the current cursor position for 'rep' times.

Does not wrap; characters at end of line are lost.

11.142.5.39 insert_rows()

```
void Fl_Terminal::insert_rows (
    int count) [protected]
```

Insert (count) rows at current cursor position.

Causes rows below to scroll down, and empty lines created. Lines deleted by scroll down are NOT moved into the scroll history.

11.142.5.40 is_inside_selection()

```
bool Fl_Terminal::is_inside_selection (
    int grow,
    int gcol) const [protected]
```

Is global row/column inside the current mouse selection?

Returns

- true – (grow, gcol) is inside a valid selection.
- false – (grow, gcol) is outside, or no valid selection.

11.142.5.41 output_translate()

```
void Fl_Terminal::output_translate (
    Fl_Terminal::OutFlags val)
```

Sets the combined output translation flags to `val`.

`val` can be sensible combinations of the OutFlags bit flags.

The default is `LF_TO_CRLF`, so that `\n` will generate both carriage-return (CR) and line-feed (LF).

For `\r` and `\n` to be handled literally, use `output_translate(Fl_Terminal::OutFlags::OFF)`;

To disable all output translations, use 0 or `Fl_Terminal::OutFlags::OFF`.

11.142.5.42 plot_char() [1/2]

```
void Fl_Terminal::plot_char (
    char c,
    int drow,
    int dcol)
```

Plot the ASCII character `c` at the terminal's display position (`drow`,`dcol`).

The character **MUST** be printable (in range 0x20 - 0x7e), and is displayed using the current text color/attributes.

Characters outside that range are either ignored or print the error character (¿), depending on `show_unknown(bool)`.

This is a very low level method.

No range checking is done on `drow`,`dcol`:

- `drow` must be in range 0..`(display_rows()-1)`
- `dcol` must be in range 0..`(display_columns()-1)`
- Does not trigger redraws
- Does NOT handle control codes, ANSI or XTERM escape sequences.

See also

`show_unknown(bool)`, `handle_unknown_char()`, `is_printable()`

11.142.5.43 plot_char() [2/2]

```
void Fl_Terminal::plot_char (
    const char * text,
    int len,
    int drow,
    int dcol)
```

Plot the UTF-8 character `text` of length `len` at display position (`drow`,`dcol`).

The character is displayed using the current text color/attributes.

This is a very low level method.

No range checking is done on `drow`,`dcol`:

- `drow` must be in range 0..`(display_rows()-1)`
- `dcol` must be in range 0..`(display_columns()-1)`
- Does not trigger redraws
- Does not handle control codes, ANSI or XTERM escape sequences.
- Invalid UTF-8 chars show the error character (¿) depending on `show_unknown(bool)`.

See also

`handle_unknown_char()`

11.142.5.44 print_char() [1/2]

```
void Fl_Terminal::print_char (
    char c)
```

Prints single ASCII char `c` at current cursor position, and advances the cursor.

The character is displayed at the current cursor position using the current text color/attributes.

- `c` must be ASCII, not utf-8
- Does not trigger redraws

11.142.5.45 print_char() [2/2]

```
void Fl_Terminal::print_char (
    const char * text,
    int len = -1)
```

Prints single UTF-8 char `text` of optional byte length `len` at current cursor position, and advances the cursor if the character is printable.

Handles ASCII and control codes (CR, LF, etc).

The character is displayed at the current cursor position using the current text color/attributes.

Handles control codes and can be used to construct ANSI/XTERM escape sequences.

- If optional `len` isn't specified or `<0`, `strlen(text)` is used.
- `text` must not be NULL.
- `len` must not be 0.
- `text` must be a single char only (whether UTF-8 or ASCII)
- `text` can be an ASCII character, though not as efficient as [print_char\(\)](#)
- Invalid UTF-8 chars show the error character (`¿`) depending on [show_unknown\(bool\)](#).
- Does not trigger redraws

See also

[show_unknown\(bool\)](#), [handle_unknown_char\(\)](#)

11.142.5.46 printf()

```
void Fl_Terminal::printf (
    const char * fmt,
    ...)
```

Appends printf formatted messages to the terminal.

The string can contain UTF-8, crlf's, and ANSI sequences are also supported. Example:

```
#include <FL/Fl_Terminal.H>
int main(..) {
    :
    // Create a terminal, and append some messages to it
    Fl_Terminal *tty = new Fl_Terminal(..);
    :
    // Append three lines of formatted text to the buffer
    tty->printf("The current date is: %s.\nThe time is: %s\n", date_str, time_str);
    tty->printf("The current PID is %ld.\n", (long) getpid());
    :
}
```

Note

The expanded string is currently limited to 1024 characters (including NULL). For longer strings use [append\(\)](#) which has no string limits.

11.142.5.47 redraw_rate()

```
void Fl_Terminal::redraw_rate (
    float val)
```

Set the maximum rate redraw speed in floating point seconds if [redraw_style\(\)](#) is set to RATE_LIMITED. When output is sent to the terminal, rather than calling [redraw\(\)](#) right away, a timer is started with this value indicating how long to wait before calling [redraw\(\)](#), causing the output to be shown. 0.10 is recommended (1/10th of a second), to limit redraws to no more than 10 redraws per second.

The value that works best depends on how fast data arrives, and how fast the font system can draw text at runtime. Values too small cause too many redraws to occur, causing the terminal to get backlogged if large bursts of data arrive quickly. Values too large cause realtime output to be too "choppy".

11.142.5.48 redraw_style() [1/2]

```
void Fl_Terminal::redraw_style (
    RedrawStyle val)
```

Set how [Fl_Terminal](#) manages screen redrawing.

This setting is relevant when [Fl_Terminal](#) is used for high bandwidth data; too many redraws will slow things down, too few cause redraws to be 'choppy' when realtime data comes in.

Redrawing can be cpu intensive, depending on how many rows/cols are being displayed; worst case: large display + small font. Speed largely depends on the end user's graphics hardware and font drawing system.

RedrawStyle enum	Description
NO_REDRAW	App must call redraw() as needed to update text to screen
RATE_LIMITED	Rate limited, timer controlled redraws. (DEFAULT) See redraw_rate()
PER_WRITE	Redraw triggered <i>every</i> call to append() / printf() / etc.

The default style is RATE_LIMITED, which is the easiest to use, and automates redrawing to be capped at 10 redraws per second max. See [redraw_rate\(float\)](#) to control this automated redraw speed.

See also

[redraw_rate\(\)](#), [RedrawStyle](#)

11.142.5.49 redraw_style() [2/2]

```
Fl_Terminal::RedrawStyle Fl_Terminal::redraw_style (
    void ) const
```

Get the redraw style.

This determines when the terminal redraws itself while text is being added to it.

Value will be one of the [Fl_Terminal::RedrawStyle](#) enum values.

See also

[redraw_style\(Fl_Terminal::RedrawStyle\)](#)

11.142.5.50 reset_terminal()

```
void Fl_Terminal::reset_terminal (
    void )
```

Resets terminal to default colors, clears screen, history and mouse selection, homes cursor, resets tabstops.

Same as "<ESC>c"

11.142.5.51 `resize()`

```
void Fl_Terminal::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Handle widget resizing, such as if user resizes parent window.

This may increase the column width of the widget if the width of the widget is made larger than it was.

Note

Resizing currently does not rewrap existing text. Currently enlarging makes room for longer lines, and shrinking the size lets long lines run off the right edge of the display, hidden from view. This behavior may change in the future to rewrap.

Reimplemented from [Fl_Group](#).

11.142.5.52 `scroll()`

```
void Fl_Terminal::scroll (
    int rows) [protected]
```

Scroll the display up(+) or down(-) the specified rows.

- Negative value scrolls "down", clearing top line, and history unaffected.
- Postive value scrolls "up", clearing bottom line, rotating top line into history.

11.142.5.53 `scrollbar_actual_size()`

```
int Fl_Terminal::scrollbar_actual_size (
    void ) const
```

Returns the scrollbar's actual "trough size", which is the width of FL_VERTICAL scrollbars, or height of FL_↔ HORIZONTAL scrollbars.

If `scrollbar_size()` is zero (default), then the value of the global [Fl::scrollbar_size\(\)](#) is returned, which is the default global scrollbar size for the entire application.

11.142.5.54 `scrollbar_size()` [1/2]

```
void Fl_Terminal::scrollbar_size (
    int val)
```

Set the pixel size of both horizontal and vertical scrollbar's "trough" to `val`.

Setting `val` to the special value 0 causes the widget to track the global [Fl::scrollbar_size\(\)](#).

Use non-zero values *only* if you need to override the global [Fl::scrollbar_size\(\)](#) size.

See also

[Fl::scrollbar_size\(\)](#), [scrollbar_actual_size\(\)](#)

11.142.5.55 `scrollbar_size()` [2/2]

```
int Fl_Terminal::scrollbar_size (
    void ) const
```

Get current pixel size of all the scrollbar's troughs for this widget, or zero if the global [Fl::scrollbar_size\(\)](#) is being used (default).

If this value returns zero, this widget's scrollbars are using the global [Fl::scrollbar_size\(\)](#), in which case use [scrollbar_actual_size\(\)](#) to get the actual (effective) pixel scrollbar size being used.

Returns

Scrollbar trough size in pixels, or 0 if the global [Fl::scrollbar_size\(\)](#) is being used.

See also

[Fl::scrollbar_size\(int\), scrollbar_actual_size\(\)](#)

11.142.5.56 selection_extend()

```
bool Fl_Terminal::selection_extend (
    int X,
    int Y) [protected]
```

Extend selection to FLTK coords X,Y.

Returns true if extended, false if nothing done (X,Y offscreen)

11.142.5.57 selection_text()

```
const char * Fl_Terminal::selection_text (
    void ) const
```

Return text selection (for copy()/paste() operations)

- Returns allocated NULL terminated string for entire selection.
- Caller must free() this memory when done.
- Unicode safe.

11.142.5.58 selection_text_len()

```
int Fl_Terminal::selection_text_len (
    void ) const
```

Return byte length of all UTF-8 chars in selection, or 0 if no selection.

NOTE: Length includes trailing white on each line.

11.142.5.59 show_unknown() [1/2]

```
void Fl_Terminal::show_unknown (
    bool val)
```

Set the "show unknown" flag.

If true, invalid utf8 and invalid ANSI sequences will be shown with the error character "¿".

If false, errors characters won't be shown.

See also

[handle_unknown_char\(\), error_char\(const char*\)](#).

11.142.5.60 show_unknown() [2/2]

```
bool Fl_Terminal::show_unknown (
    void ) const
```

Return the "show unknown" flag.

See also

[show_unknown\(bool\), error_char\(const char*\)](#).

11.142.5.61 text()

```
const char * Fl_Terminal::text (
    bool lines_below_cursor = false) const
```

Return a string copy of all lines in the terminal (including history).

The returned string is allocated with strdup(3), which the caller must free.

If 'lines_below_cursor' is false (default), lines below the cursor on down to the bottom of the display are ignored, and not included in the returned string.

If 'lines_below_cursor' is true, then all lines in the display are returned including any below the cursor, even if all are blank.

Example use:

```
Fl_Terminal *tty = new Fl_Terminal(..);
:
const char *s = tty->text(); // get a copy of the terminal's contents
printf("Terminal's contents is:\n%s\n", s);
free((void*)s); // free() the copy when done!
```

Parameters

in	<i>lines_below_cursor</i>	include lines below cursor, default: false

Returns

A string allocated with `strdup(3)` which must be free'd, text is UTF-8.

11.142.5.62 textattrib() [1/2]

```
uchar Fl_Terminal::textattrib () const
```

Get text attribute bits (underline, inverse, etc).

This is the default attribute used for all newly printed text.

See also

[textattrib\(uchar\)](#), [Fl_Terminal::Attrib](#)

11.142.5.63 textattrib() [2/2]

```
void Fl_Terminal::textattrib (
    uchar val)
```

Set text attribute bits (underline, inverse, etc).

This will be the default attribute used for all newly printed text.

See also

[Fl_Terminal::Attrib](#)

11.142.5.64 textbgcolor()

```
void Fl_Terminal::textbgcolor (
    Fl_Color val)
```

Set text background color to fltk color `val` used by any new text added.

Use this for temporary color changes, similar to `<ESC>[48;2;<R>;<G>;<B>m`

Colors set this way will NOT be influenced by the xterm Dim/Bold color intensity attributes. For that, use [textbgcolor_xterm\(uchar\)](#) instead.

This setting does *not* affect the 'default' text colors used by `<ESC>[0m`, `<ESC>c`, [reset_terminal\(\)](#), etc. To set that too, also set [textbgcolor_default\(Fl_Color\)](#), e.g.

```
// Set both 'current' and 'default' colors
Fl_Color darkamber = 0x20100000;
tty->textbgcolor(darkamber); // set 'current' bg color
tty->textbgcolor_default(darkamber); // set 'default' bg color used by ESC[0m reset
```

The special color value 0xffffffff (all ff's) is the "see through" color, which lets the widget's own [Fl_Group::color\(\)](#) show through behind the text. This special text background color is the *default*, and is what most situations need.

See also

[textbgcolor_default\(Fl_Color\)](#), [textbgcolor_xterm\(uchar\)](#)

11.142.5.65 textbgcolor_default() [1/2]

```
void Fl_Terminal::textbgcolor_default (
    Fl_Color val)
```

Set the default text background color used by any new text added after a reset (<ESC>c, <ESC>[0m, or [reset_terminal\(\)](#)).

Does not affect the 'current' text background color; use [textbgcolor\(Fl_Color\)](#) to set that.

The special color value 0xffffffff (all ff's) is the "see through" color, which lets the widget's own [Fl_Group::color\(\)](#) show through behind the text. This special text background color is the *default*, and is what most situations need.

See also

[textbgcolor\(Fl_Color\)](#)

11.142.5.66 textbgcolor_default() [2/2]

```
Fl_Color Fl_Terminal::textbgcolor_default (
    void ) const [inline]
```

Return text's default background color.

See also

[textbgcolor\(\)](#)

11.142.5.67 textbgcolor_xterm()

```
void Fl_Terminal::textbgcolor_xterm (
    uchar val)
```

Sets the background text color as one of the 8 'xterm color' values.

This will be the foreground color used for all newly printed text, similar to the <ESC>[#m escape sequence, where # is between 40 and 47.

This color will be reset to the default bg color if [reset_terminal\(\)](#) is called, or by <ESC>c, <ESC>[0m, etc.

The xterm color intensity values can be influenced by the Dim/Bold/Normal modes (which can be set with e.g. <ESC>[1m, [textattrib\(\)](#), etc), so the actual RGB values of these colors allow room for Dim/Bold to influence their brightness. For instance, "Normal Red" is not full brightness to allow "Bold Red" to be brighter. This goes for all colors except 'Black', which is not influenced by Dim or Bold; Black is always Black.

The 8 color xterm values are:

- 0 = Black
- 1 = Red
- 2 = Green
- 3 = Yellow
- 4 = Blue
- 5 = Magenta
- 6 = Cyan
- 7 = White

See also

[textbgcolor_default\(Fl_Color\)](#)

11.142.5.68 textcolor()

```
void Fl_Terminal::textcolor (
    Fl_Color val)
```

Set the text color for the terminal.

This is a convenience method that sets *both* [textfgcolor\(\)](#) and [textfgcolor_default\(\)](#), ensuring both are set to the same value.

Colors set this way will NOT be influenced by the xterm Dim/Bold color intensity attributes. For that, use [textcolor_xterm\(\)](#) instead.

See also

[textfgcolor\(Fl_Color\)](#), [textfgcolor_default\(Fl_Color\)](#), [textbgcolor_xterm\(uchar\)](#)

11.142.5.69 textfgcolor()

```
void Fl_Terminal::textfgcolor (
    Fl_Color val)
```

Set text foreground drawing color to fltk color `val` used by any new text added.

Use this for temporary color changes, similar to `<ESC>[38;2;<R>;<G>;<B>m`

Colors set this way will NOT be influenced by the xterm Dim/Bold color intensity attributes. For that, use [textfgcolor_xterm\(uchar\)](#) instead.

This setting does *not* affect the 'default' text colors used by `<ESC>[0m`, `<ESC>c`, [reset_terminal\(\)](#), etc. To change both the current *and* default fg color, also use [textfgcolor_default\(Fl_Color\)](#). Example:

```
// Set both 'current' and 'default' colors
Fl_Color amber = 0xd0704000;
tty->textfgcolor(amber);           // set 'current' fg color
tty->textfgcolor_default(amber); // set 'default' fg color used by ESC[0m reset
```

See also

[textfgcolor_default\(Fl_Color\)](#), [textfgcolor_xterm\(uchar\)](#)

11.142.5.70 textfgcolor_default() [1/2]

```
void Fl_Terminal::textfgcolor_default (
    Fl_Color val)
```

Set the default text foreground color used by `<ESC>c`, `<ESC>[0m`, and [reset_terminal\(\)](#).

Does not affect the 'current' text foreground color; use [textfgcolor\(Fl_Color\)](#) to set that.

See also

[textfgcolor\(Fl_Color\)](#)

11.142.5.71 textfgcolor_default() [2/2]

```
Fl_Color Fl_Terminal::textfgcolor_default (
    void ) const [inline]
```

Return text's default foreground color.

See also

[textfgcolor\(\)](#)

11.142.5.72 `textfgcolor_xterm()`

```
void Fl_Terminal::textfgcolor_xterm (
    uchar val)
```

Sets the foreground text color as one of the 8 'xterm color' values.

This will be the foreground color used for all newly printed text, similar to the `<ESC>[#m` escape sequence, where `#` is between 30 and 37.

This color will be reset to the default fg color if [reset_terminal\(\)](#) is called, or by `<ESC>c`, `<ESC>[0m`, etc.

The xterm color intensity values can be influenced by the Dim/Bold/Normal modes (which can be set with e.g. `<ESC>[1m`, [textattrib\(\)](#), etc), so the actual RGB values of these colors allow room for Dim/Bold to influence their brightness. For instance, "Normal Red" is not full brightness to allow "Bold Red" to be brighter. This goes for all colors except 'Black', which is not influenced by Dim or Bold; Black is always Black.

The 8 color xterm values are:

- 0 = Black
- 1 = Red
- 2 = Green
- 3 = Yellow
- 4 = Blue
- 5 = Magenta
- 6 = Cyan
- 7 = White

See also

[textfgcolor_default\(Fl_Color\)](#)

11.142.5.73 `textfont()`

```
void Fl_Terminal::textfont (
    Fl_Font val)
```

Sets the font used for all text displayed in the terminal.

This affects all existing text (in display and history) as well as any newly printed text.

Only monospace fonts are recommended, such as `FL_COURIER` or `FL_SCREEN`. Custom fonts configured with [Fl::set_font\(\)](#) will also work, as long as they are monospace.

11.142.5.74 `textsize()`

```
void Fl_Terminal::textsize (
    Fl_Fontsize val)
```

Sets the font size used for all text displayed in the terminal.

This affects all existing text (in display and history) as well as any newly printed text.

Changing this will affect the [display_rows\(\)](#) and [display_columns\(\)](#).

11.142.5.75 `u8c_disp_row()`

```
Fl_Terminal::Utf8Char * Fl_Terminal::u8c_disp_row (
    int drow) [protected]
```

Return pointer to the first u8c character in row `drow` of the display.

- 'drow' is indexed relative to the beginning of the display buffer.

- This can be used to walk all columns in the specified row, e.g.

```
// Print all chars in first row of display (ASCII and UTF-8)
Utf8Char *u8c = u8c_disp_row(0);           // first char of first display row
int scol = 0, ecol = disp_cols();          // start/end for column loop
for (int col=scol; col<ecol; col++,u8c++) { // loop from first char to last
    char *text = u8c->text_utf8();          // text string for char
    int len = u8c->length();                 // text string length for char
    ::printf("<%.s>", len, text);           // print potentially multibyte char
}
```

11.142.5.76 u8c_hist_row()

```
Fl_Terminal::Utf8Char * Fl_Terminal::u8c_hist_row (
    int hrow) [protected]
```

Return u8c for beginning of a row inside the scrollbar history.

'hrow' is indexed relative to the beginning of the scrollbar history buffer.

See also

[u8c_disp_row\(int\)](#) for example use.

11.142.5.77 u8c_hist_use_row()

```
Fl_Terminal::Utf8Char * Fl_Terminal::u8c_hist_use_row (
    int hurow) [protected]
```

Return u8c for beginning of row hurow inside the 'in use' part of the scrollbar history.

'hurow' is indexed relative to the beginning of the 'in use' part of the scrollbar history buffer. This may be a different from [u8c_hist_row\(int\)](#) if the history was recently cleared, and there aren't many (or any) rows in the history buffer that have been populated with scrollbar text yet.

See also

[u8c_disp_row\(int\)](#) for example use.

11.142.5.78 u8c_ring_row()

```
Fl_Terminal::Utf8Char * Fl_Terminal::u8c_ring_row (
    int grow) [protected]
```

Return UTF-8 char for row grow in the ring buffer.

grow is globally indexed relative to the beginning of the ring buffer, so this method can access ANY character in the entire ring buffer (hist or disp) by the global index.

Scrolling offset is NOT applied; this is raw access to the ring's rows.

Should really ONLY be used for making a complete copy of the ring.

Example:

```
// Walk ALL rows and cols in the ring buffer..
for (int row=0; row<ring.rows(); row++) {
    Utf8Char *u8c = ring.u8c_ring_row(row);
    for (int col=0; col<ring.cols(); col++,u8c++) {
        ..make use of u8c->xxx() methods..
    }
}
```

11.142.5.79 vprintf()

```
void Fl_Terminal::vprintf (
    const char * fmt,
    va_list ap)
```

Appends printf formatted messages to the terminal.

Subclasses can use this to implement their own [printf\(\)](#) functionality.

The string can contain UTF-8, crlf's, and ANSI sequences are also supported when [ansi\(bool\)](#) is set to 'true'.

Note

The expanded string is currently limited to 1024 characters (including NULL). For longer strings use [append\(\)](#) which has no string limits.

Parameters

<i>fmt</i>	is a printf format string for the message text.
<i>ap</i>	is a <code>va_list</code> created by <code>va_start()</code> and closed with <code>va_end()</code> , which the caller is responsible for handling.

11.142.5.80 w_to_col()

```
int Fl_Terminal::w_to_col (
    int W) const [protected]
```

Given a width in pixels, return number of columns that "fits" into that area.
This is used by the constructor to size the row/cols to fit the widget size.

11.142.5.81 walk_selection()

```
const Fl_Terminal::Utf8Char * Fl_Terminal::walk_selection (
    const Utf8Char * u8c,
    int & row,
    int & col) const [protected]
```

Walk the mouse selection one character at a time from beginning to end, returning a `Utf8Char*` to the next character in the selection, or NULL if the end was reached, or if there's no selection.

This is easier to use for walking the selection than [get_selection\(\)](#).

`u8c` should start out as NULL, rewinding to the beginning of the selection. If the returned `Utf8Char*` is not NULL, `row` and `col` return the character's row/column position in the ring buffer.

```
// EXAMPLE: Walk the entire mouse selection, if any
int row,col; // the returned row/col for each char
Utf8Char *u8c = NULL; // start with NULL to begin walk
while ((u8c = walk_selection(u8c, row, col))) { // loop until end char reached
    ..do something with *u8c..
}
```

See also

[get_selection\(\)](#), [is_selection\(\)](#)

Parameters

<i>u8c</i>	NULL on first iter
<i>row</i>	returned row#
<i>col</i>	returned col#

11.142.6 Member Data Documentation**11.142.6.1 hscrollbar**

```
Fl_Scrollbar* Fl_Terminal::hscrollbar
```

Horizontal scrollbar.

This is public so it can be accessed directly, e.g.

- [hscrollbar->value\(void\)](#) returns the column offset position from the left edge of the display; 0 being the left edge (default).
- [hscrollbar->value\(int\)](#) similarly sets the column offset, which should be in the range [0 .. [Fl_Scrollbar::maximum\(\)](#)].
- [hscrollbar->step\(double\)](#) sets the smoothness of scrolling, default is 0.25 for 4 steps of motion per column.

11.142.6.2 scrollbar

`Fl_Scrollbar*` `Fl_Terminal::scrollbar`

Vertical scrollbar.

This is public so it can be accessed directly, e.g.

- `scrollbar->value(void)` returns the row offset from the bottom of the display, 0 being the bottom (default).
- `scrollbar->value(int)` similarly sets the row offset, which should be in the range [0 .. `Fl_Scrollbar::maximum()`].
- `scrollbar->step(double)` sets the smoothness of scrolling, default is 0.25 for 4 steps of motion per column.

Todo Support `scrollbar_left/right()` - See `Fl_Browser_::scrollbar` docs
Support new `ScrollbarStyle`

The documentation for this class was generated from the following files:

- `Fl_Terminal.H`
- `Fl_Terminal.cxx`

11.143 Fl_Text_Buffer Class Reference

This class manages Unicode text displayed in one or more `Fl_Text_Display` widgets.

```
#include <Fl_Text_Buffer.H>
```

Public Member Functions

- void `add_modify_callback` (`Fl_Text_Modify_Cb` bufModifiedCB, void *cbArg)
Adds a callback function that is called whenever the text buffer is modified.
- void `add_predelete_callback` (`Fl_Text_Predelete_Cb` bufPreDelCB, void *cbArg)
Adds a callback routine to be called before text is deleted from the buffer.
- char * `address` (int pos)
Convert a byte offset in buffer into a memory address.
- const char * `address` (int pos) const
Convert a byte offset in buffer into a memory address.
- void `append` (const char *t, int addedLength=-1)
Appends the text string to the end of the buffer.
- int `appendfile` (const char *file, int buflen=128 * 1024)
Appends the named file to the end of the buffer.
- char `byte_at` (int pos) const
Returns the raw byte at the specified position pos in the buffer.
- void `call_modify_callbacks` ()
Calls all modify callbacks that have been registered using the `add_modify_callback()` method.
- void `call_predelete_callbacks` ()
Calls the stored pre-delete callback procedure(s) for this buffer to update the changed area(s) on the screen and any other listeners.
- bool `can_redo` () const
Check if undo is enabled and if the last undo action can be redone.
- bool `can_undo` () const
Check if undo is enabled and if the last action can be undone.
- void `canUndo` (char flag=1)
Enable or disable undo actions for this text buffer.
- unsigned int `char_at` (int pos) const
Returns the character at the specified position pos in the buffer.
- void `copy` (`Fl_Text_Buffer` *fromBuf, int fromStart, int fromEnd, int toPos)

- Copies text from another [FI_Text_Buffer](#) to this one.*

 - int [count_displayed_characters](#) (int lineStartPos, int targetPos) const

Count the number of displayed characters between buffer position `lineStartPos` and `targetPos`.
- int [count_lines](#) (int startPos, int endPos) const

Counts the number of newlines between `startPos` and `endPos` in buffer.
- int [estimate_lines](#) (int startPos, int endPos, int lineLen) const

Estimate the number of newlines between `startPos` and `endPos` in buffer.
- int [findchar_backward](#) (int startPos, unsigned int searchChar, int *foundPos) const

Search backwards in buffer `buf` for character `searchChar`, starting with the character before `startPos`, returning the result in `foundPos`.
- int [findchar_forward](#) (int startPos, unsigned searchChar, int *foundPos) const

Finds the next occurrence of the specified character.
- [FI_Text_Buffer](#) (int requestedSize=0, int preferredGapSize=1024)

Create an empty text buffer of a pre-determined size.
- int **highlight** ()

Returns a non-zero value if text has been highlighted, 0 otherwise.
- void **highlight** (int start, int end)

Highlights the specified text within the buffer.
- int **highlight_position** (int *start, int *end)

Highlights the specified text between `start` and `end` within the buffer.
- const [FI_Text_Selection](#) * **highlight_selection** () const

Returns the current highlight selection.
- char * [highlight_text](#) ()

Returns the highlighted text.
- void [insert](#) (int pos, const char *[text](#), int insertedLength=-1)

Inserts null-terminated string `text` at position `pos`.
- int [insertfile](#) (const char *file, int pos, int buflen=128 *1024)

Inserts a file at the specified position.
- bool [is_word_separator](#) (int pos) const

Returns whether character at position `pos` is a word separator.
- int [length](#) () const

Returns the number of bytes in the buffer.
- int [line_end](#) (int pos) const

Finds and returns the position of the end of the line containing position `pos` (which is either a pointer to the newline character ending the line or a pointer to one character beyond the end of the buffer).
- int [line_start](#) (int pos) const

Returns the position of the start of the line containing position `pos`.
- char * [line_text](#) (int pos) const

Returns the text from the entire line containing the specified character position.
- int [loadfile](#) (const char *file, int buflen=128 *1024)

Loads a text file into the buffer.
- int [next_char](#) (int ix) const

Returns the index of the next character.
- int **next_char_clipped** (int ix) const
- int [outputfile](#) (const char *file, int start, int end, int buflen=128 *1024)

Writes the specified portions of the text buffer to a file.
- int [prev_char](#) (int ix) const

Returns the index of the previous character.
- int **prev_char_clipped** (int ix) const
- [FI_Text_Selection](#) * **primary_selection** ()

Returns the primary selection.

- const [FI_Text_Selection](#) * **primary_selection** () const
Returns the primary selection.
- void [printf](#) (const char *fmt,...)
Appends printf formatted messages to the end of the buffer.
- int **redo** (int *cp=0)
Redo previous undo action.
- void [remove](#) (int start, int end)
Deletes a range of characters in the buffer.
- void **remove_modify_callback** (FI_Text_Modify_Cb bufModifiedCB, void *cbArg)
Removes a modify callback.
- void **remove_predelete_callback** (FI_Text_Predelete_Cb predelCB, void *cbArg)
Removes a callback routine `bufPreDeleteCB` associated with argument `cbArg` to be called before text is deleted from the buffer.
- void **remove_secondary_selection** ()
Removes the text from the buffer corresponding to the secondary text selection object.
- void **remove_selection** ()
Removes the text in the primary selection.
- void [replace](#) (int start, int end, const char *text, int insertedLength=-1)
Deletes the characters between `start` and `end`, and inserts the null-terminated string `text` in their place in the buffer.
- void **replace_secondary_selection** (const char *text)
Replaces the text from the buffer corresponding to the secondary text selection object with the new string `text`.
- void **replace_selection** (const char *text)
Replaces the text in the primary selection.
- int [rewind_lines](#) (int startPos, int nLines)
Finds and returns the position of the first character of the line `nLines` backwards from `startPos` (not counting the character pointed to by `startpos` if that is a newline) in the buffer.
- int [savefile](#) (const char *file, int buflen=128 * 1024)
Saves a text file from the current buffer.
- int [search_backward](#) (int startPos, const char *searchString, int *foundPos, int matchCase=0) const
Search backwards in buffer for string `searchString`, starting with the character at `startPos`, returning the result in `foundPos`.
- int [search_forward](#) (int startPos, const char *searchString, int *foundPos, int matchCase=0) const
Search forwards in buffer for string `searchString`, starting with the character `startPos`, and returning the result in `foundPos`.
- void **secondary_select** (int start, int end)
Selects a range of characters in the secondary selection.
- int **secondary_selected** ()
Returns a non-zero value if text has been selected in the secondary text selection, 0 otherwise.
- const [FI_Text_Selection](#) * **secondary_selection** () const
Returns the secondary selection.
- int **secondary_selection_position** (int *start, int *end)
Returns the current selection in the secondary text selection object.
- char * [secondary_selection_text](#) ()
Returns the text in the secondary selection.
- void **secondary_unselect** ()
Clears any selection in the secondary text selection object.
- void **select** (int start, int end)
Selects a range of characters in the buffer.
- int **selected** () const
Returns a non-zero value if text has been selected, 0 otherwise.
- int **selection_position** (int *start, int *end)

- Gets the selection position.*

 - char * [selection_text](#) ()

Returns the currently selected text.
- int [skip_displayed_characters](#) (int lineStartPos, int nChars)

Count forward from buffer position `startPos` in displayed characters.
- int [skip_lines](#) (int startPos, int nLines)

Finds the first character of the line `nLines` forward from `startPos` in the buffer and returns its position.
- int [tab_distance](#) () const

Gets the tab width.
- void [tab_distance](#) (int tabDist)

Set the hardware tab distance (width) used by all displays for this buffer, and used in computing offsets for rectangular selection operations.
- char * [text](#) () const

Get a copy of the entire contents of the text buffer.
- void [text](#) (const char *text)

Replaces the entire contents of the text buffer.
- char * [text_range](#) (int start, int end) const

Get a copy of a part of the text buffer.
- int [undo](#) (int *cp=0)

Undo text modification according to the undo variables or insert text from the undo buffer.
- void [unhighlight](#) ()

Unhighlights text in the buffer.
- void [unselect](#) ()

Cancels any previous selection on the primary text selection object.
- int [utf8_align](#) (int) const

Align an index into the buffer to the current or previous UTF-8 boundary.
- void [vprintf](#) (const char *fmt, va_list ap)

Can be used by subclasses that need their own `printf()` style functionality.
- int [word_end](#) (int pos) const

Returns the position corresponding to the end of the word.
- int [word_start](#) (int pos) const

Returns the position corresponding to the start of the word.
- ~FI_Text_Buffer ()

Frees a text buffer.

Public Attributes

- int [input_file_was_transcoded](#)

true if the loaded file has been transcoded to UTF-8.
- void(* [transcoding_warning_action](#))(FI_Text_Buffer *)

Pointer to a function called after reading a non UTF-8 encoded file.

Static Public Attributes

- static const char * [file_encoding_warning_message](#)

This message may be displayed using the `fl_alert()` function when a file which was not UTF-8 encoded is input.

Protected Member Functions

- int **apply_undo** (FI_Text_Undo_Action *action, int *cursorPos)
Apply the current undo/redo operation, called from [undo\(\)](#) or [redo\(\)](#).
- void **call_modify_callbacks** (int pos, int nDeleted, int nInserted, int nRestyled, const char *deletedText) const
Calls the stored modify callback procedure(s) for this buffer to update the changed area(s) on the screen and any other listeners.
- void **call_predelete_callbacks** (int pos, int nDeleted) const
Calls the stored pre-delete callback procedure(s) for this buffer to update the changed area(s) on the screen and any other listeners.
- int **insert_** (int pos, const char *text, int insertedLength=-1)
Internal (non-redisplaying) version of [insert\(\)](#).
- void **move_gap** (int pos)
Move the gap to start at a new position.
- void **reallocate_with_gap** (int newGapStart, int newGapLen)
*Reallocates the text storage in the buffer to have a gap starting at *newGapStart* and a gap size of *newGapLen*, preserving the buffer's current contents.*
- void **redisplay_selection** (FI_Text_Selection *oldSelection, FI_Text_Selection *newSelection) const
Calls the stored redisplay procedure(s) for this buffer to update the screen for a change in a selection.
- void **remove_** (int start, int end)
Internal (non-redisplaying) version of [remove\(\)](#).
- void **remove_selection_** (FI_Text_Selection *sel)
*Removes the text from the buffer corresponding to *sel*.*
- void **replace_selection_** (FI_Text_Selection *sel, const char *text)
*Replaces the *text* in selection *sel*.*
- char * **selection_text_** (FI_Text_Selection *sel) const
- void **update_selections** (int pos, int nDeleted, int nInserted)
Updates all of the selections in the buffer for changes in the buffer's text.

Protected Attributes

- char * **mBuf**
allocated memory where the text is stored
- char **mCanUndo**
if this buffer is used for attributes, it must not do any undo calls
- void ** **mCbArgs**
caller arguments for modifyProcs above
- int **mCursorPosHint**
hint for reasonable cursor position after a buffer modification operation
- int **mGapEnd**
points to the first character after the gap
- int **mGapStart**
points to the first character of the gap
- FI_Text_Selection **mHighlight**
highlighted areas
- int **mLength**
*length of the text in the buffer (the length of the buffer itself must be calculated: *gapEnd* - *gapStart* + *length*)*
- FI_Text_Modify_Cb * **mModifyProcs**
procedures to call when buffer is modified to redisplay contents
- int **mNModifyProcs**
number of modify-redisplay procs attached
- int **mNPredeleteProcs**

- number of pre-delete procs attached*
- void ** **mPredeleteCbArgs**
caller argument for pre-delete proc above
- Fl_Text_Predelete_Cb * **mPredeleteProcs**
procedure to call before text is deleted from the buffer; at most one is supported.
- int **mPreferredGapSize**
the default allocation for the text gap is 1024 bytes and should only be increased if frequent and large changes in buffer size are expected
- [Fl_Text_Selection](#) **mPrimary**
highlighted areas
- Fl_Text_Undo_Action_List * **mRedoList**
List of redo event.
- [Fl_Text_Selection](#) **mSecondary**
highlighted areas
- int **mTabDist**
equiv.
- Fl_Text_Undo_Action * **mUndo**
local undo event
- Fl_Text_Undo_Action_List * **mUndoList**
List of undo event.

11.143.1 Detailed Description

This class manages Unicode text displayed in one or more [Fl_Text_Display](#) widgets.

All text in [Fl_Text_Buffer](#) must be encoded in UTF-8. All indices used in the function calls must be aligned to the start of a UTF-8 sequence. All indices and pointers returned will be aligned. All functions that return a single character will return that in an unsigned int in UCS-4 encoding.

The [Fl_Text_Buffer](#) class is used by the [Fl_Text_Display](#) and [Fl_Text_Editor](#) to manage complex text data and is based upon the excellent NEdit text editor engine - see <https://sourceforge.net/projects/nedit/>.

11.143.2 Constructor & Destructor Documentation

11.143.2.1 Fl_Text_Buffer()

```
Fl_Text_Buffer::Fl_Text_Buffer (
    int requestedSize = 0,
    int preferredGapSize = 1024)
```

Create an empty text buffer of a pre-determined size.

Parameters

<i>requestedSize</i>	use this to avoid unnecessary re-allocation if you know exactly how much the buffer will need to hold
<i>preferredGapSize</i>	Initial size for the buffer gap (empty space in the buffer where text might be inserted if the user is typing sequential characters)

11.143.3 Member Function Documentation

11.143.3.1 add_modify_callback()

```
void Fl_Text_Buffer::add_modify_callback (
    Fl_Text_Modify_Cb bufModifiedCB,
    void * cbArg)
```

Adds a callback function that is called whenever the text buffer is modified.

The callback function is declared as follows:

```
typedef void (*Fl_Text_Modify_Cb)(int pos, int nInserted, int nDeleted,
    int nRestyled, const char* deletedText,
    void* cbArg);
```

11.143.3.2 address() [1/2]

```
char * Fl_Text_Buffer::address (
    int pos) [inline]
```

Convert a byte offset in buffer into a memory address.

Parameters

<i>pos</i>	byte offset into buffer

Returns

byte offset converted to a memory address

11.143.3.3 address() [2/2]

```
const char * Fl_Text_Buffer::address (
    int pos) const [inline]
```

Convert a byte offset in buffer into a memory address.

Parameters

<i>pos</i>	byte offset into buffer

Returns

byte offset converted to a memory address

11.143.3.4 append()

```
void Fl_Text_Buffer::append (
    const char * t,
    int addedLength = -1) [inline]
```

Appends the text string to the end of the buffer.

Parameters

<i>t</i>	UTF-8 encoded text
<i>addedLength</i>	number of bytes to append, or -1 to indicate <i>t</i> is null-terminated

11.143.3.5 appendfile()

```
int Fl_Text_Buffer::appendfile (
    const char * file,
    int buflen = 128*1024) [inline]
```

Appends the named file to the end of the buffer.

See also [insertfile\(\)](#).

11.143.3.6 byte_at()

```
char Fl_Text_Buffer::byte_at (
    int pos) const
```

Returns the raw byte at the specified position *pos* in the buffer.

Positions start at 0.

Parameters

<i>pos</i>	byte offset into buffer

Returns

unencoded raw byte

11.143.3.7 can_redo()

```
bool Fl_Text_Buffer::can_redo () const
```

Check if undo is enabled and if the last undo action can be redone.

Check if undo is enabled and if the last undo action can be redone.

See also

[canUndo\(\)](#)

11.143.3.8 can_undo()

```
bool Fl_Text_Buffer::can_undo () const
```

Check if undo is enabled and if the last action can be undone.

See also

[canUndo\(\)](#)

11.143.3.9 canUndo()

```
void Fl_Text_Buffer::canUndo (
    char flag = 1)
```

Enable or disable undo actions for this text buffer.

Undo actions are enable for text buffer by default. If used as a style buffer in [Fl_Text_Display](#), undo actions are disabled as they are handled by the text buffer.

See also

[can_undo\(\)](#)

11.143.3.10 char_at()

```
unsigned int Fl_Text_Buffer::char_at (
    int pos) const
```

Returns the character at the specified position *pos* in the buffer.

Positions start at 0.

Parameters

<i>pos</i>	byte offset into buffer, <i>pos</i> must be at a UTF-8 character boundary

Returns

Unicode UCS-4 encoded character

11.143.3.11 copy()

```
void Fl_Text_Buffer::copy (
    Fl_Text_Buffer * fromBuf,
    int fromStart,
    int fromEnd,
    int toPos)
```

Copies text from another [Fl_Text_Buffer](#) to this one.

Parameters

<i>fromBuf</i>	source text buffer, may be the same as this
<i>fromStart</i>	byte offset into buffer
<i>fromEnd</i>	byte offset into buffer
<i>toPos</i>	destination byte offset into buffer

11.143.3.12 count_displayed_characters()

```
int Fl_Text_Buffer::count_displayed_characters (
    int lineStartPos,
    int targetPos) const
```

Count the number of displayed characters between buffer position `lineStartPos` and `targetPos`. Displayed characters are the characters shown on the screen to represent characters in the buffer, where tabs and control characters are expanded.

11.143.3.13 count_lines()

```
int Fl_Text_Buffer::count_lines (
    int startPos,
    int endPos) const
```

Counts the number of newlines between `startPos` and `endPos` in buffer. The character at position `endPos` is not counted.

11.143.3.14 estimate_lines()

```
int Fl_Text_Buffer::estimate_lines (
    int startPos,
    int endPos,
    int lineLen) const
```

Estimate the number of newlines between `startPos` and `endPos` in buffer. This call takes line wrapping into account. It assumes a line break at every `lineLen` characters after the beginning of a line.

11.143.3.15 findchar_backward()

```
int Fl_Text_Buffer::findchar_backward (
    int startPos,
    unsigned int searchChar,
    int * foundPos) const
```

Search backwards in buffer `buf` for character `searchChar`, starting with the character *before* `startPos`, returning the result in `foundPos`.

Returns 1 if found, 0 if not. The difference between this and [search_backward\(\)](#) is that it's optimized for single characters. The overall performance of the text widget is dependent on its ability to count lines quickly, hence searching for a single character: newline.

Parameters

<i>startPos</i>	byte offset to start position
<i>searchChar</i>	UCS-4 character that we want to find
<i>foundPos</i>	byte offset where the character was found

Returns

1 if found, 0 if not

11.143.3.16 findchar_forward()

```
int Fl_Text_Buffer::findchar_forward (
    int startPos,
    unsigned searchChar,
    int * foundPos) const
```

Finds the next occurrence of the specified character.

Search forwards in buffer for character `searchChar`, starting with the character `startPos`, and returning the result in `foundPos`. Returns 1 if found, 0 if not. The difference between this and [search_forward\(\)](#) is that it's optimized for single characters. The overall performance of the text widget is dependent on its ability to count lines quickly, hence searching for a single character: `newline`.

Parameters

<i>startPos</i>	byte offset to start position
<i>searchChar</i>	UCS-4 character that we want to find
<i>foundPos</i>	byte offset where the character was found

Returns

1 if found, 0 if not

11.143.3.17 highlight_text()

```
char * Fl_Text_Buffer::highlight_text ()
```

Returns the highlighted text.

When you are done with the text, free it using the `free()` function.

11.143.3.18 insert()

```
void Fl_Text_Buffer::insert (
    int pos,
    const char * text,
    int insertedLength = -1)
```

Inserts null-terminated string `text` at position `pos`.

Parameters

<i>pos</i>	insertion position as byte offset (must be UTF-8 character aligned)
<i>text</i>	UTF-8 encoded text
<i>insertedLength</i>	number of bytes to insert, or -1 to indicate <code>text</code> is null-terminated

11.143.3.19 insert_()

```
int Fl_Text_Buffer::insert_ (
    int pos,
    const char * text,
    int insertedLength = -1) [protected]
```

Internal (non-redisplaying) version of [insert\(\)](#).

Returns the length of text inserted (this is just `strlen(text)` if `insertedLength == -1`, however this calculation can be expensive and the length will be required by any caller who will continue on to call `redisplay`). `pos` must be contiguous with the existing text in the buffer (i.e. not past the end).

Returns

the number of bytes inserted

11.143.3.20 insertfile()

```
int Fl_Text_Buffer::insertfile (
    const char * file,
    int pos,
    int buflen = 128*1024)
```

Inserts a file at the specified position.

Returns

- 0 on success
- non-zero on error (strerror() contains reason)
- 1 indicates open for read failed (no data loaded)
- 2 indicates error occurred while reading data (data was partially loaded)

File can be UTF-8 or CP1252 encoded. If the input file is not UTF-8 encoded, the [Fl_Text_Buffer](#) widget will contain data transcoded to UTF-8. By default, the message [Fl_Text_Buffer::file_encoding_warning_message](#) will warn the user about this.

See also

[input_file_was_transcoded](#) and [transcoding_warning_action](#).

11.143.3.21 is_word_separator()

```
bool Fl_Text_Buffer::is_word_separator (
    int pos) const
```

Returns whether character at position `pos` is a word separator.

Pos must be at a character boundary.

11.143.3.22 length()

```
int Fl_Text_Buffer::length () const [inline]
```

Returns the number of bytes in the buffer.

Returns

size of text in bytes

11.143.3.23 line_end()

```
int Fl_Text_Buffer::line_end (
    int pos) const
```

Finds and returns the position of the end of the line containing position `pos` (which is either a pointer to the newline character ending the line or a pointer to one character beyond the end of the buffer).

Parameters

<i>pos</i>	byte index into buffer

Returns

byte offset to line end

11.143.3.24 line_start()

```
int Fl_Text_Buffer::line_start (
    int pos) const
```

Returns the position of the start of the line containing position `pos`.

Parameters

<i>pos</i>	byte index into buffer

Returns

byte offset to line start

11.143.3.25 line_text()

```
char * Fl_Text_Buffer::line_text (
    int pos) const
```

Returns the text from the entire line containing the specified character position. When you are done with the text, free it using the `free()` function.

Parameters

<i>pos</i>	byte index into buffer

Returns

copy of UTF-8 text, must be free'd

11.143.3.26 loadfile()

```
int Fl_Text_Buffer::loadfile (
    const char * file,
    int buflen = 128*1024) [inline]
```

Loads a text file into the buffer.
See also [insertfile\(\)](#).

11.143.3.27 next_char()

```
int Fl_Text_Buffer::next_char (
    int ix) const
```

Returns the index of the next character.

Parameters

<i>ix</i>	index to the current character

11.143.3.28 outputfile()

```
int Fl_Text_Buffer::outputfile (
    const char * file,
    int start,
    int end,
    int buflen = 128*1024)
```

Writes the specified portions of the text buffer to a file.
Returns

- 0 on success
- non-zero on error (`strerror()` contains reason)
- 1 indicates open for write failed (no data saved)
- 2 indicates error occurred while writing data (data was partially saved)

See also

[savefile\(const char *file, int buflen\)](#)

11.143.3.29 prev_char()

```
int Fl_Text_Buffer::prev_char (
    int ix) const
```

Returns the index of the previous character.

Parameters

<i>ix</i>	index to the current character

11.143.3.30 printf()

```
void Fl_Text_Buffer::printf (
    const char * fmt,
    ...)
```

Appends printf formatted messages to the end of the buffer.

Example:

```
#include <FL/Fl_Text_Display.H>
int main(..) {
    :
    // Create a text display widget and assign it a text buffer
    Fl_Text_Display *tdsp = new Fl_Text_Display(..);
    Fl_Text_Buffer *tbuf = new Fl_Text_Buffer();
    tdsp->buffer(tbuf);
    :
    // Append three lines of formatted text to the buffer
    tbuf->printf("The current date is: %s.\nThe time is: %s\n", date_str, time_str);
    tbuf->printf("The current PID is %ld.\n", (long)getpid());
    :
}
```

Note

The expanded string is currently limited to 1024 characters.

Parameters

<i>in</i>	<i>fmt</i>	is a printf format string for the message text.

11.143.3.31 remove()

```
void Fl_Text_Buffer::remove (
    int start,
    int end)
```

Deletes a range of characters in the buffer.

Parameters

<i>start</i>	byte offset to first character to be removed
<i>end</i>	byte offset to character after last character to be removed

11.143.3.32 remove_()

```
void Fl_Text_Buffer::remove_ (
    int start,
    int end) [protected]
```

Internal (non-redisplaying) version of [remove\(\)](#).

Removes the contents of the buffer between *start* and *end* (and moves the gap to the site of the delete).

11.143.3.33 replace()

```
void Fl_Text_Buffer::replace (
    int start,
    int end,
    const char * text,
    int insertedLength = -1)
```

Deletes the characters between `start` and `end`, and inserts the null-terminated string `text` in their place in the buffer.

Parameters

<i>start</i>	byte offset to first character to be removed and new insert position
<i>end</i>	byte offset to character after last character to be removed
<i>text</i>	UTF-8 encoded text
<i>insertedLength</i>	number of bytes to insert, or -1 to indicate <code>text</code> is null-terminated

11.143.3.34 rewind_lines()

```
int Fl_Text_Buffer::rewind_lines (
    int startPos,
    int nLines)
```

Finds and returns the position of the first character of the line `nLines` backwards from `startPos` (not counting the character pointed to by `startpos` if that is a newline) in the buffer.

`nLines == 0` means find the beginning of the line.

11.143.3.35 savefile()

```
int Fl_Text_Buffer::savefile (
    const char * file,
    int buflen = 128*1024) [inline]
```

Saves a text file from the current buffer.

Returns

- 0 on success
- non-zero on error (`strerror()` contains reason)
- 1 indicates open for write failed (no data saved)
- 2 indicates error occurred while writing data (data was partially saved)

See also

[outfile\(const char *file, int start, int end, int buflen\)](#)

11.143.3.36 search_backward()

```
int Fl_Text_Buffer::search_backward (
    int startPos,
    const char * searchString,
    int * foundPos,
    int matchCase = 0) const
```

Search backwards in buffer for string `searchString`, starting with the character *at* `startPos`, returning the result in `foundPos`.

Returns 1 if found, 0 if not.

Parameters

<i>startPos</i>	byte offset to start position
<i>searchString</i>	UTF-8 string that we want to find
<i>foundPos</i>	byte offset where the string was found
<i>matchCase</i>	if set, match character case

Returns

1 if found, 0 if not

11.143.3.37 search_forward()

```
int Fl_Text_Buffer::search_forward (
    int startPos,
    const char * searchString,
    int * foundPos,
    int matchCase = 0) const
```

Search forwards in buffer for string `searchString`, starting with the character `startPos`, and returning the result in `foundPos`.

Returns 1 if found, 0 if not.

Parameters

<i>startPos</i>	byte offset to start position
<i>searchString</i>	UTF-8 string that we want to find
<i>foundPos</i>	byte offset where the string was found
<i>matchCase</i>	if set, match character case

Returns

1 if found, 0 if not

11.143.3.38 secondary_selection_text()

```
char * Fl_Text_Buffer::secondary_selection_text ()
```

Returns the text in the secondary selection.

When you are done with the text, free it using the `free()` function.

11.143.3.39 selection_text()

```
char * Fl_Text_Buffer::selection_text ()
```

Returns the currently selected text.

When you are done with the text, free it using the `free()` function.

11.143.3.40 skip_displayed_characters()

```
int Fl_Text_Buffer::skip_displayed_characters (
    int lineStartPos,
    int nChars)
```

Count forward from buffer position `startPos` in displayed characters.

Displayed characters are the characters shown on the screen to represent characters in the buffer, where tabs and control characters are expanded.

Parameters

<i>lineStartPos</i>	byte offset into buffer
<i>nChars</i>	number of bytes that are sent to the display

Returns

byte offset in input after all output bytes are sent

11.143.3.41 tab_distance()

```
int Fl_Text_Buffer::tab_distance () const [inline]
```

Gets the tab width.

The tab width is measured in characters. The pixel position is calculated using an average character width.

11.143.3.42 text() [1/2]

```
char * Fl_Text_Buffer::text () const
```

Get a copy of the entire contents of the text buffer.

Memory is allocated to contain the returned string, which the caller must free.

Returns

newly allocated text buffer - must be free'd, text is UTF-8

11.143.3.43 text() [2/2]

```
void Fl_Text_Buffer::text (
    const char * text)
```

Replaces the entire contents of the text buffer.

Parameters

<i>text</i>	Text must be valid UTF-8. If null, an empty string is substituted.

11.143.3.44 text_range()

```
char * Fl_Text_Buffer::text_range (
    int start,
    int end) const
```

Get a copy of a part of the text buffer.

Return a copy of the text between *start* and *end* character positions from text buffer *buf*. Positions start at 0, and the range does not include the character pointed to by *end*. When you are done with the text, free it using the `free()` function.

Parameters

<i>start</i>	byte offset to first character
<i>end</i>	byte offset after last character in range

Returns

newly allocated text buffer - must be free'd, text is UTF-8

11.143.3.45 undo()

```
int Fl_Text_Buffer::undo (
    int * cursorPos = 0)
```

Undo text modification according to the undo variables or insert text from the undo buffer.

Take the previous changes and undo them.

Return the previous cursor position in cursorPos. Returns 1 if the undo was applied. CursorPos will be at a character boundary.

11.143.3.46 vprintf()

```
void Fl_Text_Buffer::vprintf (
    const char * fmt,
    va_list ap)
```

Can be used by subclasses that need their own [printf\(\)](#) style functionality.

Note

The expanded string is currently limited to 1024 characters.

Parameters

in	<i>fmt</i>	is a printf format string for the message text.
in	<i>ap</i>	is a va_list created by va_start() and closed with va_end(), which the caller is responsible for handling.

11.143.3.47 word_end()

```
int Fl_Text_Buffer::word_end (
    int pos) const
```

Returns the position corresponding to the end of the word.

Parameters

<i>pos</i>	byte index into buffer

Returns

byte offset to word end

11.143.3.48 word_start()

```
int Fl_Text_Buffer::word_start (
    int pos) const
```

Returns the position corresponding to the start of the word.

Parameters

<i>pos</i>	byte index into buffer

Returns

byte offset to word start

11.143.4 Member Data Documentation

11.143.4.1 file_encoding_warning_message

```
const char * Fl_Text_Buffer::file_encoding_warning_message [static]
```

Initial value:

```
=
"Displayed text contains the UTF-8 transcoding\n"
"of the input file which was not UTF-8 encoded.\n"
"Some changes may have occurred."
```

This message may be displayed using the `fl_alert()` function when a file which was not UTF-8 encoded is input.

11.143.4.2 mTabDist

```
int Fl_Text_Buffer::mTabDist [protected]
```

equiv.

number of characters in a tab

11.143.4.3 transcoding_warning_action

```
void(* Fl_Text_Buffer::transcoding_warning_action) (Fl_Text_Buffer *)
```

Pointer to a function called after reading a non UTF-8 encoded file.

This function is called after reading a file if the file content was transcoded to UTF-8. Its default implementation calls `fl_alert()` with the text of [file_encoding_warning_message](#). No warning message is displayed if this pointer is set to NULL. Use [input_file_was_transcoded](#) to be informed if file input required transcoding to UTF-8.

The documentation for this class was generated from the following files:

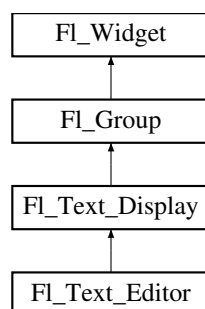
- `Fl_Text_Buffer.H`
- `Fl_Text_Buffer.cxx`

11.144 Fl_Text_Display Class Reference

Rich text display widget.

```
#include <Fl_Text_Display.H>
```

Inheritance diagram for `Fl_Text_Display`:



Classes

- struct [Style_Table_Entry](#)

This structure associates the color, font, and font size of a string to draw with an attribute mask matching `attr`.

Public Types

- enum {
[NORMAL_CURSOR](#) , [CARET_CURSOR](#) , [DIM_CURSOR](#) , [BLOCK_CURSOR](#) ,
[HEAVY_CURSOR](#) , [SIMPLE_CURSOR](#) }
text display cursor shapes enumeration
- enum { [CURSOR_POS](#) , [CHARACTER_POS](#) }

the character position is the left edge of a character, whereas the cursor is thought to be between the centers of two consecutive characters.

- enum {
DRAG_NONE = -2 , **DRAG_START_DND** = -1 , **DRAG_CHAR** = 0 , **DRAG_WORD** = 1 ,
DRAG_LINE = 2 }
drag types - they match [Fl::event_clicks\(\)](#) so that single clicking to start a collection selects by character, double clicking selects by word and triple clicking selects by line.
- enum { **WRAP_NONE** , **WRAP_AT_COLUMN** , **WRAP_AT_PIXEL** , **WRAP_AT_BOUNDS** }
wrap types - used in [wrap_mode\(\)](#)
- enum {
ATTR_BGCOLOR = 0x0001 , **ATTR_BGCOLOR_EXT_** = 0x0002 , **ATTR_BGCOLOR_EXT** = 0x0003 ,
ATTR_UNDERLINE = 0x0004 ,
ATTR_GRAMMAR = 0x0008 , **ATTR_SPELLING** = 0x000C , **ATTR_STRIKE_THROUGH** = 0x0010 ,
ATTR_LINES_MASK = 0x001C }
attribute flags in [Style_Table_Entry.attr](#)
- typedef void(*) **Unfinished_Style_Cb**(int, void *)

Public Member Functions

- [Fl_Text_Buffer](#) * [buffer](#) () const
Gets the current text buffer associated with the text widget.
- void [buffer](#) ([Fl_Text_Buffer](#) &buf)
Sets the current text buffer associated with the text widget.
- void [buffer](#) ([Fl_Text_Buffer](#) *buf)
Attach a text buffer to display, replacing the current buffer (if any).
- double [col_to_x](#) (double col) const
Convert a column number into an x pixel position.
- int [count_lines](#) (int start, int end, bool start_pos_is_line_start) const
Count the number of lines between two positions.
- [Fl_Color](#) [cursor_color](#) () const
Gets the text cursor color.
- void [cursor_color](#) ([Fl_Color](#) n)
Sets the text cursor color.
- int **cursor_style** () const
- void [cursor_style](#) (int style)
Sets the text cursor style.
- virtual void [display_needs_recalc](#) ()
Schedule a [recalc_display\(\)](#) to be done on next [draw\(\)](#).
- [Fl_Text_Display](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new text display widget.
- int [get_absolute_top_line_number](#) () const
Returns the absolute (non-wrapped) line number of the first line displayed.
- [Fl_Color](#) [grammar_underline_color](#) () const
Gets the underline color for style attribute **ATTR_GRAMMAR**.
- void [grammar_underline_color](#) ([Fl_Color](#) color)
Sets the underline color for style attribute **ATTR_GRAMMAR**.
- int [handle](#) (int e) **FL_OVERRIDE**
Event handling.
- void **hide_cursor** ()
Hides the text cursor.
- void [highlight_data](#) ([Fl_Text_Buffer](#) *styleBuffer, const [Style_Table_Entry](#) *styleTable, int nStyles, char unfinishedStyle, **Unfinished_Style_Cb** unfinishedHighlightCB, void *cbArg)
Attach (or remove) highlight information in text display and redisplay.

- int [in_selection](#) (int x, int y) const
Check if a pixel position is within the primary selection.
- void [insert](#) (const char *text)
Inserts "text" at the current cursor location.
- int [insert_position](#) () const
Gets the position of the text insertion cursor for text display.
- void [insert_position](#) (int newPos)
Sets the position of the text insertion cursor for text display.
- int [line_end](#) (int startPos, bool startPosIsLineStart) const
Returns the end of a line.
- int [line_start](#) (int pos) const
Return the beginning of a line.
- [FI_Align](#) [linenumber_align](#) () const
Returns the alignment used for line numbers (if enabled).
- void [linenumber_align](#) ([FI_Align](#) val)
Set alignment for line numbers (if enabled).
- [FI_Color](#) [linenumber_bgcolor](#) () const
Returns the background color used for line numbers (if enabled).
- void [linenumber_bgcolor](#) ([FI_Color](#) val)
Set the background color used for line numbers (if enabled).
- [FI_Color](#) [linenumber_fgcolor](#) () const
Return the foreground color used for line numbers (if enabled).
- void [linenumber_fgcolor](#) ([FI_Color](#) val)
Set the foreground color used for line numbers (if enabled).
- [FI_Font](#) [linenumber_font](#) () const
Return the font used for line numbers (if enabled).
- void [linenumber_font](#) ([FI_Font](#) val)
Set the font used for line numbers (if enabled).
- const char * [linenumber_format](#) () const
Returns the line number printf() format string.
- void [linenumber_format](#) (const char *val)
Sets the printf() style format string used for line numbers.
- [FI_Fontsize](#) [linenumber_size](#) () const
Return the font size used for line numbers (if enabled).
- void [linenumber_size](#) ([FI_Fontsize](#) val)
Set the font size used for line numbers (if enabled).
- int [linenumber_width](#) () const
Return the screen area width provided for line numbers.
- void [linenumber_width](#) (int width)
Set width of screen area for line numbers.
- int [move_down](#) ()
Moves the current insert position down one line.
- int [move_left](#) ()
Moves the current insert position left one character.
- int [move_right](#) ()
Moves the current insert position right one character.
- int [move_up](#) ()
Moves the current insert position up one line.
- void [next_word](#) (void)
Moves the current insert position right one word.
- void [overstrike](#) (const char *text)

- Replaces text at the current insert position.*

 - int [position_style](#) (int lineStartPos, int lineLen, int lineIndex) const

Find the correct style for a character.
- int [position_to_xy](#) (int pos, int *x, int *y) const

Convert a character index into a pixel position.
- void [previous_word](#) (void)

Moves the current insert position left one word.
- virtual void [recalc_display](#) ()

Recalculate the display's visible lines and scrollbar sizes.
- void [redisplay_range](#) (int start, int end)

Marks text from start to end as needing a redraw.
- void [resize](#) (int X, int Y, int W, int H) [FL_OVERRIDE](#)

Change the size of the displayed text area.
- int [rewind_lines](#) (int startPos, int nLines)

Skip a number of lines back.
- void [scroll](#) (int topLineNum, int horizOffset)

Scrolls the current buffer to start at the specified line and column.
- int [scroll_col](#) ()
- int [scroll_row](#) ()
- [FL_Align](#) [scrollbar_align](#) () const

Gets the scrollbar alignment type.
- void [scrollbar_align](#) ([FL_Align](#) a)

Sets the scrollbar alignment type.
- int [scrollbar_size](#) () const

Gets the current size of the scrollbars' troughs, in pixels.
- void [scrollbar_size](#) (int newSize)

Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.
- int [scrollbar_width](#) () const

Returns the global value `Fl::scrollbar_size()` unless a specific `scrollbar_width_` has been set.
- void [scrollbar_width](#) (int width)

Sets the global `Fl::scrollbar_size()`, and forces this instance of the widget to use it.
- [FL_Color](#) [secondary_selection_color](#) () const

Gets the background color for the secondary selection block.
- void [secondary_selection_color](#) ([FL_Color](#) color)

Sets the background color for the secondary selection block.
- int [shortcut](#) () const
- void [shortcut](#) (int s)
- void [show_cursor](#) (int b=1)

Shows the text cursor.
- void [show_insert_position](#) ()

Scrolls the text buffer to show the current insert position.
- int [skip_lines](#) (int startPos, int nLines, bool startPosIsLineStart)

Skip a number of lines forward.
- [FL_Color](#) [spelling_underline_color](#) () const

Gets the underline color for style attribute `ATTR_SPELLING`.
- void [spelling_underline_color](#) ([FL_Color](#) color)

Sets the underline color for style attribute `ATTR_SPELLING`.
- [FL_Text_Buffer](#) * [style_buffer](#) () const

Gets the current style buffer associated with the text widget.
- [FL_Color](#) [textcolor](#) () const

Gets the default color of text in the widget.

- void [textcolor](#) ([FL_Color](#) n)
Sets the default color of text in the widget.
- [FL_Font](#) [textfont](#) () const
Gets the default font used when drawing text in the widget.
- void [textfont](#) ([FL_Font](#) s)
Sets the default font used when drawing text in the widget.
- [FL_Fonsize](#) [textsize](#) () const
Gets the default size of text in the widget.
- void [textsize](#) ([FL_Fonsize](#) s)
Sets the default size of text in the widget.
- int [word_end](#) (int pos) const
Moves the insert position to the end of the current word.
- int [word_start](#) (int pos) const
Moves the insert position to the beginning of the current word.
- void [wrap_mode](#) (int wrap, int wrap_margin)
Set the new text wrap mode.
- int [wrapped_column](#) (int row, int column) const
Nobody knows what this function does.
- int [wrapped_row](#) (int row) const
Nobody knows what this function does.
- double [x_to_col](#) (double x) const
Convert an x pixel position into a column number.
- [~FL_Text_Display](#) ()
Free a text display and release its associated memory.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget](#) *[_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void [add](#) ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void [add](#) ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void [add_resizable](#) ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget](#) *const * [array](#) () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * [as_group](#) () const [FL_OVERRIDE](#)
- [FL_Group](#) * [as_group](#) () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void [begin](#) ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * [child](#) (int n) const
Returns the n'th child.
- int [children](#) () const
Returns how many child widgets the group has.
- void [clear](#) ()
Deletes all child widgets from memory recursively.
- unsigned int [clip_children](#) ()
Returns the current clipping mode.
- void [clip_children](#) (int c)

- Controls whether the group widget clips the drawing of child widgets to its bounding box.*

 - virtual int **delete_child** (int n)

Removes the widget at `index` from the group and deletes it.
- void **end** ()
- Exactly the same as `current(this->parent())`.*
- int **find** (const **FL_Widget** &o) const
- See `int FL_Group::find(const FL_Widget *w) const`.*
- int **find** (const **FL_Widget** *) const
- Searches the children for the widget and returns the index.*
- **FL_Group** (int, int, int, int, const char *s=0)
- Creates a new **FL_Group** widget using the given position, size, and label string.*
- void **focus** (**FL_Widget** *w)
- void **forms_end** ()
- This is for forms compatibility only.*
- void **init_sizes** ()
- Resets the internal array of widget sizes and positions.*
- void **insert** (**FL_Widget** &, int i)
- The widget is removed from its current group (if any) and then inserted into this group.*
- void **insert** (**FL_Widget** &o, **FL_Widget** *before)
- This does `insert(w, find(before))`.*
- void **remove** (**FL_Widget** &)
- Removes a widget from the group but does not delete it.*
- void **remove** (**FL_Widget** *o)
- Removes the widget `o` from the group.*
- void **remove** (int index)
- Removes the widget at `index` from the group but does not delete it.*
- **FL_Widget** * **resizable** () const
- Returns the group's resizable widget.*
- void **resizable** (**FL_Widget** &o)
- Sets the group's resizable widget.*
- void **resizable** (**FL_Widget** *o)
- The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
- virtual **~FL_Group** ()
- The destructor also deletes all the children.*

Public Member Functions inherited from **FL_Widget**

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
- Activates the widget.*
- unsigned int **active** () const
- Returns whether the widget is active.*
- int **active_r** () const
- Returns whether the widget and all of its parents are active.*
- **FL_Align** **align** () const
- Gets the label alignment.*
- void **align** (**FL_Align** alignment)
- Sets the label alignment.*
- long **argument** () const
- Gets the current user data (long) argument that is passed to the callback function.*

- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FL_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).
- virtual class [FL_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FL_Window](#) * [as_window](#) ()
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).
- virtual [FL_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FL_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FL_Boxtype](#) new_box)
Sets the box type for the widget.
- [FL_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FL_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FL_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FL_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([FL_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FL_Color](#) bg, [FL_Color](#) sel)

- Sets the background and selection color of the widget.*

 - `FL_Color color2 () const`

For back compatibility only.
- `void color2 (unsigned a)`

For back compatibility only.
- `int contains (const FL_Widget *w) const`

Checks if w is a child of this widget.
- `void copy_label (const char *new_label)`

Sets the current label.
- `void copy_tooltip (const char *text)`

Sets the current tooltip text.
- `uchar damage () const`

Returns non-zero if `draw()` needs to be called.
- `void damage (uchar c)`

Sets the damage bits for the widget.
- `void damage (uchar c, int x, int y, int w, int h)`

Sets the damage bits for an area inside the widget.
- `int damage_resize (int, int, int, int)`

Internal use only.
- `void deactivate ()`

Deactivates the widget.
- `FL_Image * deimage ()`

Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`

Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`

Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`

Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`

Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.
- `void draw_label (int, int, int, int, FL_Align) const`

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`

Gets the widget height.
- `virtual void hide ()`

Makes a widget invisible.
- `int horizontal_label_margin ()`

Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`

Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`

Gets the image that is used as part of the widget label when in the active state.
- `const FL_Image * image () const`

Gets the image that is used as part of the widget label when in the active state.

- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()

- Schedules the drawing of the widget.*
 - void `redraw_label` ()
- Schedules the drawing of the label.*
 - `Fl_Color selection_color` () const
- Gets the selection color.*
 - void `selection_color` (`Fl_Color` a)
- Sets the selection color.*
 - void `set_active` ()
- Marks the widget as active without sending events or changing focus.*
 - void `set_changed` ()
- Marks the value of the widget as changed.*
 - void `set_output` ()
- Sets a widget to output only.*
 - void `set_visible` ()
- Makes the widget visible.*
 - void `set_visible_focus` ()
- Enables keyboard focus navigation with this widget.*
 - int `shortcut_label` () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*
 - void `shortcut_label` (int value)
- Sets whether the widget's label uses '&' to indicate shortcuts.*
 - virtual void `show` ()
- Makes a widget visible.*
 - void `size` (int W, int H)
- Changes the size of the widget.*
 - int `take_focus` ()
- Gives the widget the keyboard focus.*
 - unsigned int `takeevents` () const
- Returns if the widget is able to take events.*
 - int `test_shortcut` ()
- Returns true if the widget's label contains the entered '&x' shortcut.*
 - const char * `tooltip` () const
- Gets the current tooltip text.*
 - void `tooltip` (const char *text)
- Sets the current tooltip text.*
 - `Fl_Window * top_window` () const
- Returns a pointer to the top-level window for the widget.*
 - `Fl_Window * top_window_offset` (int &xoff, int &yoff) const
- Finds the x/y offset of the current widget relative to the top-level window.*
 - `uchar type` () const
- Gets the widget type.*
 - void `type` (`uchar` t)
- Sets the widget type.*
 - int `use_accents_menu` ()
- Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*
 - void * `user_data` () const
- Gets the user data for this widget.*
 - void `user_data` (`Fl_Callback_User_Data` *v, bool auto_free)
- Sets the user data for this widget.*
 - void `user_data` (void *v)
- Sets the user data for this widget.*

- int `vertical_label_margin` ()
Get the spacing between the label and the vertical edge of the widget.
- void `vertical_label_margin` (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int `visible` () const
Returns whether a widget is visible.
- unsigned int `visible_focus` () const
Checks whether this widget has a visible focus.
- void `visible_focus` (int v)
Modifies keyboard focus navigation.
- int `visible_r` () const
Returns whether a widget and all its parents are visible.
- int `w` () const
Gets the widget width.
- `Fl_When` `when` () const
Returns the conditions under which the callback is called.
- void `when` (uchar i)
Sets the flags used to decide when a callback is called.
- `Fl_Window` * `window` () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int `x` () const
Gets the widget position in its window.
- int `y` () const
Gets the widget position in its window.
- virtual `~Fl_Widget` ()
Destroys the widget.

Protected Types

- enum {
 DRAW_LINE , **FIND_INDEX** , **FIND_INDEX_FROM_ZERO** , **GET_WIDTH** ,
 FIND_CURSOR_INDEX }

Protected Types inherited from `Fl_Widget`

- enum {
 INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
 FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
 ,
 OVERRIDE = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
 ,
 MENU_WINDOW = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
 ,
 GROUP_RELATIVE = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
 = 1<<19 ,
 NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
 AUTO_DELETE_USER_DATA = 1<<23 ,
 MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
 USERFLAG1 = 1<<31 }
flags possible values enumeration.

Protected Member Functions

- void [absolute_top_line_number](#) (int oldFirstChar)
Re-calculate absolute top line number for a change in scroll position.
- void [calc_last_char](#) ()
Update last display character index.
- void [calc_line_starts](#) (int startLine, int endLine)
Update the line starts array.
- void [clear_rect](#) (int style, int x, int y, int width, int height) const
Clear a rectangle with the appropriate background color for `style`.
- void [display_insert](#) ()
Scroll the display to bring insertion cursor into view.
- void [draw](#) () [FL_OVERRIDE](#)
Draw the widget.
- void [draw_cursor](#) (int, int)
Draw a cursor with top center at `X`, `Y`.
- void [draw_line_numbers](#) (bool clearAll)
Refresh the line number area.
- void [draw_range](#) (int start, int end)
Draw a range of text.
- void [draw_string](#) (int style, int x, int y, int toX, const char *string, int nChars) const
Draw a text segment in a single style.
- void [draw_text](#) (int X, int Y, int W, int H)
Refresh a rectangle of the text display.
- void [draw_vline](#) (int visLineNum, int leftClip, int rightClip, int leftCharIndex, int rightCharIndex)
Draw a single line of text.
- int [empty_vlines](#) () const
Return true if there are lines visible with no corresponding buffer text.
- void [extend_range_for_styles](#) (int *start, int *end)
I don't know what this does!
- void [find_line_end](#) (int pos, bool start_pos_is_line_start, int *lineEnd, int *nextLineStart) const
Finds both the end of the current line and the start of the next line.
- void [find_wrap_range](#) (const char *deletedText, int pos, int nInserted, int nDeleted, int *modRangeStart, int *modRangeEnd, int *linesInserted, int *linesDeleted)
Wrapping calculations.
- int [find_x](#) (const char *s, int len, int style, int x) const
Find the index of the character that lies at the given `x` position / closest cursor position.
- int [handle_rmb](#) (int readonly)
Handle right mouse button down events.
- int [handle_vline](#) (int mode, int lineStart, int lineLen, int leftChar, int rightChar, int topClip, int bottomClip, int leftClip, int rightClip) const
Universal pixel machine.
- int [longest_vline](#) () const
Find the longest line of all visible lines.
- void [maintain_absolute_top_line_number](#) (int state)
Line numbering stuff, currently unused.
- int [maintaining_absolute_top_line_number](#) () const
Returns true if a separate absolute top line number is being maintained.
- void [measure_deleted_lines](#) (int pos, int nDeleted)
Wrapping calculations.
- double [measure_proportional_character](#) (const char *s, int colNum, int pos) const

- Wrapping calculations.*
- int [measure_vline](#) (int visLineNum) const
 - Returns the width in pixels of the displayed line pointed to by "visLineNum".*
- void [offset_line_starts](#) (int newTopLineNum)
 - Offset line start counters for a new vertical scroll position.*
- int [position_to_line](#) (int pos, int *lineNum) const
 - Convert a position index into a line number offset.*
- int [position_to_linecol](#) (int pos, int *lineNum, int *column) const
 - Find the line and column number of position pos.*
- void [reset_absolute_top_line_number](#) ()
 - Reestablish the absolute (non-wrapped) top line number.*
- int [scroll_](#) (int topLineNum, int horizOffset)
 - Scrolls the current buffer to start at the specified line and column.*
- double [string_width](#) (const char *string, int length, int style) const
 - Find the width of a string in the font of a particular style.*
- void [update_h_scrollbar](#) ()
 - Update horizontal scrollbar.*
- void [update_line_starts](#) (int pos, int charsInserted, int charsDeleted, int linesInserted, int linesDeleted, int *scrolled)
 - Update line start arrays and variables.*
- void [update_v_scrollbar](#) ()
 - Update vertical scrollbar.*
- int [vline_length](#) (int visLineNum) const
 - Count number of bytes in a visible line.*
- int [wrap_uses_character](#) (int lineEndPos) const
 - Check if the line break is caused by a newline or by line wrapping.*
- void [wrapped_line_counter](#) ([Fl_Text_Buffer](#) *buf, int startPos, int maxPos, int maxLines, bool startPosIs↵
LineStart, int styleBufOffset, int *retPos, int *retLines, int *retLineStart, int *retLineEnd, bool countLast↵
LineMissingNewLine=true) const
 - Wrapping calculations.*
- int [xy_to_position](#) (int x, int y, int PosType=CHARACTER_POS) const
 - Translate a pixel position into a character index.*
- void [xy_to_rowcol](#) (int x, int y, int *row, int *column, int PosType=CHARACTER_POS) const
 - Translate pixel coordinates into row and column.*

Protected Member Functions inherited from [Fl_Group](#)

- [Fl_Rect](#) * [bounds](#) ()
 - Returns the internal array of widget sizes and positions.*
- void [draw_child](#) ([Fl_Widget](#) &widget) const
 - Forces a child to redraw.*
- void [draw_children](#) ()
 - Draws all children of the group.*
- void [draw_outside_label](#) (const [Fl_Widget](#) &widget) const
 - Parents normally call this to draw outside labels of child widgets.*
- virtual int [on_insert](#) ([Fl_Widget](#) *, int)
 - Allow derived groups to act when a widget is added as a child.*
- virtual int [on_move](#) (int, int)
 - Allow derived groups to act when a widget is moved within the group.*
- virtual void [on_remove](#) (int)
 - Allow derived groups to act when a child widget is removed from the group.*
- int * [sizes](#) ()

Returns the internal array of widget sizes and positions.

- void **update_child** ([FI_Widget](#) &widget) const

Draws a child only if it needs it.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Static Protected Member Functions

- static void **buffer_modified_cb** (int pos, int nInserted, int nDeleted, int nRestyled, const char *deletedText, void *cbArg)
This is called whenever the buffer is modified.
- static void **buffer_predelete_cb** (int pos, int nDeleted, void *cbArg)
This is called before any characters are deleted.
- static void **h_scrollbar_cb** ([FI_Scrollbar](#) *w, [FI_Text_Display](#) *d)
Callback for drag or valueChanged on horizontal scrollbar.
- static void **scroll_timer_cb** (void *)
Timer callback for scroll events.
- static void **v_scrollbar_cb** ([FI_Scrollbar](#) *w, [FI_Text_Display](#) *d)
Callback for drag or valueChanged on vertical scrollbar.

Protected Attributes

- int **damage_range1_end**
- int **damage_range1_start**
- int **damage_range2_end**
- int **damage_range2_start**
- int **display_insert_position_hint**
- bool **display_needs_recalc_**
- int **dragging**
- int **dragPos**
- int **dragType**
- [Fl_Color](#) **grammar_underline_color_**
- [Fl_Align](#) **linenumber_align_**
- [Fl_Color](#) **linenumber_bgcolor_**
- [Fl_Color](#) **linenumber_fgcolor_**
- [Fl_Font](#) **linenumber_font_**
- const char * **linenumber_format_**
- [Fl_Fontsize](#) **linenumber_size_**
- int **mAbsTopLineNum**
- [Fl_Text_Buffer](#) * **mBuffer**
- double **mColumnScale**
- int **mContinuousWrap**
- [Fl_Color](#) **mCursor_color**
- int **mCursorOldY**
- int **mCursorOn**
- int **mCursorPos**
- int **mCursorPreferredXPos**
- int **mCursorStyle**
- int **mCursorToHint**
- int **mFirstChar**
- void * **mHighlightCBArg**
- int **mHorizOffset**
- int **mHorizOffsetHint**
- [Fl_Scrollbar](#) * **mHScrollbar**
- int **mLastChar**
- int **mLineNumLeft**
- int **mLineNumWidth**
- int * **mLineStarts**
- int **mMaxsize**
- int **mModifyingTabDistance**
- int **mNBufferLines**
- int **mNeedAbsTopLineNum**
- int **mNLinesDeleted**
- int **mNStyles**
- int **mNVisibleLines**
- [Fl_Text_Buffer](#) * **mStyleBuffer**
- const [Style_Table_Entry](#) * **mStyleTable**
- int **mSuppressResync**
- int **mTopLineNum**
- int **mTopLineNumHint**
- Unfinished_Style_Cb **mUnfinishedHighlightCB**
- char **mUnfinishedStyle**
- [Fl_Scrollbar](#) * **mVScrollbar**
- int **mWrapMarginPix**
- [Fl_Align](#) **scrollbar_align_**

- int **scrollbar_width_**
- [FL_Color](#) **secondary_selection_color_**
- int **shortcut_**
- [FL_Color](#) **spelling_underline_color_**
- struct {
 - int **h**
 - int **w**
 - int **x**
 - int **y**
 } **text_area**
- [FL_Color](#) **textcolor_**
- [FL_Font](#) **textfont_**
- [FL_Fonsize](#) **textsize_**

Friends

- void **fl_text_drag_me** (int pos, [FL_Text_Display](#) *d)
- int **fl_text_drag_prepare** (int pos, int key, [FL_Text_Display](#) *d)

Additional Inherited Members

Static Public Member Functions inherited from [FL_Group](#)

- static [FL_Group](#) * **current** ()
Returns the currently active group.
- static void **current** ([FL_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [FL_Widget](#)

- static void **default_callback** ([FL_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

11.144.1 Detailed Description

Rich text display widget.

This is the FLTK text display widget. It allows the user to view multiple lines of text and supports highlighting, word wrap, mixes of font faces and colors, line numbers and scrolling. The buffer that is displayed in the widget is managed by the [FL_Text_Buffer](#) class. A single Text Buffer can be displayed by multiple Text Displays.

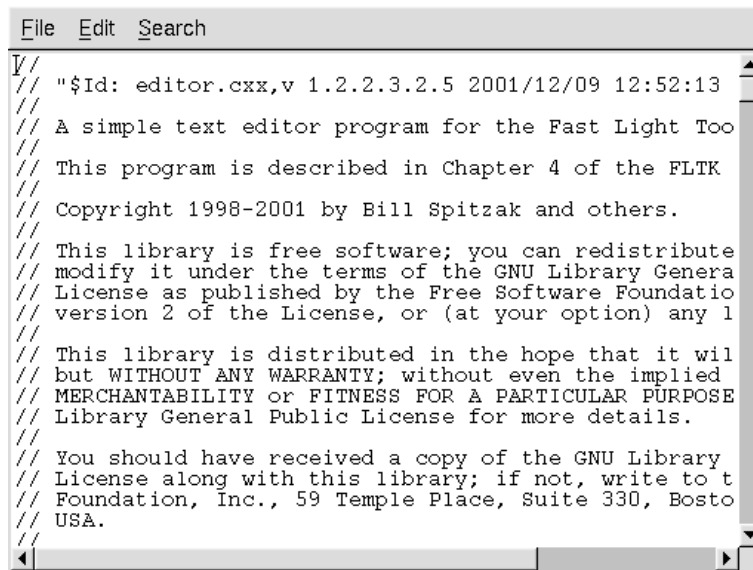


Figure 11.58 Fl_Text_Display widget

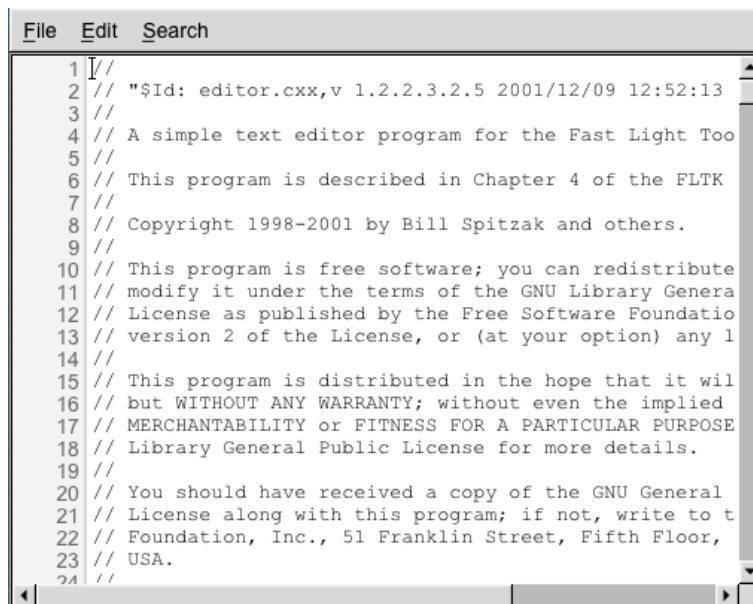


Figure 11.59 Fl_Text_Display widget with line numbers enabled

Example Use

```
#include <FL/Fl_Text_Display.H>
..
int main() {
    ..
    Fl_Text_Buffer *buff = new Fl_Text_Buffer();
    Fl_Text_Display *disp = new Fl_Text_Display(10, 10, 640, 480);
    disp->buffer(buff);           // attach text buffer to display widget
    buff->text("line one\nline two"); // add some text to buffer
    ..
}
```

Features

- Word wrap: [wrap_mode\(\)](#), [wrapped_column\(\)](#), [wrapped_row\(\)](#)
- Font control: [textfont\(\)](#), [textsize\(\)](#), [textcolor\(\)](#)

- Font styling: [highlight_data\(\)](#)
- Cursor: [cursor_style\(\)](#), [show_cursor\(\)](#), [hide_cursor\(\)](#), [cursor_color\(\)](#)
- Line numbers: [linenumber_width\(\)](#), [linenumber_font\(\)](#), [linenumber_size\(\)](#), [linenumber_fgcolor\(\)](#), [linenumber_bgcolor\(\)](#), [linenumber_align\(\)](#), [linenumber_format\(\)](#)

Note that other features may be available via [Fl_Text_Editor](#) and [Fl_Text_Buffer](#) classes.

Note

Line numbers were added in FLTK 1.3.3.

See also

[Fl_Widget::shortcut_label\(int\)](#)

11.144.2 Member Enumeration Documentation

11.144.2.1 anonymous enum

anonymous enum

text display cursor shapes enumeration

Enumerator

NORMAL_CURSOR	I-beam.
CARET_CURSOR	caret under the text
DIM_CURSOR	dim I-beam
BLOCK_CURSOR	unfill box under the current character
HEAVY_CURSOR	thick I-beam
SIMPLE_CURSOR	as cursor as Fl_Input cursor

11.144.2.2 anonymous enum

anonymous enum

wrap types - used in [wrap_mode\(\)](#)

Enumerator

WRAP_NONE	don't wrap text at all
WRAP_AT_COLUMN	wrap text at the given text column
WRAP_AT_PIXEL	wrap text at a pixel position
WRAP_AT_BOUNDS	wrap text so that it fits into the widget width

11.144.2.3 anonymous enum

anonymous enum

attribute flags in [Style_Table_Entry.attr](#)

Enumerator

ATTR_BGCOLOR	use the background color in the <code>bgcolor</code> field
ATTR_BGCOLOR_EXT_	(internal use)
ATTR_BGCOLOR_EXT	extend background color to the end of the line
ATTR_UNDERLINE	a single underline, underline types are mutually exclusive
ATTR_GRAMMAR	grammar suggestion (blue dotted underline)

Enumerator

ATTR_SPELLING	spelling suggestion (red dotted underline)
ATTR_STRIKE_THROUGH	line through the middle of the text
ATTR_LINES_MASK	the mask for all underline and strike through types

11.144.3 Constructor & Destructor Documentation**11.144.3.1 Fl_Text_Display()**

```
Fl_Text_Display::Fl_Text_Display (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new text display widget.

Parameters

<i>X,Y,W,H</i>	position and size of widget
<i>l</i>	label text, defaults to none

11.144.3.2 ~Fl_Text_Display()

```
Fl_Text_Display::~~Fl_Text_Display ()
```

Free a text display and release its associated memory.

Note

The text buffer that the text display displays is a separate entity and is not freed, nor are the style buffer or style table.

See also

[Fl_Text_Display::buffer\(Fl_Text_Buffer* buf\)](#)

11.144.4 Member Function Documentation**11.144.4.1 absolute_top_line_number()**

```
void Fl_Text_Display::absolute_top_line_number (
    int oldFirstChar) [protected]
```

Re-calculate absolute top line number for a change in scroll position.
Does nothing if the absolute top line number is not being maintained.

11.144.4.2 buffer() [1/3]

```
Fl_Text_Buffer * Fl_Text_Display::buffer () const [inline]
```

Gets the current text buffer associated with the text widget.
Multiple text widgets can be associated with the same text buffer.

Returns

current text buffer

See also

[Fl_Text_Display::buffer\(Fl_Text_Buffer* buf\)](#)

[Fl_Text_Display::buffer\(Fl_Text_Buffer& buf\)](#)

11.144.4.3 buffer() [2/3]

```
void Fl_Text_Display::buffer (
    Fl_Text_Buffer & buf) [inline]
```

Sets the current text buffer associated with the text widget.
Multiple text widgets can be associated with the same text buffer.

Parameters

<i>buf</i>	new text buffer

See also

[Fl_Text_Display::buffer\(Fl_Text_Buffer* buf\)](#)

11.144.4.4 buffer() [3/3]

```
void Fl_Text_Display::buffer (
    Fl_Text_Buffer * buf)
```

Attach a text buffer to display, replacing the current buffer (if any).
Multiple text widgets can be associated with the same text buffer.

Note

The caller is responsible for the old (replaced) buffer (if any). This method does not delete the old buffer.

Parameters

<i>buf</i>	attach this text buffer

11.144.4.5 buffer_modified_cb()

```
void Fl_Text_Display::buffer_modified_cb (
    int pos,
    int nInserted,
    int nDeleted,
    int nRestyled,
    const char * deletedText,
    void * cbArg) [static], [protected]
```

This is called whenever the buffer is modified.

Callback attached to the text buffer to receive modification information.

This callback can be used to adjust the display or update other setting. It is not advisable to change any buffers or text in this callback, or line counting may get out of sync.

Parameters

<i>pos</i>	starting index of modification
<i>nInserted</i>	number of bytes we inserted (must be UTF-8 aligned!)
<i>nDeleted</i>	number of bytes deleted (must be UTF-8 aligned!)
<i>nRestyled</i>	??
<i>deletedText</i>	this is what was removed, must not be NULL if nDeleted is set
<i>cbArg</i>	"this" pointer for static callback function

11.144.4.6 buffer_predelete_cb()

```
void Fl_Text_Display::buffer_predelete_cb (
    int pos,
    int nDeleted,
    void * cbArg) [static], [protected]
```

This is called before any characters are deleted.

Callback attached to the text buffer to receive delete information before the modifications are actually made.

This callback can be used to adjust the display or update other setting. It is not advisable to change any buffers or text in this callback, or line counting may get out of sync.

Parameters

<i>pos</i>	starting index of deletion
<i>nDeleted</i>	number of bytes we will delete (must be UTF-8 aligned!)
<i>cbArg</i>	"this" pointer for static callback function

11.144.4.7 calc_last_char()

```
void Fl_Text_Display::calc_last_char () [protected]
```

Update last display character index.

Given a [Fl_Text_Display](#) with a complete, up-to-date lineStarts array, update the lastChar entry to point to the last buffer position displayed.

11.144.4.8 calc_line_starts()

```
void Fl_Text_Display::calc_line_starts (
    int startLine,
    int endLine) [protected]
```

Update the line starts array.

Scan through the text in the Text Display's buffer and recalculate the line starts array values beginning at index "startLine" and continuing through (including) "endLine". It assumes that the line starts entry preceding "startLine" (or mFirstChar if startLine is 0) is good, and re-counts newlines to fill in the requested entries. Out of range values for "startLine" and "endLine" are acceptable.

Parameters

<i>startLine,endLine</i>	range of lines to scan as line numbers

11.144.4.9 clear_rect()

```
void Fl_Text_Display::clear_rect (
    int style,
    int X,
    int Y,
    int width,
    int height) const [protected]
```

Clear a rectangle with the appropriate background color for *style*.

Parameters

<i>style</i>	index into style table
<i>X,Y,width,height</i>	size and position of background area

11.144.4.10 col_to_x()

```
double Fl_Text_Display::col_to_x (
    double col) const
```

Convert a column number into an x pixel position.

Parameters

<i>col</i>	an approximate column number based on the main font

Returns

number of pixels from the left margin to the left of an average sized character

See also

[x_to_col\(\)](#)

11.144.4.11 count_lines()

```
int Fl_Text_Display::count_lines (
    int startPos,
    int endPos,
    bool startPosIsLineStart) const
```

Count the number of lines between two positions.

Same as [Fl_Text_Buffer::count_lines\(\)](#), but takes into account wrapping if wrapping is turned on. If the caller knows that *startPos* is at a line start, it can pass *startPosIsLineStart* as *True* to make the call more efficient by avoiding the additional step of scanning back to the last newline.

Parameters

<i>startPos</i>	index to first character
<i>endPos</i>	index after last character
<i>startPosIsLineStart</i>	avoid scanning back to the line start

Returns

number of lines

11.144.4.12 cursor_color() [1/2]

```
Fl_Color Fl_Text_Display::cursor_color () const [inline]
```

Gets the text cursor color.

Returns

cursor color

11.144.4.13 cursor_color() [2/2]

```
void Fl_Text_Display::cursor_color (
    Fl_Color n) [inline]
```

Sets the text cursor color.

Parameters

<i>n</i>	new cursor color

11.144.4.14 cursor_style()

```
void Fl_Text_Display::cursor_style (
    int style)
```

Sets the text cursor style.

Sets the text cursor style to one of the following:

- [Fl_Text_Display::NORMAL_CURSOR](#) - Shows an I beam.
- [Fl_Text_Display::CARET_CURSOR](#) - Shows a caret under the text.
- [Fl_Text_Display::DIM_CURSOR](#) - Shows a dimmed I beam.
- [Fl_Text_Display::BLOCK_CURSOR](#) - Shows an unfilled box around the current character.
- [Fl_Text_Display::HEAVY_CURSOR](#) - Shows a thick I beam.

This call also switches the cursor on and may trigger a redraw.

Parameters

<i>style</i>	new cursor style

11.144.4.15 display_insert()

```
void Fl_Text_Display::display_insert () [protected]
```

Scroll the display to bring insertion cursor into view.

Note: it would be nice to be able to do this without counting lines twice ([scroll_\(\)](#) counts them too) and/or to count from the most efficient starting point, but the efficiency of this routine is not as important to the overall performance of the text display.

11.144.4.16 display_needs_recalc()

```
void Fl_Text_Display::display_needs_recalc () [virtual]
```

Schedule a [recalc_display\(\)](#) to be done on next [draw\(\)](#).

Call this from methods that might be called repeatedly, to defers potentially CPU intensive [recalc_display\(\)](#) until it's actually needed just before [draw\(\)](#).

11.144.4.17 draw()

```
void Fl_Text_Display::draw (
    void ) [protected], [virtual]
```

Draw the widget.

This function tries to limit drawing to smaller areas if possible.

Reimplemented from [Fl_Group](#).

11.144.4.18 draw_cursor()

```
void Fl_Text_Display::draw_cursor (
    int X,
    int Y) [protected]
```

Draw a cursor with top center at X, Y.

Parameters

<i>X,Y</i>	cursor position in pixels

11.144.4.19 draw_line_numbers()

```
void Fl_Text_Display::draw_line_numbers (
    bool clearAll) [protected]
```

Refresh the line number area.

Parameters

<i>clearAll</i>	– (currently unused) If False, only draws the line number text, does not clear the area behind it. If True, clears the area and redraws the text. Use False to avoid a 'flash' for single buffered windows.

11.144.4.20 draw_range()

```
void Fl_Text_Display::draw_range (
    int startpos,
    int endpos) [protected]
```

Draw a range of text.

Refresh all of the text between buffer positions *startpos* and *endpos* not including the character at the position *endpos*.

If *endpos* points beyond the end of the buffer, refresh the whole display after *startpos*, including blank lines which are not technically part of any range of characters.

Parameters

<i>startpos</i>	index of first character to draw
<i>endpos</i>	index after last character to draw

11.144.4.21 draw_string()

```
void Fl_Text_Display::draw_string (
    int style,
    int X,
    int Y,
    int toX,
    const char * string,
    int nChars) const [protected]
```

Draw a text segment in a single style.

Draw a string or blank area according to parameter *style*, using the appropriate colors and drawing method for that style, with top left corner at X, Y. If *style* says to draw text, use *string* as source of characters, and draw *nChars*, if *style* is FILL, erase rectangle where text would have drawn from X to *toX* and from Y to the maximum y extent of the current font(s).

Parameters

<i>style</i>	index into style lookup table
<i>X,Y</i>	drawing origin
<i>toX</i>	rightmost position if this is a fill operation
<i>string</i>	text if this is a drawing operation
<i>nChars</i>	number of characters to draw

11.144.4.22 draw_text()

```
void Fl_Text_Display::draw_text (
    int left,
    int top,
    int width,
    int height) [protected]
```

Refresh a rectangle of the text display.

Parameters

<i>left,top</i>	are in coordinates of the text drawing window.
<i>width,height</i>	size in pixels

11.144.4.23 draw_vline()

```
void Fl_Text_Display::draw_vline (
    int visLineNum,
    int leftClip,
    int rightClip,
    int leftCharIndex,
    int rightCharIndex) [protected]
```

Draw a single line of text.

Draw the text on a single line represented by `visLineNum` (the number of lines down from the top of the display), limited by `leftClip` and `rightClip` window coordinates and `leftCharIndex` and `rightCharIndex` character positions (not including the character at position `rightCharIndex`).

Parameters

<i>visLineNum</i>	index of line in the visible line number lookup
<i>leftClip,rightClip</i>	pixel position of clipped area
<i>leftCharIndex,rightCharIndex</i>	index into line of segment that we want to draw

11.144.4.24 empty_vlines()

```
int Fl_Text_Display::empty_vlines () const [protected]
```

Return true if there are lines visible with no corresponding buffer text.

Returns

1 if there are empty lines

11.144.4.25 extend_range_for_styles()

```
void Fl_Text_Display::extend_range_for_styles (
    int * startpos,
    int * endpos) [protected]
```

I don't know what this does!

Extend the range of a redraw request (from `*start` to `*end`) with additional redraw requests resulting from changes to the attached style buffer (which contains auxiliary information for coloring or styling text).

Parameters

<i>startpos</i>	??
<i>endpos</i>	??

Todo Unicode?

11.144.4.26 find_line_end()

```
void Fl_Text_Display::find_line_end (
    int startPos,
    bool startPosIsLineStart,
    int * lineEnd,
    int * nextLineStart) const [protected]
```

Finds both the end of the current line and the start of the next line.

Why? In continuous wrap mode, if you need to know both, figuring out one from the other can be expensive or error prone. The problem comes when there's a trailing space or tab just before the end of the buffer. To translate an end of line value to or from the next lines start value, you need to know whether the trailing space or tab is being used as a line break or just a normal character, and to find that out would otherwise require counting all the way back to the beginning of the line.

Parameters

	<i>startPos</i>	
	<i>startPosIsLineStart</i>	
out	<i>lineEnd</i>	
out	<i>nextLineStart</i>	

11.144.4.27 find_wrap_range()

```
void Fl_Text_Display::find_wrap_range (
    const char * deletedText,
    int pos,
    int nInserted,
    int nDeleted,
    int * modRangeStart,
    int * modRangeEnd,
    int * linesInserted,
    int * linesDeleted) [protected]
```

Wrapping calculations.

When continuous wrap is on, and the user inserts or deletes characters, wrapping can happen before and beyond the changed position. This routine finds the extent of the changes, and counts the deleted and inserted lines over that range. It also attempts to minimize the size of the range to what has to be counted and re-displayed, so the results can be useful both for delimiting where the line starts need to be recalculated, and for deciding what part of the text to redisplay.

Parameters

<i>deletedText</i>	
<i>pos</i>	
<i>nInserted</i>	
<i>nDeleted</i>	
<i>modRangeStart</i>	
<i>modRangeEnd</i>	
<i>linesInserted</i>	
<i>linesDeleted</i>	

11.144.4.28 find_x()

```
int Fl_Text_Display::find_x (
    const char * s,
    int len,
```

```
int style,
int x) const [protected]
```

Find the index of the character that lies at the given x position / closest cursor position.

Parameters

<i>s</i>	UTF-8 text string
<i>len</i>	length of string
<i>style</i>	index into style lookup table
<i>x</i>	position in pixels - negative returns closest cursor position

Returns

index into buffer

11.144.4.29 `get_absolute_top_line_number()`

```
int Fl_Text_Display::get_absolute_top_line_number () const
```

Returns the absolute (non-wrapped) line number of the first line displayed.

Returns 0 if the absolute top line number is not being maintained.

11.144.4.30 `grammar_underline_color()` [1/2]

```
Fl_Color Fl_Text_Display::grammar_underline_color () const [inline]
```

Gets the underline color for style attribute ATTR_GRAMMAR.

Returns

underline color

11.144.4.31 `grammar_underline_color()` [2/2]

```
void Fl_Text_Display::grammar_underline_color (
```

```
    Fl_Color color) [inline]
```

Sets the underline color for style attribute ATTR_GRAMMAR.

Parameters

<i>color</i>	underline color

11.144.4.32 `handle()`

```
int Fl_Text_Display::handle (
    int e) [virtual]
```

Event handling.

Reimplemented from [Fl_Group](#).

Reimplemented in [Fl_Text_Editor](#).

11.144.4.33 `handle_rmb()`

```
int Fl_Text_Display::handle_rmb (
    int readonly) [protected]
```

Handle right mouse button down events.

Returns

0 for no op, 1 to cut, 2 to copy, 3 to paste

11.144.4.34 handle_vline()

```
int Fl_Text_Display::handle_vline (
    int mode,
    int lineStartPos,
    int lineLen,
    int leftChar,
    int rightChar,
    int Y,
    int bottomClip,
    int leftClip,
    int rightClip) const [protected]
```

Universal pixel machine.

We use a single function that handles all line layout, measuring, and drawing

- draw a text range
- return the width of a text range in pixels
- return the index of a character that is at a pixel position

Parameters

in	<i>mode</i>	DRAW_LINE, GET_WIDTH, FIND_INDEX, FIND_INDEX_FROM_ZERO, or FIND_CURSOR_INDEX
in	<i>lineStartPos</i>	index of first character
in	<i>lineLen</i>	size of string in bytes
in	<i>leftChar, rightChar</i>	
in	<i>Y</i>	drawing position
in	<i>bottomClip, leftClip, rightClip</i>	stop work when we reach the clipped area. rightClip is the X position that we search in FIND_INDEX.

Return values

<i>DRAW_LINE</i>	index of last drawn character
<i>GET_WIDTH</i>	width in pixels of text segment if we would draw it
<i>FIND_INDEX</i>	index of character at given x position in window coordinates
<i>FIND_INDEX_FROM_ZERO</i>	index of character at given x position without scrolling and widget offsets

Todo we need to handle hidden hyphens and tabs here!

we handle all styles and selections

we must provide code to get pixel positions of the middle of a character as well

11.144.4.35 highlight_data()

```
void Fl_Text_Display::highlight_data (
    Fl_Text_Buffer * styleBuffer,
    const Style_Table_Entry * styleTable,
    int nStyles,
    char unfinishedStyle,
    Unfinished_Style_Cb unfinishedHighlightCB,
    void * cbArg)
```

Attach (or remove) highlight information in text display and redisplay.

Highlighting information consists of a style buffer which parallels the normal text buffer, but codes font and color information for the display; a style table which translates style buffer codes (indexed by buffer character - 'A') into fonts

and colors; and a callback mechanism for as-needed highlighting, triggered by a style buffer entry of "unfinished↵ Style". Style buffer can trigger additional redisplay during a normal buffer modification if the buffer contains a primary [Fl_Text_Selection](#) (see [extend_range_for_styles\(\)](#) for more information on this protocol).

Style buffers, tables and their associated memory are managed by the caller.

Styles are ranged from 65 ('A') to 126.

Note

Style information in the style buffer must have the same byte offset as the corresponding character in the text buffer. UTF-8 characters can have a maximum length of four bytes. Style information must take this into account and fill the unused bytes with 0. See [fl_utf8len\(\)](#).

Text: "`*g* r ü *n*`", where normal style is 'A', and bold is 'B'

Text Buffer(hex): 67 72 c3 bc 6e : gr..n

Style Buffer(hex): 42 41 41 00 42 : BAA.B

Parameters

<i>styleBuffer</i>	this buffer works in parallel to the text buffer. For every character in the text buffer, the style buffer has a byte at the same offset that contains an index into an array of possible styles.
<i>styleTable</i>	a list of styles indexed by the style buffer
<i>nStyles</i>	number of styles in the style table
<i>unfinishedStyle</i>	if this style is found, the callback below is called
<i>unfinishedHighlightCB</i>	if a character with an unfinished style is found, this callback will be called
<i>cbArg</i>	an optional argument for the callback above, usually a pointer to the Text Display.

See also

[Fl_Text_Display::style_buffer\(\)](#)

11.144.4.36 in_selection()

```
int Fl_Text_Display::in_selection (
    int X,
    int Y) const
```

Check if a pixel position is within the primary selection.

Parameters

<i>X,Y</i>	pixel position to test

Returns

1 if position (X, Y) is inside of the primary [Fl_Text_Selection](#)

11.144.4.37 insert()

```
void Fl_Text_Display::insert (
    const char * text)
```

Inserts "text" at the current cursor location.

This has the same effect as inserting the text into the buffer using `insert(insert_position(),text)` and then moving the insert position after the newly inserted text, except that it's optimized to do less redrawing.

Parameters

<i>text</i>	new text in UTF-8 encoding.

11.144.4.38 insert_position() [1/2]

```
int Fl_Text_Display::insert_position () const [inline]
```

Gets the position of the text insertion cursor for text display.

The insert position is the byte count (offset) from the beginning of the text buffer (starting with 0). Returns 0 (zero) if no buffer is associated to the text display. Returns `buffer()->length()` if the insert position is at the end of the buffer.

Returns

insert position index into text buffer

See also

[insert_position\(int\)](#)

11.144.4.39 insert_position() [2/2]

```
void Fl_Text_Display::insert_position (
    int newPos)
```

Sets the position of the text insertion cursor for text display.

Moves the insertion cursor in front of the character at `newPos`. This function may trigger a redraw.

Parameters

<i>newPos</i>	new caret position

11.144.4.40 line_end()

```
int Fl_Text_Display::line_end (
    int startPos,
    bool startPosIsLineStart) const
```

Returns the end of a line.

Same as `buffer()->line_end(startPos)`, but takes into account line breaks when wrapping is turned on. If the caller knows that `startPos` is at a line start, it can pass `startPosIsLineStart` as `True` to make the call more efficient by avoiding the additional step of scanning back to the last newline.

Note that the definition of the end of a line is less clear when continuous wrap is on. With continuous wrap off, it's just a pointer to the newline that ends the line. When it's on, it's the character beyond the last **displayable** character on the line, where a whitespace character which has been "converted" to a newline for wrapping is not considered displayable. Also note that a line can be wrapped at a non-whitespace character if the line had no whitespace. In this case, this routine returns a pointer to the start of the next line. This is also consistent with the model used by `visLineLength`.

Parameters

<i>startPos</i>	index to starting character
<i>startPosIsLineStart</i>	avoid scanning back to the line start

Returns

new position as index

11.144.4.41 line_start()

```
int Fl_Text_Display::line_start (
    int pos) const
```

Return the beginning of a line.

Same as `buffer()->line_start(pos)`, but returns the character after last wrap point rather than the last newline.

Parameters

<i>pos</i>	index to starting character

Returns

new position as index

11.144.4.42 `linenumber_align()`

```
void Fl_Text_Display::linenumber_align (
    Fl_Align val)
```

Set alignment for line numbers (if enabled).

Valid values are FL_ALIGN_LEFT, FL_ALIGN_CENTER or FL_ALIGN_RIGHT.

Version

1.3.3

11.144.4.43 `linenumber_bgcolor()`

```
void Fl_Text_Display::linenumber_bgcolor (
    Fl_Color val)
```

Set the background color used for line numbers (if enabled).

Version

1.3.3

11.144.4.44 `linenumber_fgcolor()`

```
void Fl_Text_Display::linenumber_fgcolor (
    Fl_Color val)
```

Set the foreground color used for line numbers (if enabled).

Version

1.3.3

11.144.4.45 `linenumber_font()`

```
void Fl_Text_Display::linenumber_font (
    Fl_Font val)
```

Set the font used for line numbers (if enabled).

Version

1.3.3

11.144.4.46 `linenumber_format()`

```
void Fl_Text_Display::linenumber_format (
    const char * val)
```

Sets the printf() style format string used for line numbers.

Default is "%d" for normal unpadded decimal integers.

An internal copy of *val* is allocated and managed; it is automatically freed whenever a new value is assigned, or when the widget is destroyed.

The value of *val* must *not* be NULL.

Example values:


```
- "%d"    -- For normal line numbers without padding (Default)
- "%03d"  -- For 000 padding
- "%x"    -- For hexadecimal line numbers
- "%o"    -- For octal line numbers
```

Version

1.3.3

11.144.4.47 linenumber_size()

```
void Fl_Text_Display::linenumber_size (
    Fl_Fontsize val)
```

Set the font size used for line numbers (if enabled).

Version

1.3.3

11.144.4.48 linenumber_width()

```
void Fl_Text_Display::linenumber_width (
    int width)
```

Set width of screen area for line numbers.

Use to also enable/disable line numbers. A value of 0 disables line numbering, values >0 enable the line number display.

Parameters

<i>width</i>	The new width of the area for line numbers to appear, in pixels. 0 disables line numbers (default)

11.144.4.49 longest_vline()

```
int Fl_Text_Display::longest_vline () const [protected]
```

Find the longest line of all visible lines.

Returns

the width of the longest visible line in pixels

11.144.4.50 maintain_absolute_top_line_number()

```
void Fl_Text_Display::maintain_absolute_top_line_number (
    int state) [protected]
```

Line numbering stuff, currently unused.

In continuous wrap mode, internal line numbers are calculated after wrapping. A separate non-wrapped line count is maintained when line numbering is turned on. There is some performance cost to maintaining this line count, so normally absolute line numbers are not tracked if line numbering is off. This routine allows callers to specify that they still want this line count maintained (for use via [Fl_Text_Display::position_to_linecol\(\)](#)). More specifically, this allows the line number reported in the statistics line to be calibrated in absolute lines, rather than post-wrapped lines.

11.144.4.51 maintaining_absolute_top_line_number()

```
int Fl_Text_Display::maintaining_absolute_top_line_number () const [protected]
```

Returns true if a separate absolute top line number is being maintained.

The absolute top line number is used for displaying line numbers in continuous wrap mode or showing in the statistics line (the latter is currently not available in FLTK).

11.144.4.52 measure_deleted_lines()

```
void Fl_Text_Display::measure_deleted_lines (
    int pos,
    int nDeleted) [protected]
```

Wrapping calculations.

This is a stripped-down version of the findWrapRange() function above, intended to be used to calculate the number of "deleted" lines during a buffer modification. It is called *before* the modification takes place.

This function should only be called in continuous wrap mode with a non-fixed font width. In that case, it is impossible to calculate the number of deleted lines, because the necessary style information is no longer available *after* the modification. In other cases, we can still perform the calculation afterwards (possibly even more efficiently).

Parameters

<i>pos</i>	
<i>nDeleted</i>	

11.144.4.53 measure_proportional_character()

```
double Fl_Text_Display::measure_proportional_character (
    const char * s,
    int xPix,
    int pos) const [protected]
```

Wrapping calculations.

Measure the width in pixels of the first character of string "s" at a particular column "colNum" and buffer position "pos". This is for measuring characters in proportional or mixed-width highlighting fonts.

A note about proportional and mixed-width fonts: the mixed width and proportional font code in nedit does not get much use in general editing, because nedit doesn't allow per-language-mode fonts, and editing programs in a proportional font is usually a bad idea, so very few users would choose a proportional font as a default. There are still probably mixed-width syntax highlighting cases where things don't redraw properly for insertion/deletion, though static display and wrapping and resizing should now be solid because they are now used for online help display.

Parameters

<i>s</i>	text string
<i>xPix</i>	x pixel position needed for calculating tab widths
<i>pos</i>	offset within string

Returns

width of character in pixels

11.144.4.54 measure_vline()

```
int Fl_Text_Display::measure_vline (
    int visLineNum) const [protected]
```

Returns the width in pixels of the displayed line pointed to by "visLineNum".

Parameters

<i>visLineNum</i>	index into visible lines array

Returns

width of line in pixels

11.144.4.55 move_down()

```
int Fl_Text_Display::move_down ()
```

Moves the current insert position down one line.

Returns

1 if the cursor moved, 0 if the beginning of the text was reached

11.144.4.56 move_left()

```
int Fl_Text_Display::move_left ()
```

Moves the current insert position left one character.

Returns

1 if the cursor moved, 0 if the beginning of the text was reached

11.144.4.57 move_right()

```
int Fl_Text_Display::move_right ()
```

Moves the current insert position right one character.

Returns

1 if the cursor moved, 0 if the end of the text was reached

11.144.4.58 move_up()

```
int Fl_Text_Display::move_up ()
```

Moves the current insert position up one line.

Returns

1 if the cursor moved, 0 if the beginning of the text was reached

11.144.4.59 offset_line_starts()

```
void Fl_Text_Display::offset_line_starts (
    int newTopLineNum) [protected]
```

Offset line start counters for a new vertical scroll position.

Offset the line starts array, mTopLineNum, mFirstChar and lastChar, for a new vertical scroll position given by new↔TopLineNum. If any currently displayed lines will still be visible, salvage the line starts values, otherwise, count lines from the nearest known line start (start or end of buffer, or the closest value in the mLineStarts array)

Parameters

<i>newTopLineNum</i>	index into buffer

11.144.4.60 overstrike()

```
void Fl_Text_Display::overstrike (
    const char * text)
```

Replaces text at the current insert position.

Parameters

<i>text</i>	new text in UTF-8 encoding

Todo Unicode? Find out exactly what we do here and simplify.

11.144.4.61 position_style()

```
int Fl_Text_Display::position_style (
    int lineStartPos,
    int lineLen,
    int lineIndex) const
```

Find the correct style for a character.

Determine the drawing method to use to draw a specific character from "buf".

`lineStartPos` gives the character index where the line begins, `lineIndex`, the number of characters past the beginning of the line, and `lineLen` the number of displayed characters past the beginning of the line. Passing `lineStartPos` of -1 returns the drawing style for "no text".

Why not just: `position_style(pos)`? Because style applies to blank areas of the window beyond the text boundaries, and because this routine must also decide whether a position is inside of a rectangular [Fl_Text_Selection](#), and do so efficiently, without re-counting character positions from the start of the line.

Note that style is a somewhat incorrect name, drawing method would be more appropriate.

If `lineIndex` is pointing to the last character in a line, and the second to last character has the `ATTR_BGCOLOR_EXT` set, the background color will extend into the remaining line.

Parameters

<i>lineStartPos</i>	beginning of this line
<i>lineLen</i>	number of bytes in line
<i>lineIndex</i>	position of character within line

Returns

style for the given character

11.144.4.62 position_to_line()

```
int Fl_Text_Display::position_to_line (
    int pos,
    int * lineNum) const [protected]
```

Convert a position index into a line number offset.

Find the line number of position `pos` relative to the first line of displayed text, counting from 0 to *visible lines* - 1.

The line number is returned in `lineNum`.

Returns 0 if the line is not displayed. In this case `lineNum` is 0 as well.

Returns 1 if the line is displayed. In this case `lineNum` is the relative line number.

Parameters

in	<i>pos</i>	byte position in buffer
out	<i>lineNum</i>	relative line number of byte <code>pos</code> in buffer

Returns

whether the character at byte position `pos` is currently displayed

Return values

0	<code>pos</code> is not displayed; <code>lineNum</code> is invalid (zero)
1	<code>pos</code> is displayed; <code>lineNum</code> is valid

11.144.4.63 position_to_linecol()

```
int Fl_Text_Display::position_to_linecol (
    int pos,
    int * lineNum,
    int * column) const [protected]
```

Find the line and column number of position `pos`.

This only works for displayed lines. If the line is not displayed, the function returns 0 (without the `mLineStarts` array it could turn in to very long calculation involving scanning large amounts of text in the buffer). If continuous wrap mode is on, returns the absolute line number (as opposed to the wrapped line number which is used for scrolling).

Parameters

	<i>pos</i>	character index
out	<i>lineNum</i>	absolute (unwrapped) line number
out	<i>column</i>	character offset to the beginning of the line

Returns

0 if `pos` is off screen, line number otherwise

Todo a column number makes little sense in the UTF-8/variable font width environment. We will have to further define what exactly we want to return. Please check the functions that call this particular function.

11.144.4.64 position_to_xy()

```
int Fl_Text_Display::position_to_xy (
    int pos,
    int * X,
    int * Y) const
```

Convert a character index into a pixel position.

Translate a buffer text position to the XY location where the top left of the cursor would be positioned to point to that character. Returns 0 if the position is not displayed because it is **vertically out** of view. If the position is horizontally out of view, returns the X coordinate where the position would be if it were visible.

Parameters

	<i>pos</i>	character index
out	<i>X, Y</i>	pixel position of character on screen

Returns

0 if character vertically out of view, X & Y positions otherwise

11.144.4.65 recalc_display()

```
void Fl_Text_Display::recalc_display () [virtual]
```

Recalculate the display's visible lines and scrollbar sizes.

Beware calling this directly may cause a lot of CPU if called repeatedly (issue 300). Better to call [display_needs_recalc\(\)](#) to flag a recalc to be done during next [draw\(\)](#).

11.144.4.66 redisplay_range()

```
void Fl_Text_Display::redisplay_range (
    int startpos,
    int endpos)
```

Marks text from start to end as needing a redraw.

This function will trigger a damage event and later a redraw of parts of the widget.

Parameters

<i>startpos</i>	index of first character needing redraw
<i>endpos</i>	index after last character needing redraw

11.144.4.67 reset_absolute_top_line_number()

```
void Fl_Text_Display::reset_absolute_top_line_number () [protected]
```

Reestablish the absolute (non-wrapped) top line number.

Count lines from the beginning of the buffer to reestablish the absolute (non-wrapped) top line number. If mode is not continuous wrap, or the number is not being maintained, does nothing.

11.144.4.68 resize()

```
void Fl_Text_Display::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Change the size of the displayed text area.

Calling this function will trigger a recalculation of all visible lines and of all scrollbar sizes.

Parameters

<i>X,Y,W,H</i>	new position and size of this widget

Reimplemented from [Fl_Group](#).

11.144.4.69 rewind_lines()

```
int Fl_Text_Display::rewind_lines (
    int startPos,
    int nLines)
```

Skip a number of lines back.

Same as `buffer()->rewind_lines(startPos, nLines)`, but takes into account line breaks when wrapping is turned on.

Parameters

<i>startPos</i>	index to starting character
<i>nLines</i>	number of lines to skip back

Returns

new position as index

11.144.4.70 scroll()

```
void Fl_Text_Display::scroll (
    int topLineNum,
    int horizOffset)
```

Scrolls the current buffer to start at the specified line and column.

Parameters

<i>topLineNum</i>	top line number
<i>horizOffset</i>	column number

Todo Column numbers make little sense here.

11.144.4.71 scroll_()

```
int Fl_Text_Display::scroll_ (
    int topLineNum,
    int horizOffset) [protected]
```

Scrolls the current buffer to start at the specified line and column.

Parameters

<i>topLineNum</i>	top line number
<i>horizOffset</i>	in pixels

Returns

0 if nothing changed, 1 if we scrolled

11.144.4.72 scroll_timer_cb()

```
void Fl_Text_Display::scroll_timer_cb (
    void * user_data) [static], [protected]
```

Timer callback for scroll events.

This timer event scrolls the text view proportionally to how far the mouse pointer has left the text area. This allows for smooth scrolling without "wiggeling" the mouse.

11.144.4.73 scrollbar_align() [1/2]

```
Fl_Align Fl_Text_Display::scrollbar_align () const [inline]
```

Gets the scrollbar alignment type.

Returns

scrollbar alignment

11.144.4.74 scrollbar_align() [2/2]

```
void Fl_Text_Display::scrollbar_align (
    Fl_Align a) [inline]
```

Sets the scrollbar alignment type.

Parameters

<i>a</i>	new scrollbar alignment

11.144.4.75 scrollbar_size() [1/2]

```
int Fl_Text_Display::scrollbar_size () const [inline]
```

Gets the current size of the scrollbars' troughs, in pixels.

If this value is zero (default), this widget will use the [Fl::scrollbar_size\(\)](#) value as the scrollbar's width.

Returns

Scrollbar size in pixels, or 0 if the global [Fl::scrollbar_size\(\)](#) is being used.

See also

[Fl::scrollbar_size\(int\)](#)

11.144.4.76 scrollbar_size() [2/2]

```
void Fl_Text_Display::scrollbar_size (
    int newSize) [inline]
```

Sets the pixel size of the scrollbars' troughs to `newSize`, in pixels.

Normally you should not need this method, and should use [Fl::scrollbar_size\(int\)](#) instead to manage the size of ALL your widgets' scrollbars. This ensures your application has a consistent UI, is the default behavior, and is normally what you want.

Only use THIS method if you really need to override the global scrollbar size. The need for this should be rare.

Setting `newSize` to the special value of 0 causes the widget to track the global [Fl::scrollbar_size\(\)](#), which is the default.

Parameters

<code>in</code>	<code>newSize</code>	Sets the scrollbar size in pixels. If 0 (default), scrollbar size tracks the global Fl::scrollbar_size()

See also

[Fl::scrollbar_size\(\)](#)

11.144.4.77 scrollbar_width() [1/2]

```
int Fl_Text_Display::scrollbar_width () const [inline]
```

Returns the global value [Fl::scrollbar_size\(\)](#) unless a specific `scrollbar_width_` has been set.

Deprecated Use [scrollbar_size\(\)](#) instead.

Todo This method should eventually be removed.

11.144.4.78 scrollbar_width() [2/2]

```
void Fl_Text_Display::scrollbar_width (
    int width) [inline]
```

Sets the global [Fl::scrollbar_size\(\)](#), and forces this instance of the widget to use it.

Deprecated Use [scrollbar_size\(\)](#) instead.

Todo This method should eventually be removed

11.144.4.79 secondary_selection_color() [1/2]

```
Fl_Color Fl_Text_Display::secondary_selection_color () const [inline]
```

Gets the background color for the secondary selection block.

Returns

background color color

11.144.4.80 secondary_selection_color() [2/2]

```
void Fl_Text_Display::secondary_selection_color (
    Fl_Color color) [inline]
```

Sets the background color for the secondary selection block.

Parameters

<code>color</code>	background color

11.144.4.81 shortcut() [1/2]

```
int Fl_Text_Display::shortcut () const [inline]
```

Todo FIXME : get set methods pointing on shortcut_ have no effects as shortcut_ is unused in this class and derived!

Returns

the current shortcut key

11.144.4.82 shortcut() [2/2]

```
void Fl_Text_Display::shortcut (
    int s) [inline]
```

Todo FIXME : get set methods pointing on shortcut_ have no effects as shortcut_ is unused in this class and derived!

Parameters

<i>s</i>	the new shortcut key

11.144.4.83 show_cursor()

```
void Fl_Text_Display::show_cursor (
    int b = 1)
```

Shows the text cursor.

This function may trigger a redraw.

Parameters

<i>b</i>	show(1) or hide(0) the text cursor (caret).

11.144.4.84 show_insert_position()

```
void Fl_Text_Display::show_insert_position ()
```

Scrolls the text buffer to show the current insert position.

This function triggers a complete recalculation, ending in a call to [Fl_Text_Display::display_insert\(\)](#)

11.144.4.85 skip_lines()

```
int Fl_Text_Display::skip_lines (
    int startPos,
    int nLines,
    bool startPosIsLineStart)
```

Skip a number of lines forward.

Same as `Fl_Text_Buffer::skip_lines(startPos, nLines)`, but takes into account line breaks when wrapping is turned on. If the caller knows that `startPos` is at a line start, it can pass `startPosIsLineStart` as `True` to make the call more efficient by avoiding the additional step of scanning back to the last newline.

Parameters

<i>startPos</i>	index to starting character
<i>nLines</i>	number of lines to skip ahead
<i>startPosIsLineStart</i>	avoid scanning back to the line start

Returns

new position as index

11.144.4.86 spelling_underline_color() [1/2]

```
Fl_Color Fl_Text_Display::spelling_underline_color () const [inline]
```

Gets the underline color for style attribute ATTR_SPELLING.

Returns

underline color

11.144.4.87 spelling_underline_color() [2/2]

```
void Fl_Text_Display::spelling_underline_color (
    Fl_Color color) [inline]
```

Sets the underline color for style attribute ATTR_SPELLING.

Parameters

<i>color</i>	underline color

11.144.4.88 string_width()

```
double Fl_Text_Display::string_width (
    const char * string,
    int length,
    int style) const [protected]
```

Find the width of a string in the font of a particular style.

Parameters

<i>string</i>	the text
<i>length</i>	number of bytes in string
<i>style</i>	index into style table

Returns

width of text segment in pixels

11.144.4.89 style_buffer()

```
Fl_Text_Buffer * Fl_Text_Display::style_buffer () const [inline]
```

Gets the current style buffer associated with the text widget.

Multiple text widgets can be associated with the same style buffer.

Returns

current style buffer

See also

[Fl_Text_Display::highlight_data\(\)](#)

11.144.4.90 textcolor() [1/2]

```
Fl_Color Fl_Text_Display::textcolor () const [inline]
```

Gets the default color of text in the widget.

Returns

text color unless overridden by a style

11.144.4.91 textcolor() [2/2]

```
void Fl_Text_Display::textcolor (
    Fl_Color n) [inline]
```

Sets the default color of text in the widget.

Parameters

<i>n</i>	new text color

11.144.4.92 textfont() [1/2]

```
Fl_Font Fl_Text_Display::textfont () const [inline]
```

Gets the default font used when drawing text in the widget.

Returns

current text font face unless overridden by a style

11.144.4.93 textfont() [2/2]

```
void Fl_Text_Display::textfont (
    Fl_Font s) [inline]
```

Sets the default font used when drawing text in the widget.

Parameters

<i>s</i>	default text font face

11.144.4.94 textsize() [1/2]

```
Fl_Fontsize Fl_Text_Display::textsize () const [inline]
```

Gets the default size of text in the widget.

Returns

current text height unless overridden by a style

11.144.4.95 textsize() [2/2]

```
void Fl_Text_Display::textsize (
    Fl_Fontsize s) [inline]
```

Sets the default size of text in the widget.

Parameters

<i>s</i>	new text size

11.144.4.96 update_h_scrollbar()

```
void Fl_Text_Display::update_h_scrollbar () [protected]
```

Update horizontal scrollbar.

Update the minimum, maximum, slider size, page increment, and value for the horizontal scrollbar.

11.144.4.97 update_line_starts()

```
void Fl_Text_Display::update_line_starts (
    int pos,
    int charsInserted,
    int charsDeleted,
    int linesInserted,
    int linesDeleted,
    int * scrolled) [protected]
```

Update line start arrays and variables.

Update the line starts array, mTopLineNum, mFirstChar and lastChar for this text display after a modification to the text buffer, given by the position *pos* where the change began, and the numbers of characters and lines inserted and deleted.

Parameters

	<i>pos</i>	index into buffer of recent changes
	<i>charsInserted</i>	number of bytes(!) inserted
	<i>charsDeleted</i>	number of bytes(!) deleted
	<i>linesInserted</i>	number of lines
	<i>linesDeleted</i>	number of lines
out	<i>scrolled</i>	set to 1 if the text display needs to be scrolled

11.144.4.98 update_v_scrollbar()

```
void Fl_Text_Display::update_v_scrollbar () [protected]
```

Update vertical scrollbar.

Update the minimum, maximum, slider size, page increment, and value for the vertical scrollbar.

11.144.4.99 vline_length()

```
int Fl_Text_Display::vline_length (
    int visLineNum) const [protected]
```

Count number of bytes in a visible line.

Return the length of a line (number of bytes) by examining entries in the line starts array rather than by scanning for newlines.

Parameters

<i>visLineNum</i>	index of line in visible line array

Returns

number of bytes in this line

11.144.4.100 word_end()

```
int Fl_Text_Display::word_end (
    int pos) const [inline]
```

Moves the insert position to the end of the current word.

Parameters

<i>pos</i>	start calculation at this index

Returns

index of first character after the end of the word

11.144.4.101 word_start()

```
int Fl_Text_Display::word_start (
    int pos) const [inline]
```

Moves the insert position to the beginning of the current word.

Parameters

<i>pos</i>	start calculation at this index

Returns

beginning of the words

11.144.4.102 wrap_mode()

```
void Fl_Text_Display::wrap_mode (
    int wrap,
    int wrapMargin)
```

Set the new text wrap mode.

If `wrap` mode is not zero, this call enables automatic word wrapping at column `wrapMargin`. Word-wrapping does not change the text buffer itself, only the way the text is displayed. Different Text Displays can have different wrap modes, even if they share the same Text Buffer.

Valid wrap modes are:

- `WRAP_NONE` : don't wrap text at all
- `WRAP_AT_COLUMN` : wrap text at the given text column
- `WRAP_AT_PIXEL` : wrap text at a pixel position
- `WRAP_AT_BOUNDS` : wrap text so that it fits into the widget width

Parameters

<i>wrap</i>	new wrap mode (see above)
<i>wrapMargin</i>	in <code>WRAP_AT_COLUMN</code> mode, text will wrap at the n'th character. For variable width fonts, an average character width is calculated. The column width is calculated using the current textfont or the first style when this function is called. If the font size changes, this function must be called again. In <code>WRAP_AT_PIXEL</code> mode, this is the pixel position.

11.144.4.103 wrap_uses_character()

```
int Fl_Text_Display::wrap_uses_character (
    int lineEndPos) const [protected]
```

Check if the line break is caused by a newline or by line wrapping.

Line breaks in continuous wrap mode usually happen at newlines (`\n`) or whitespace. This line-terminating character is not included in line width measurements and has a special status as a non-visible character. However, lines with

no whitespace are wrapped without the benefit of a line terminating character, and this distinction causes endless trouble with all of the text display code which was originally written without continuous wrap mode and always expects to wrap at a newline character.

Given the position of the end of the line, as returned by [Fl_Text_Display::line_end\(\)](#) or [Fl_Text_Buffer::line_end\(\)](#), this returns true if there is a line terminating character, and false if there's not. On the last character in the buffer, this function can't tell for certain whether a trailing space was used as a wrap point, and just guesses that it wasn't. So if an exact accounting is necessary, don't use this function.

Parameters

<i>lineEndPos</i>	index of character where the line wraps

Returns

1 if a \n character causes the line wrap

11.144.4.104 wrapped_column()

```
int Fl_Text_Display::wrapped_column (
    int row,
    int column) const
```

Nobody knows what this function does.

Correct a column number based on an unconstrained position (as returned by [TextDXYToUnconstrainedPosition](#)) to be relative to the last actual newline in the buffer before the row and column position given, rather than the last line start created by line wrapping. This is an adapter for rectangular selections and code written before continuous wrap mode, which thinks that the unconstrained column is the number of characters from the last newline. Obviously this is time consuming, because it involves character re-counting.

Parameters

<i>row</i>	
<i>column</i>	

Returns

something unknown

Todo What does this do and how is it useful? Column numbers mean little in this context. Which functions depend on this one? Function [TextDXYToUnconstrainedPosition](#) does not exist (nedit port?)

Todo Unicode?

11.144.4.105 wrapped_line_counter()

```
void Fl_Text_Display::wrapped_line_counter (
    Fl_Text_Buffer * buf,
    int startPos,
    int maxPos,
    int maxLines,
    bool startPosIsLineStart,
    int styleBufOffset,
    int * retPos,
    int * retLines,
    int * retLineStart,
    int * retLineEnd,
    bool countLastLineMissingNewLine = true) const [protected]
```

Wrapping calculations.

Count forward from startPos to either maxPos or maxLines (whichever is reached first), and return all relevant positions and line count. The provided textBuffer may differ from the actual text buffer of the widget. In that case it must be a (partial) copy of the actual text buffer and the styleBufOffset argument must indicate the starting position of the copy, to take into account the correct style information.

Parameters

in	<i>buf</i>	The text buffer to operate on
in	<i>startPos</i>	Starting index position into the buffer
in	<i>maxPos</i>	Maximum index position into the buffer we'll reach
in	<i>maxLines</i>	Maximum number of lines we'll reach
in	<i>startPosIsLineStart</i>	Flag indicating if startPos is start of line. (If set, prevents our having to find the line start)
in	<i>styleBufOffset</i>	Offset index position into style buffer.
out	<i>retPos</i>	Position where counting ended. When counting lines, the position returned is the start of the line "maxLines" lines beyond "startPos".
out	<i>retLines</i>	Number of line breaks counted
out	<i>retLineStart</i>	Start of the line where counting ended
out	<i>retLineEnd</i>	End position of the last line traversed
out	<i>countLastLineMissingNewLine</i>	

11.144.4.106 wrapped_row()

```
int Fl_Text_Display::wrapped_row (
    int row) const
```

Nobody knows what this function does.

Correct a row number from an unconstrained position (as returned by TextDXYToUnconstrainedPosition) to a straight number of newlines from the top line of the display. Because rectangular selections are based on newlines, rather than display wrapping, and anywhere a rectangular selection needs a row, it needs it in terms of un-wrapped lines.

Parameters

<i>row</i>	

Returns

something unknown

Todo What does this do and how is it useful? Column numbers mean little in this context. Which functions depend on this one? Function TextDXYToUnconstrainedPosition does not exist (nedit port?)

11.144.4.107 x_to_col()

```
double Fl_Text_Display::x_to_col (
    double x) const
```

Convert an x pixel position into a column number.

Parameters

<i>x</i>	number of pixels from the left margin

Returns

an approximate column number based on the main font

11.144.4.108 xy_to_position()

```
int Fl_Text_Display::xy_to_position (
    int X,
    int Y,
    int posType = CHARACTER_POS) const [protected]
```

Translate a pixel position into a character index.

Translate window coordinates to the nearest (insert cursor or character cell) text position. The parameter `posType` specifies how to interpret the position: `CURSOR_POS` means translate the coordinates to the nearest cursor position, and `CHARACTER_POS` means return the position of the character closest to (X, Y).

Parameters

<i>X,Y</i>	pixel position
<i>posType</i>	CURSOR_POS or CHARACTER_POS

Returns

index into text buffer

11.144.4.109 xy_to_rowcol()

```
void Fl_Text_Display::xy_to_rowcol (
    int X,
    int Y,
    int * row,
    int * column,
    int posType = CHARACTER_POS) const [protected]
```

Translate pixel coordinates into row and column.

Translate window coordinates to the nearest row and column number for positioning the cursor. This, of course, makes no sense when the font is proportional, since there are no absolute columns. The parameter `posType` specifies how to interpret the position: `CURSOR_POS` means translate the coordinates to the nearest position between characters, and `CHARACTER_POS` means translate the position to the nearest character cell.

Parameters

	<i>X,Y</i>	pixel coordinates
<i>out</i>	<i>row,column</i>	nearest row and column
	<i>posType</i>	CURSOR_POS or CHARACTER_POS

The documentation for this class was generated from the following files:

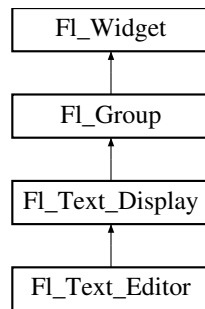
- Fl_Text_Display.H
- Fl_Text_Display.cxx

11.145 Fl_Text_Editor Class Reference

This is the FLTK text editor widget.

```
#include <Fl_Text_Editor.H>
```

Inheritance diagram for Fl_Text_Editor:



Classes

- struct [Key_Binding](#)
Simple linked list item associating a key/state to a function.

Public Types

- typedef int(*) **Key_Func**(int key, [Fl_Text_Editor](#) *editor)
Key function binding callback type.

Public Types inherited from [Fl_Text_Display](#)

- enum {
[NORMAL_CURSOR](#) , [CARET_CURSOR](#) , [DIM_CURSOR](#) , [BLOCK_CURSOR](#) ,
[HEAVY_CURSOR](#) , [SIMPLE_CURSOR](#) }
text display cursor shapes enumeration
- enum { [CURSOR_POS](#) , [CHARACTER_POS](#) }
the character position is the left edge of a character, whereas the cursor is thought to be between the centers of two consecutive characters.
- enum {
DRAG_NONE = -2 , **DRAG_START_DND** = -1 , **DRAG_CHAR** = 0 , **DRAG_WORD** = 1 ,
DRAG_LINE = 2 }
drag types - they match [Fl::event_clicks\(\)](#) so that single clicking to start a collection selects by character, double clicking selects by word and triple clicking selects by line.
- enum { [WRAP_NONE](#) , [WRAP_AT_COLUMN](#) , [WRAP_AT_PIXEL](#) , [WRAP_AT_BOUNDS](#) }
wrap types - used in [wrap_mode\(\)](#)
- enum {
[ATTR_BGCOLOR](#) = 0x0001 , [ATTR_BGCOLOR_EXT](#) = 0x0002 , [ATTR_BGCOLOR_EXT](#) = 0x0003 ,
[ATTR_UNDERLINE](#) = 0x0004 ,
[ATTR_GRAMMAR](#) = 0x0008 , [ATTR_SPELLING](#) = 0x000C , [ATTR_STRIKE_THROUGH](#) = 0x0010 ,
[ATTR_LINES_MASK](#) = 0x001C }
attribute flags in [Style_Table_Entry.attr](#)
- typedef void(*) **Unfinished_Style_Cb**(int, void *)

Public Member Functions

- void **add_default_key_bindings** ([Key_Binding](#) **list)
Adds all of the default editor key bindings to the specified key binding list.
- void **add_key_binding** (int key, int state, [Key_Func](#) f)
Adds a key of state state with the function f.
- void **add_key_binding** (int key, int state, [Key_Func](#) f, [Key_Binding](#) **list)
Adds a key of state state with the function function to an arbitrary key binding list list.
- [Key_Func](#) **bound_key_function** (int key, int state) const
Returns the function associated with a key binding.

- **Key_Func bound_key_function** (int key, int state, [Key_Binding](#) *list) const
Returns the function associated with a key binding.
- void **default_key_function** ([Key_Func](#) f)
Sets the default key function for unassigned keys.
- **Fl_Text_Editor** (int X, int Y, int W, int H, const char *l=0)
The constructor creates a new text editor widget.
- int **handle** (int e) [FL_OVERRIDE](#)
Event handling.
- int **insert_mode** ()
Gets the current insert mode; if non-zero, new text is inserted before the current cursor position.
- void **insert_mode** (int b)
Sets the current insert mode; if non-zero, new text is inserted before the current cursor position.
- void **remove_all_key_bindings** ()
Removes all of the key bindings associated with the text editor or list.
- void **remove_all_key_bindings** ([Key_Binding](#) **list)
Removes all of the key bindings associated with the text editor or list.
- void **remove_key_binding** (int key, int state)
Removes the key binding associated with the key "key" of state "state".
- void **remove_key_binding** (int key, int state, [Key_Binding](#) **list)
Removes the key binding associated with the key key of state state from the [Key_Binding](#) list list.
- int **tab_nav** () const
Check if Tab focus navigation is enabled.
- void **tab_nav** (int val)
Enables or disables Tab key focus navigation.

Public Member Functions inherited from [Fl_Text_Display](#)

- [Fl_Text_Buffer](#) * **buffer** () const
Gets the current text buffer associated with the text widget.
- void **buffer** ([Fl_Text_Buffer](#) &buf)
Sets the current text buffer associated with the text widget.
- void **buffer** ([Fl_Text_Buffer](#) *buf)
Attach a text buffer to display, replacing the current buffer (if any).
- double **col_to_x** (double col) const
Convert a column number into an x pixel position.
- int **count_lines** (int start, int [end](#), bool start_pos_is_line_start) const
Count the number of lines between two positions.
- [Fl_Color](#) **cursor_color** () const
Gets the text cursor color.
- void **cursor_color** ([Fl_Color](#) n)
Sets the text cursor color.
- int **cursor_style** () const
- void **cursor_style** (int style)
Sets the text cursor style.
- virtual void **display_needs_recalc** ()
Schedule a [recalc_display\(\)](#) to be done on next [draw\(\)](#).
- [Fl_Text_Display](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new text display widget.
- int **get_absolute_top_line_number** () const
Returns the absolute (non-wrapped) line number of the first line displayed.
- [Fl_Color](#) **grammar_underline_color** () const

- Gets the underline color for style attribute ATTR_GRAMMAR.*

 - void [grammar_underline_color](#) ([FI_Color](#) color)

Sets the underline color for style attribute ATTR_GRAMMAR.
- void **hide_cursor** ()

Hides the text cursor.
- void [highlight_data](#) ([FI_Text_Buffer](#) *styleBuffer, const [Style_Table_Entry](#) *styleTable, int nStyles, char unfinishedStyle, Unfinished_Style_Cb unfinishedHighlightCB, void *cbArg)

Attach (or remove) highlight information in text display and redisplay.
- int [in_selection](#) (int x, int y) const

Check if a pixel position is within the primary selection.
- void [insert](#) (const char *text)

Inserts "text" at the current cursor location.
- int [insert_position](#) () const

Gets the position of the text insertion cursor for text display.
- void [insert_position](#) (int newPos)

Sets the position of the text insertion cursor for text display.
- int [line_end](#) (int startPos, bool startPosIsLineStart) const

Returns the end of a line.
- int [line_start](#) (int pos) const

Return the beginning of a line.
- [FI_Align](#) **linenumber_align** () const

Returns the alignment used for line numbers (if enabled).
- void [linenumber_align](#) ([FI_Align](#) val)

Set alignment for line numbers (if enabled).
- [FI_Color](#) **linenumber_bgcolor** () const

Returns the background color used for line numbers (if enabled).
- void [linenumber_bgcolor](#) ([FI_Color](#) val)

Set the background color used for line numbers (if enabled).
- [FI_Color](#) **linenumber_fgcolor** () const

Return the foreground color used for line numbers (if enabled).
- void [linenumber_fgcolor](#) ([FI_Color](#) val)

Set the foreground color used for line numbers (if enabled).
- [FI_Font](#) **linenumber_font** () const

Return the font used for line numbers (if enabled).
- void [linenumber_font](#) ([FI_Font](#) val)

Set the font used for line numbers (if enabled).
- const char * **linenumber_format** () const

Returns the line number printf() format string.
- void [linenumber_format](#) (const char *val)

Sets the printf() style format string used for line numbers.
- [FI_Fontsize](#) **linenumber_size** () const

Return the font size used for line numbers (if enabled).
- void [linenumber_size](#) ([FI_Fontsize](#) val)

Set the font size used for line numbers (if enabled).
- int **linenumber_width** () const

Return the screen area width provided for line numbers.
- void [linenumber_width](#) (int width)

Set width of screen area for line numbers.
- int [move_down](#) ()

Moves the current insert position down one line.
- int [move_left](#) ()

- Moves the current insert position left one character.*
- int [move_right](#) ()
- Moves the current insert position right one character.*
- int [move_up](#) ()
- Moves the current insert position up one line.*
- void [next_word](#) (void)
- Moves the current insert position right one word.*
- void [overstrike](#) (const char *text)
- Replaces text at the current insert position.*
- int [position_style](#) (int lineStartPos, int lineLen, int lineIndex) const
- Find the correct style for a character.*
- int [position_to_xy](#) (int pos, int *x, int *y) const
- Convert a character index into a pixel position.*
- void [previous_word](#) (void)
- Moves the current insert position left one word.*
- virtual void [recalc_display](#) ()
- Recalculate the display's visible lines and scrollbar sizes.*
- void [redisplay_range](#) (int start, int end)
- Marks text from start to end as needing a redraw.*
- void [resize](#) (int X, int Y, int W, int H) [FL_OVERRIDE](#)
- Change the size of the displayed text area.*
- int [rewind_lines](#) (int startPos, int nLines)
- Skip a number of lines back.*
- void [scroll](#) (int topLineNum, int horizOffset)
- Scrolls the current buffer to start at the specified line and column.*
- int [scroll_col](#) ()
- int [scroll_row](#) ()
- [Fl_Align scrollbar_align](#) () const
- Gets the scrollbar alignment type.*
- void [scrollbar_align](#) ([Fl_Align](#) a)
- Sets the scrollbar alignment type.*
- int [scrollbar_size](#) () const
- Gets the current size of the scrollbars' troughs, in pixels.*
- void [scrollbar_size](#) (int newSize)
- Sets the pixel size of the scrollbars' troughs to newSize, in pixels.*
- int [scrollbar_width](#) () const
- Returns the global value [Fl::scrollbar_size\(\)](#) unless a specific scrollbar_width_ has been set.*
- void [scrollbar_width](#) (int width)
- Sets the global [Fl::scrollbar_size\(\)](#), and forces this instance of the widget to use it.*
- [Fl_Color secondary_selection_color](#) () const
- Gets the background color for the secondary selection block.*
- void [secondary_selection_color](#) ([Fl_Color](#) color)
- Sets the background color for the secondary selection block.*
- int [shortcut](#) () const
- void [shortcut](#) (int s)
- void [show_cursor](#) (int b=1)
- Shows the text cursor.*
- void [show_insert_position](#) ()
- Scrolls the text buffer to show the current insert position.*
- int [skip_lines](#) (int startPos, int nLines, bool startPosIsLineStart)
- Skip a number of lines forward.*

- [FL_Color spelling_underline_color](#) () const
Gets the underline color for style attribute ATTR_SPELLING.
- void [spelling_underline_color](#) ([FL_Color](#) color)
Sets the underline color for style attribute ATTR_SPELLING.
- [FL_Text_Buffer * style_buffer](#) () const
Gets the current style buffer associated with the text widget.
- [FL_Color textcolor](#) () const
Gets the default color of text in the widget.
- void [textcolor](#) ([FL_Color](#) n)
Sets the default color of text in the widget.
- [FL_Font textfont](#) () const
Gets the default font used when drawing text in the widget.
- void [textfont](#) ([FL_Font](#) s)
Sets the default font used when drawing text in the widget.
- [FL_Fontsize textsize](#) () const
Gets the default size of text in the widget.
- void [textsize](#) ([FL_Fontsize](#) s)
Sets the default size of text in the widget.
- int [word_end](#) (int pos) const
Moves the insert position to the end of the current word.
- int [word_start](#) (int pos) const
Moves the insert position to the beginning of the current word.
- void [wrap_mode](#) (int wrap, int wrap_margin)
Set the new text wrap mode.
- int [wrapped_column](#) (int row, int column) const
Nobody knows what this function does.
- int [wrapped_row](#) (int row) const
Nobody knows what this function does.
- double [x_to_col](#) (double x) const
Convert an x pixel position into a column number.
- [~FL_Text_Display](#) ()
Free a text display and release its associated memory.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget *& _ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void [add](#) ([FL_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void [add](#) ([FL_Widget](#) *o)
See void [FL_Group::add\(FL_Widget &w\)](#)
- void [add_resizable](#) ([FL_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget *const * array](#) () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * [as_group](#) () const [FL_OVERRIDE](#)
- [FL_Group](#) * [as_group](#) () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void [begin](#) ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget](#) * [child](#) (int n) const

- Returns the n'th child.*
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
Removes the widget at `index` from the group and deletes it.
- void **end** ()
Exactly the same as `current(this->parent())`.
- int **find** (const FL_Widget &o) const
See int `FL_Group::find(const FL_Widget *w)` const.
- int **find** (const FL_Widget *) const
Searches the children for the widget and returns the index.
- **FL_Group** (int, int, int, int, const char *s=0)
Creates a new `FL_Group` widget using the given position, size, and label string.
- void **focus** (FL_Widget *W)
- void **forms_end** ()
This is for forms compatibility only.
- void **init_sizes** ()
Resets the internal array of widget sizes and positions.
- void **insert** (FL_Widget &, int i)
The widget is removed from its current group (if any) and then inserted into this group.
- void **insert** (FL_Widget &o, FL_Widget *before)
This does `insert(w, find(before))`.
- void **remove** (FL_Widget &)
Removes a widget from the group but does not delete it.
- void **remove** (FL_Widget *o)
Removes the widget `o` from the group.
- void **remove** (int index)
Removes the widget at `index` from the group but does not delete it.
- **FL_Widget *resizable** () const
Returns the group's resizable widget.
- void **resizable** (FL_Widget &o)
Sets the group's resizable widget.
- void **resizable** (FL_Widget *o)
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- virtual **~FL_Group** ()
The destructor also deletes all the children.

Public Member Functions inherited from FL_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.

- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [Fl_Align](#) align () const
Gets the label alignment.
- void [align](#) ([Fl_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()
Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [Fl_Window](#) * [as_window](#) ()
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- virtual [Fl_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype](#) box () const
Gets the box type of the widget.
- void [box](#) ([Fl_Boxtype](#) new_box)
Sets the box type for the widget.
- [Fl_Callback_p](#) callback () const
Gets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([Fl_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()

- Hides the widget.*

 - void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FL_Color` bg)

Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)

Sets the background and selection color of the widget.
- `FL_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)

Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const

Gets the widget height.
- virtual void `hide` ()

Makes a widget invisible.

- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image * image ()`
Gets the image that is used as part of the widget label when in the active state.
- `const FI_Image * image () const`
Gets the image that is used as part of the widget label when in the active state.
- `void image (FI_Image &img)`
Sets the image to use as part of the widget label when in the active state.
- `void image (FI_Image *img)`
Sets the image to use as part of the widget label when in the active state.
- `int image_bound () const`
Returns whether the image is managed by the widget.
- `int inside (const FI_Widget *wgt) const`
Checks if this widget is a child of wgt.
- `int is_label_copied () const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FI_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FI_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FI_Color c)`
Sets the label color.
- `FI_Font labelfont () const`
Gets the font to use.
- `void labelfont (FI_Font f)`
Sets the font to use.
- `FI_Fonsize labelsz () const`
Gets the font size in pixels.
- `void labelsz (FI_Fonsize pix)`
Sets the font size in pixels.
- `FI_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (FI_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`
Sets width ww and height hh accordingly with the label size.
- `bool needs_keyboard () const`
Returns whether this widget needs a keyboard.
- `void needs_keyboard (bool needs)`
Sets whether this widget needs a keyboard.
- `unsigned int output () const`

- Returns if a widget is used for output only.*

 - `Fl_Group * parent () const`

Returns a pointer to the parent widget.
- `void parent (Fl_Group *p)`

Internal use only - "for hacks only".
- `void position (int X, int Y)`

Repositions the window or widget.
- `void redraw ()`

Schedules the drawing of the widget.
- `void redraw_label ()`

Schedules the drawing of the label.
- `Fl_Color selection_color () const`

Gets the selection color.
- `void selection_color (Fl_Color a)`

Sets the selection color.
- `void set_active ()`

Marks the widget as active without sending events or changing focus.
- `void set_changed ()`

Marks the value of the widget as changed.
- `void set_output ()`

Sets a widget to output only.
- `void set_visible ()`

Makes the widget visible.
- `void set_visible_focus ()`

Enables keyboard focus navigation with this widget.
- `int shortcut_label () const`

Returns whether the widget's label uses '&' to indicate shortcuts.
- `void shortcut_label (int value)`

Sets whether the widget's label uses '&' to indicate shortcuts.
- `virtual void show ()`

Makes a widget visible.
- `void size (int W, int H)`

Changes the size of the widget.
- `int take_focus ()`

Gives the widget the keyboard focus.
- `unsigned int takeevents () const`

Returns if the widget is able to take events.
- `int test_shortcut ()`

Returns true if the widget's label contains the entered '&x' shortcut.
- `const char * tooltip () const`

Gets the current tooltip text.
- `void tooltip (const char *text)`

Sets the current tooltip text.
- `Fl_Window * top_window () const`

Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`

Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`

Gets the widget type.
- `void type (uchar t)`

Sets the widget type.

- int **use_accents_menu** ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * **user_data** () const
Gets the user data for this widget.
- void **user_data** (FL_Callback_User_Data *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int **vertical_label_margin** ()
Get the spacing between the label and the vertical edge of the widget.
- void **vertical_label_margin** (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int **visible** () const
Returns whether a widget is visible.
- unsigned int **visible_focus** () const
Checks whether this widget has a visible focus.
- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- FL_When **when** () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- FL_Window * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual ~**FL_Widget** ()
Destroys the widget.

Static Public Member Functions

- static int **kf_backspace** (int c, FL_Text_Editor *e)
Does a backspace for key 'c' in the current buffer of editor 'e'.
- static int **kf_c_s_move** (int c, FL_Text_Editor *e)
Extends the current selection in the direction indicated by control key 'c' in editor 'e'.
- static int **kf_copy** (int c, FL_Text_Editor *e)
Does a copy of selected text or the current character in the current buffer of editor 'e'.
- static int **kf_ctrl_move** (int c, FL_Text_Editor *e)
Moves the current text cursor in the direction indicated by control key 'c' in editor 'e'.
- static int **kf_cut** (int c, FL_Text_Editor *e)
Does a cut of selected text in the current buffer of editor 'e'.
- static int **kf_default** (int c, FL_Text_Editor *e)
Inserts the text associated with key 'c' in editor 'e'.
- static int **kf_delete** (int c, FL_Text_Editor *e)
Does a delete of selected text or the current character in the current buffer of editor 'e'.

- static int `kf_down` (int c, `Fl_Text_Editor` *e)
Moves the text cursor one line down for editor 'e'.
- static int `kf_end` (int c, `Fl_Text_Editor` *e)
Moves the text cursor to the end of the current line in editor 'e'.
- static int `kf_enter` (int c, `Fl_Text_Editor` *e)
Inserts a newline for key 'c' at the current cursor position in editor 'e'.
- static int `kf_home` (int, `Fl_Text_Editor` *e)
Moves the text cursor to the beginning of the current line in editor 'e'.
- static int `kf_ignore` (int c, `Fl_Text_Editor` *e)
Ignores the key 'c' in editor 'e'.
- static int `kf_insert` (int c, `Fl_Text_Editor` *e)
Toggles the insert mode for editor 'e'.
- static int `kf_left` (int c, `Fl_Text_Editor` *e)
Moves the text cursor one character to the left in editor 'e'.
- static int `kf_m_s_move` (int c, `Fl_Text_Editor` *e)
Extends the current selection in the direction indicated by meta key 'c' in editor 'e'.
- static int `kf_meta_move` (int c, `Fl_Text_Editor` *e)
Moves the current text cursor in the direction indicated by meta key 'c' in editor 'e'.
- static int `kf_move` (int c, `Fl_Text_Editor` *e)
Moves the text cursor in the direction indicated by key 'c' in editor 'e'.
- static int `kf_page_down` (int c, `Fl_Text_Editor` *e)
Moves the text cursor down one page for editor 'e'.
- static int `kf_page_up` (int c, `Fl_Text_Editor` *e)
Moves the text cursor up one page for editor 'e'.
- static int `kf_paste` (int c, `Fl_Text_Editor` *e)
Does a paste of selected text in the current buffer of editor 'e'.
- static int `kf_redo` (int c, `Fl_Text_Editor` *e)
Redo last undo action.
- static int `kf_right` (int c, `Fl_Text_Editor` *e)
Moves the text cursor one character to the right for editor 'e'.
- static int `kf_select_all` (int c, `Fl_Text_Editor` *e)
Selects all text in the current buffer in editor 'e'.
- static int `kf_shift_move` (int c, `Fl_Text_Editor` *e)
Extends the current selection in the direction of key 'c' in editor 'e'.
- static int `kf_undo` (int c, `Fl_Text_Editor` *e)
Undo last edit in the current buffer of editor 'e'.
- static int `kf_up` (int c, `Fl_Text_Editor` *e)
Moves the text cursor one line up for editor 'e'.

Static Public Member Functions inherited from `Fl_Group`

- static `Fl_Group` * `current` ()
Returns the currently active group.
- static void `current` (`Fl_Group` *g)
Sets the current group.

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget` *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Member Functions

- int **handle_key** ()
Handles a key press in the editor.
- void **maybe_do_callback** (FI_Callback_Reason reason=FL_REASON_CHANGED)
does or does not a callback according to [changed\(\)](#) and [when\(\)](#) settings

Protected Member Functions inherited from [FI_Text_Display](#)

- void [absolute_top_line_number](#) (int oldFirstChar)
Re-calculate absolute top line number for a change in scroll position.
- void [calc_last_char](#) ()
Update last display character index.
- void [calc_line_starts](#) (int startLine, int endLine)
Update the line starts array.
- void [clear_rect](#) (int style, int x, int y, int width, int height) const
*Clear a rectangle with the appropriate background color for *style*.*
- void [display_insert](#) ()
Scroll the display to bring insertion cursor into view.
- void [draw](#) () [FL_OVERRIDE](#)
Draw the widget.
- void [draw_cursor](#) (int, int)
Draw a cursor with top center at X, Y.
- void [draw_line_numbers](#) (bool clearAll)
Refresh the line number area.
- void [draw_range](#) (int start, int end)
Draw a range of text.
- void [draw_string](#) (int style, int x, int y, int toX, const char *string, int nChars) const
Draw a text segment in a single style.
- void [draw_text](#) (int X, int Y, int W, int H)
Refresh a rectangle of the text display.
- void [draw_vline](#) (int visLineNum, int leftClip, int rightClip, int leftCharIndex, int rightCharIndex)
Draw a single line of text.
- int [empty_vlines](#) () const
Return true if there are lines visible with no corresponding buffer text.
- void [extend_range_for_styles](#) (int *start, int *end)
I don't know what this does!
- void [find_line_end](#) (int pos, bool start_pos_is_line_start, int *lineEnd, int *nextLineStart) const
Finds both the end of the current line and the start of the next line.
- void [find_wrap_range](#) (const char *deletedText, int pos, int nInserted, int nDeleted, int *modRangeStart, int *modRangeEnd, int *linesInserted, int *linesDeleted)
Wrapping calculations.
- int [find_x](#) (const char *s, int len, int style, int x) const
Find the index of the character that lies at the given x position / closest cursor position.
- int [handle_rmb](#) (int readonly)
Handle right mouse button down events.
- int [handle_vline](#) (int mode, int lineStart, int lineLen, int leftChar, int rightChar, int topClip, int bottomClip, int leftClip, int rightClip) const
Universal pixel machine.
- int [longest_vline](#) () const
Find the longest line of all visible lines.
- void [maintain_absolute_top_line_number](#) (int state)

- Line numbering stuff, currently unused.*
- int [maintaining_absolute_top_line_number](#) () const
Returns true if a separate absolute top line number is being maintained.
- void [measure_deleted_lines](#) (int pos, int nDeleted)
Wrapping calculations.
- double [measure_proportional_character](#) (const char *s, int colNum, int pos) const
Wrapping calculations.
- int [measure_vline](#) (int visLineNum) const
Returns the width in pixels of the displayed line pointed to by "visLineNum".
- void [offset_line_starts](#) (int newTopLineNum)
Offset line start counters for a new vertical scroll position.
- int [position_to_line](#) (int pos, int *lineNum) const
Convert a position index into a line number offset.
- int [position_to_linecol](#) (int pos, int *lineNum, int *column) const
Find the line and column number of position pos.
- void [reset_absolute_top_line_number](#) ()
Reestablish the absolute (non-wrapped) top line number.
- int [scroll_](#) (int topLineNum, int horizOffset)
Scrolls the current buffer to start at the specified line and column.
- double [string_width](#) (const char *string, int length, int style) const
Find the width of a string in the font of a particular style.
- void [update_h_scrollbar](#) ()
Update horizontal scrollbar.
- void [update_line_starts](#) (int pos, int charsInserted, int charsDeleted, int linesInserted, int linesDeleted, int *scrolled)
Update line start arrays and variables.
- void [update_v_scrollbar](#) ()
Update vertical scrollbar.
- int [vline_length](#) (int visLineNum) const
Count number of bytes in a visible line.
- int [wrap_uses_character](#) (int lineEndPos) const
Check if the line break is caused by a newline or by line wrapping.
- void [wrapped_line_counter](#) (FI_Text_Buffer *buf, int startPos, int maxPos, int maxLines, bool startPosIs↵LineStart, int styleBufOffset, int *retPos, int *retLines, int *retLineStart, int *retLineEnd, bool countLast↵LineMissingNewLine=true) const
Wrapping calculations.
- int [xy_to_position](#) (int x, int y, int PosType=CHARACTER_POS) const
Translate a pixel position into a character index.
- void [xy_to_rowcol](#) (int x, int y, int *row, int *column, int PosType=CHARACTER_POS) const
Translate pixel coordinates into row and column.

Protected Member Functions inherited from [FI_Group](#)

- [FI_Rect * bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw_child](#) (FI_Widget &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const FI_Widget &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) (FI_Widget *, int)

- *Allow derived groups to act when a widget is added as a child.*
- virtual int **on_move** (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void **on_remove** (int)
Allow derived groups to act when a child widget is removed from the group.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** (FI_Widget &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Static Protected Attributes

- static Key_Binding * **global_key_bindings**
Global key binding list.

Additional Inherited Members**Public Attributes inherited from [FI_Text_Display](#)****Protected Types inherited from [FI_Text_Display](#)**

- enum {
DRAW_LINE , **FIND_INDEX** , **FIND_INDEX_FROM_ZERO** , **GET_WIDTH** ,
FIND_CURSOR_INDEX }

Protected Types inherited from [FI_Widget](#)

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }

flags possible values enumeration.

Static Protected Member Functions inherited from [FI_Text_Display](#)

- static void [buffer_modified_cb](#) (int pos, int nInserted, int nDeleted, int nRestyled, const char *deletedText, void *cbArg)
This is called whenever the buffer is modified.
- static void [buffer_predelete_cb](#) (int pos, int nDeleted, void *cbArg)
This is called before any characters are deleted.
- static void [h_scrollbar_cb](#) ([FI_Scrollbar](#) *w, [FI_Text_Display](#) *d)
Callback for drag or valueChanged on horizontal scrollbar.
- static void [scroll_timer_cb](#) (void *)
Timer callback for scroll events.
- static void [v_scrollbar_cb](#) ([FI_Scrollbar](#) *w, [FI_Text_Display](#) *d)
Callback for drag or valueChanged on vertical scrollbar.

Protected Attributes inherited from [FI_Text_Display](#)

- int **damage_range1_end**
- int **damage_range1_start**
- int **damage_range2_end**
- int **damage_range2_start**
- int **display_insert_position_hint**
- bool **display_needs_recalc_**
- int **dragging**
- int **dragPos**
- int **dragType**
- [FI_Color](#) **grammar_underline_color_**
- [FI_Align](#) **linenumber_align_**
- [FI_Color](#) **linenumber_bgcolor_**
- [FI_Color](#) **linenumber_fgcolor_**

- [FL_Font](#) `linenumber_font_`
- `const char * linenumber_format_`
- [FL_Fontsize](#) `linenumber_size_`
- `int mAbsTopLineNum`
- [FL_Text_Buffer](#) * `mBuffer`
- `double mColumnScale`
- `int mContinuousWrap`
- [FL_Color](#) `mCursor_color`
- `int mCursorOldY`
- `int mCursorOn`
- `int mCursorPos`
- `int mCursorPreferredXPos`
- `int mCursorStyle`
- `int mCursorToHint`
- `int mFirstChar`
- `void * mHighlightCBAArg`
- `int mHorizOffset`
- `int mHorizOffsetHint`
- [FL_Scrollbar](#) * `mHScrollBar`
- `int mLastChar`
- `int mLineNumLeft`
- `int mLineNumWidth`
- `int * mLineStarts`
- `int mMaxsize`
- `int mModifyingTabDistance`
- `int mNBufferLines`
- `int mNeedAbsTopLineNum`
- `int mNLinesDeleted`
- `int mNStyles`
- `int mNVisibleLines`
- [FL_Text_Buffer](#) * `mStyleBuffer`
- `const Style\_Table\_Entry * mStyleTable`
- `int mSuppressResync`
- `int mTopLineNum`
- `int mTopLineNumHint`
- `Unfinished_Style_Cb mUnfinishedHighlightCB`
- `char mUnfinishedStyle`
- [FL_Scrollbar](#) * `mVScrollBar`
- `int mWrapMarginPix`
- [FL_Align](#) `scrollbar_align_`
- `int scrollbar_width_`
- [FL_Color](#) `secondary_selection_color_`
- `int shortcut_`
- [FL_Color](#) `spelling_underline_color_`
- `struct {`
 - `int h`
 - `int w`
 - `int x`
 - `int y`
- `} text_area`
- [FL_Color](#) `textcolor_`
- [FL_Font](#) `textfont_`
- [FL_Fontsize](#) `textsize_`

11.145.1 Detailed Description

This is the FLTK text editor widget.

It allows the user to edit multiple lines of text and supports highlighting and scrolling. The buffer that is displayed in the widget is managed by the [Fl_Text_Buffer](#) class.

11.145.2 Member Function Documentation

11.145.2.1 add_key_binding()

```
void Fl_Text_Editor::add_key_binding (
    int key,
    int state,
    Key_Func function,
    Key_Binding ** list)
```

Adds a key of state `state` with the function `function` to an arbitrary key binding list `list`.

This can be used in derived classes to add global key bindings by using the global (static) [Key_Binding](#) list [Fl_Text_Editor::global_key_bindings](#).

11.145.2.2 handle()

```
int Fl_Text_Editor::handle (
    int e) [virtual]
```

Event handling.

Reimplemented from [Fl_Text_Display](#).

11.145.2.3 insert_mode() [1/2]

```
int Fl_Text_Editor::insert_mode () [inline]
```

Gets the current insert mode; if non-zero, new text is inserted before the current cursor position.

Otherwise, new text replaces text at the current cursor position.

11.145.2.4 insert_mode() [2/2]

```
void Fl_Text_Editor::insert_mode (
    int b) [inline]
```

Sets the current insert mode; if non-zero, new text is inserted before the current cursor position.

Otherwise, new text replaces text at the current cursor position.

11.145.2.5 kf_backspace()

```
int Fl_Text_Editor::kf_backspace (
    int c,
    Fl_Text_Editor * e) [static]
```

Does a backspace for key 'c' in the current buffer of editor 'e'.

Any current selection is deleted. Otherwise, the character left is deleted and the cursor moved. The key value 'c' is currently unused.

11.145.2.6 kf_c_s_move()

```
int Fl_Text_Editor::kf_c_s_move (
    int c,
    Fl_Text_Editor * e) [static]
```

Extends the current selection in the direction indicated by control key 'c' in editor 'e'.

See also

[kf_ctrl_move\(\)](#).

11.145.2.7 kf_copy()

```
int Fl_Text_Editor::kf_copy (
    int c,
    Fl_Text_Editor * e) [static]
```

Does a copy of selected text or the current character in the current buffer of editor 'e'.
The key value 'c' is currently unused.

11.145.2.8 kf_ctrl_move()

```
int Fl_Text_Editor::kf_ctrl_move (
    int c,
    Fl_Text_Editor * e) [static]
```

Moves the current text cursor in the direction indicated by control key 'c' in editor 'e'.

Supported values for 'c' are currently:

```
FL_Home    -- moves the cursor to the beginning of the document
FL_End     -- moves the cursor to the end of the document
FL_Left    -- moves the cursor left one word
FL_Right    -- moves the cursor right one word
FL_Up      -- scrolls up one line, without moving cursor
FL_Down    -- scrolls down one line, without moving cursor
FL_Page_Up -- moves the cursor to the beginning of the top line on the current page
FL_Page_Down -- moves the cursor to the beginning of the last line on the current page
```

11.145.2.9 kf_cut()

```
int Fl_Text_Editor::kf_cut (
    int c,
    Fl_Text_Editor * e) [static]
```

Does a cut of selected text in the current buffer of editor 'e'.
The key value 'c' is currently unused.

11.145.2.10 kf_default()

```
int Fl_Text_Editor::kf_default (
    int c,
    Fl_Text_Editor * e) [static]
```

Inserts the text associated with key 'c' in editor 'e'.
Honors the current selection and insert/overstrike mode.

11.145.2.11 kf_delete()

```
int Fl_Text_Editor::kf_delete (
    int c,
    Fl_Text_Editor * e) [static]
```

Does a delete of selected text or the current character in the current buffer of editor 'e'.
The key value 'c' is currently unused.

11.145.2.12 kf_down()

```
int Fl_Text_Editor::kf_down (
    int c,
    Fl_Text_Editor * e) [static]
```

Moves the text cursor one line down for editor 'e'.
Same as kf_move(FL_Down, e). The key value 'c' is currently unused.

11.145.2.13 kf_end()

```
int Fl_Text_Editor::kf_end (
    int c,
    Fl_Text_Editor * e) [static]
```

Moves the text cursor to the end of the current line in editor 'e'.
Same as kf_move(FL_End, e). The key value 'c' is currently unused.

11.145.2.14 kf_enter()

```
int Fl_Text_Editor::kf_enter (
    int c,
    Fl_Text_Editor * e) [static]
```

Inserts a newline for key 'c' at the current cursor position in editor 'e'.
The key value 'c' is currently unused.

11.145.2.15 kf_home()

```
int Fl_Text_Editor::kf_home (
    int ,
    Fl_Text_Editor * e) [static]
```

Moves the text cursor to the beginning of the current line in editor 'e'.
Same as kf_move(FL_Home, e). The key value 'c' is currently unused.

11.145.2.16 kf_ignore()

```
int Fl_Text_Editor::kf_ignore (
    int c,
    Fl_Text_Editor * e) [static]
```

Ignores the key 'c' in editor 'e'.

This method can be used as a keyboard binding to disable a key that might otherwise be handled or entered as text.
An example would be disabling FL_Escape, so that it isn't added to the buffer when invoked by the user.

11.145.2.17 kf_insert()

```
int Fl_Text_Editor::kf_insert (
    int c,
    Fl_Text_Editor * e) [static]
```

Toggles the insert mode for editor 'e'.

The key value 'c' is currently unused.

11.145.2.18 kf_left()

```
int Fl_Text_Editor::kf_left (
    int c,
    Fl_Text_Editor * e) [static]
```

Moves the text cursor one character to the left in editor 'e'.

Same as kf_move(FL_Left, e). The key value 'c' is currently unused.

11.145.2.19 kf_m_s_move()

```
int Fl_Text_Editor::kf_m_s_move (
    int c,
    Fl_Text_Editor * e) [static]
```

Extends the current selection in the direction indicated by meta key 'c' in editor 'e'.

See also

[kf_meta_move\(\)](#).

11.145.2.20 kf_meta_move()

```
int Fl_Text_Editor::kf_meta_move (
    int c,
    Fl_Text_Editor * e) [static]
```

Moves the current text cursor in the direction indicated by meta key 'c' in editor 'e'.

Supported values for 'c' are currently:

[FL_Up](#) -- moves cursor to the beginning of the [current](#) document

```

FL_Down      -- moves cursor to the end of the current document
FL_Left      -- moves the cursor to the beginning of the current line
FL_Right     -- moves the cursor to the end of the current line

```

11.145.2.21 kf_move()

```

int Fl_Text_Editor::kf_move (
    int c,
    Fl_Text_Editor * e) [static]

```

Moves the text cursor in the direction indicated by key 'c' in editor 'e'.

Supported values for 'c' are currently:

```

FL_Home      -- moves the cursor to the beginning of the current line
FL_End       -- moves the cursor to the end of the current line
FL_Left      -- moves the cursor left one character
FL_Right     -- moves the cursor right one character
FL_Up        -- moves the cursor up one line
FL_Down      -- moves the cursor down one line
FL_Page_Up   -- moves the cursor up one page
FL_Page_Down -- moves the cursor down one page

```

11.145.2.22 kf_page_down()

```

int Fl_Text_Editor::kf_page_down (
    int c,
    Fl_Text_Editor * e) [static]

```

Moves the text cursor down one page for editor 'e'.

Same as kf_move(FL_Page_Down, e). The key value 'c' is currently unused.

11.145.2.23 kf_page_up()

```

int Fl_Text_Editor::kf_page_up (
    int c,
    Fl_Text_Editor * e) [static]

```

Moves the text cursor up one page for editor 'e'.

Same as kf_move(FL_Page_Up, e). The key value 'c' is currently unused.

11.145.2.24 kf_paste()

```

int Fl_Text_Editor::kf_paste (
    int c,
    Fl_Text_Editor * e) [static]

```

Does a paste of selected text in the current buffer of editor 'e'.

Any current selection is replaced with the pasted content. The key value 'c' is currently unused.

11.145.2.25 kf_redo()

```

int Fl_Text_Editor::kf_redo (
    int c,
    Fl_Text_Editor * e) [static]

```

Redo last undo action.

Also deselects previous selection. The key value 'c' is currently unused.

11.145.2.26 kf_right()

```

int Fl_Text_Editor::kf_right (
    int c,
    Fl_Text_Editor * e) [static]

```

Moves the text cursor one character to the right for editor 'e'.

Same as kf_move(FL_Right, e). The key value 'c' is currently unused.

11.145.2.27 kf_select_all()

```
int Fl_Text_Editor::kf_select_all (
    int c,
    Fl_Text_Editor * e) [static]
```

Selects all text in the current buffer in editor 'e'.

The key value 'c' is currently unused.

11.145.2.28 kf_shift_move()

```
int Fl_Text_Editor::kf_shift_move (
    int c,
    Fl_Text_Editor * e) [static]
```

Extends the current selection in the direction of key 'c' in editor 'e'.

See also

[kf_move\(\)](#)

11.145.2.29 kf_undo()

```
int Fl_Text_Editor::kf_undo (
    int c,
    Fl_Text_Editor * e) [static]
```

Undo last edit in the current buffer of editor 'e'.

Also deselects previous selection. The key value 'c' is currently unused.

11.145.2.30 kf_up()

```
int Fl_Text_Editor::kf_up (
    int c,
    Fl_Text_Editor * e) [static]
```

Moves the text cursor one line up for editor 'e'.

Same as `kf_move(FL_Up, e)`. The key value 'c' is currently unused.

11.145.2.31 remove_key_binding()

```
void Fl_Text_Editor::remove_key_binding (
    int key,
    int state,
    Key_Binding ** list)
```

Removes the key binding associated with the key `key` of state `state` from the [Key_Binding](#) list `list`.

This can be used in derived classes to remove global key bindings by using the global (static) [Key_Binding](#) list `Fl_Text_Editor::global_key_bindings`.

11.145.2.32 tab_nav() [1/2]

```
int Fl_Text_Editor::tab_nav () const
```

Check if Tab focus navigation is enabled.

If disabled (default), hitting Tab inserts a tab character into the editor buffer.

If enabled, hitting Tab navigates focus to the next widget, and Shift-Tab navigates focus to the previous widget.

Returns

if Tab inserts tab characters or moves the focus

Return values

0	Tab inserts tab characters (default)
1	Tab navigation is enabled.

See also

[tab_nav\(int\)](#), [Fl::OPTION_ARROW_FOCUS](#).

Version

1.3.4 ABI feature

11.145.2.33 `tab_nav()` [2/2]

```
void Fl_Text_Editor::tab_nav (
    int val)
```

Enables or disables Tab key focus navigation.

When disabled (default), tab characters are inserted into [Fl_Text_Editor](#). Only the mouse can change focus. This behavior is desirable when [Fl_Text_Editor](#) is used, e.g. in a source code editor.

When enabled, Tab navigates focus to the next widget, and Shift-Tab navigates focus to the previous widget. This behavior is desirable when [Fl_Text_Editor](#) is used e.g. in a database input form.

Currently, this method is implemented as a convenience method that adjusts the key bindings for the Tab key. This implementation detail may change in the future. Know that changing the editor's key bindings for Tab and Shift-Tab may affect tab navigation.

Parameters

<code>in</code>	<code>val</code>	If <code>val</code> is 0, Tab inserts a tab character (default). If <code>val</code> is 1, Tab navigates widget focus.

See also

[tab_nav\(\)](#), [Fl::OPTION_ARROW_FOCUS](#).

Version

1.3.4 ABI feature

11.145.3 Member Data Documentation

11.145.3.1 `global_key_bindings`

```
Key\_Binding\* Fl_Text_Editor::global_key_bindings [static], [protected]
```

Global key binding list.

Derived classes can add key bindings for all [Fl_Text_Editor](#) widgets by adding a [Key_Binding](#) to this list.

See also

[add_key_binding\(int key, int state, Key_Func f, Key_Binding** list\);](#)

The documentation for this class was generated from the following files:

- `Fl_Text_Editor.H`
- `Fl_Text_Editor.cxx`

11.146 `Fl_Text_Selection` Class Reference

This is an internal class for [Fl_Text_Buffer](#) to manage text selections.

```
#include <Fl_Text_Buffer.H>
```


Public Member Functions

- int [end](#) () const
Returns the byte offset to the character after the last selected character.
- int [includes](#) (int pos) const
Returns true if position `pos` is in the [FI_Text_Selection](#).
- int [length](#) () const
Returns the size in bytes of the selection.
- int [position](#) (int *startpos, int *endpos) const
- bool [selected](#) () const
Returns true if any text is selected.
- void [selected](#) (bool b)
Modifies the 'selected' flag.
- int [selected](#) (int *startpos, int *endpos) const
Returns the status and the positions of this selection.
- void [set](#) (int startpos, int endpos)
Sets the selection range.
- int [start](#) () const
Returns the byte offset to the first selected character.
- void [update](#) (int pos, int nDeleted, int nInserted)
Updates a selection after text was modified.

Protected Attributes

- int **mEnd**
byte offset to the character after the last selected character
- bool **mSelected**
this flag is set if any text is selected
- int **mStart**
byte offset to the first selected character

Friends

- class [FI_Text_Buffer](#)

11.146.1 Detailed Description

This is an internal class for [FI_Text_Buffer](#) to manage text selections.

All methods use byte (not UTF-8 character) offsets and start at 0. This class works correctly with UTF-8 strings assuming that the parameters for all calls are on character boundaries.

If the selection is inactive (not currently used), then [selected\(\)](#) returns `false` and [start\(\)](#) and [end\(\)](#) return 0 (zero).

The stored offsets are in ascending order, hence the following conditions are true (pseudo code):

```
if ( !selected() ) : (start() == 0) && (end() == 0) && (start() == end())
if ( selected() ) : start() < end()
always           : 0 <= start() <= end()
always           : length() == end() - start()
```

The selection size in bytes can always (unconditionally) be computed by

```
int size = sel->end() - sel->start();
```

See also

[length\(\)](#)

Note

The **protected** member variables `mStart` and `mEnd` are not necessarily 0 (zero) if `mSelected == false` because they are not cleared when `selected(false)` is called (as of Jul 2017). This may be changed in the future.

11.146.2 Member Function Documentation

11.146.2.1 end()

```
int Fl_Text_Selection::end () const [inline]
```

Returns the byte offset to the character after the last selected character.

The returned offset is only valid if [selected\(\)](#) returns true (non-zero). The offset is 0 if no text is selected (since FLTK 1.4.0).

Note

In FLTK 1.3.x the returned offset could be non-zero even if [selected\(\)](#) would have returned 0.

Returns

byte offset or 0 if not selected.

11.146.2.2 includes()

```
int Fl_Text_Selection::includes (
    int pos) const
```

Returns true if position `pos` is in the [Fl_Text_Selection](#).

`pos` must be at a character boundary.

11.146.2.3 length()

```
int Fl_Text_Selection::length () const [inline]
```

Returns the size in bytes of the selection.

This is a convenience method. It always returns the same as

[end\(\)](#) - [start\(\)](#)

and it returns 0 if [selected\(\)](#) == false.

Returns

size in bytes or 0 if not selected.

Since

FLTK 1.4.0

11.146.2.4 position()

```
int Fl_Text_Selection::position (
    int * startpos,
    int * endpos) const [inline]
```

Deprecated "since 1.4.0 - use [selected\(startpos, endpos\)](#) instead"

11.146.2.5 selected() [1/3]

```
bool Fl_Text_Selection::selected () const [inline]
```

Returns true if any text is selected.

Returns

true if any text has been selected, or false if no text is selected.

11.146.2.6 selected() [2/3]

```
void Fl_Text_Selection::selected (
    bool b) [inline]
```

Modifies the 'selected' flag.

Parameters

<i>b</i>	new flag

11.146.2.7 selected() [3/3]

```
int Fl_Text_Selection::selected (
    int * startpos,
    int * endpos) const
```

Returns the status and the positions of this selection.

This method returns the same as [selected\(\)](#) as an `int` (0 or 1) in its return value and the offsets to the start of the selection in *startpos* and to the byte after the last selected character in *endpos*, if [selected\(\)](#) is `true`. If [selected\(\)](#) is `false`, both offsets are set to 0.

Note

In FLTK 1.3.x *startpos* and *endpos* were **not modified** if [selected\(\)](#) was `false`.

Parameters

<i>startpos</i>	return byte offset to first selected character
<i>endpos</i>	return byte offset pointing after last selected character

Returns

whether the selection is active ([selected\(\)](#)) or not

Return values

0	if not selected
1	if selected

See also

[selected\(\)](#), [start\(\)](#), [end\(\)](#)

11.146.2.8 set()

```
void Fl_Text_Selection::set (
    int startpos,
    int endpos)
```

Sets the selection range.

startpos and *endpos* must be at a character boundary.

If *startpos* != *endpos* [selected\(\)](#) is set to `true`, else to `false`.

If *startpos* is greater than *endpos* they are swapped so that *startpos* <= *endpos*.

Parameters

in	<i>startpos</i>	byte offset to first selected character
in	<i>endpos</i>	byte offset pointing after last selected character

11.146.2.9 start()

```
int Fl_Text_Selection::start () const [inline]
```

Returns the byte offset to the first selected character.

The returned offset is only valid if [selected\(\)](#) returns true. If the selection is not valid the returned offset is 0 since FLTK 1.4.0.

Note

In FLTK 1.3.x the returned offset could be non-zero even if [selected\(\)](#) would have returned 0.

Returns

byte offset or 0 if not selected.

11.146.2.10 update()

```
void Fl_Text_Selection::update (
    int pos,
    int nDeleted,
    int nInserted)
```

Updates a selection after text was modified.

Updates an individual selection for changes in the corresponding text.

Parameters

<i>pos</i>	byte offset into text buffer at which the change occurred
<i>nDeleted</i>	number of bytes deleted from the buffer
<i>nInserted</i>	number of bytes inserted into the buffer

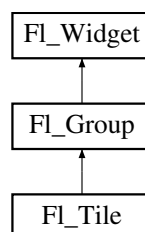
The documentation for this class was generated from the following files:

- [Fl_Text_Buffer.H](#)
- [Fl_Text_Buffer.cxx](#)

11.147 Fl_Tile Class Reference

The [Fl_Tile](#) class lets you resize its children by dragging the border between them.

Inheritance diagram for [Fl_Tile](#):



Classes

- struct [Size_Range](#)

Public Member Functions

- virtual void [drag_intersection](#) (int oldx, int oldy, int newx, int newy)
Drags the intersection at (oldx,oldy) to (newx,newy).
- [Fl_Tile](#) (int X, int Y, int W, int H, const char *L=0)

- Creates a new *FL_Tile* widget using the given position, size, and label string.
- int **handle** (int event) **FL_OVERRIDE**
Handles the specified event.
- void **init_size_range** (int default_min_w=-1, int default_min_h=-1)
Initialize the size range mode of *FL_Tile* and set the default minimum width and height.
- virtual void **move_intersection** (int oldx, int oldy, int newx, int newy)
Drags the intersection at (*oldx,oldy*) to (*newx,newy*).
- void **position** (int oldx, int oldy, int newx, int newy)
- void **position** (int x, int y)
- void **resize** (int X, int Y, int W, int H) **FL_OVERRIDE**
Resizes the *FL_Tile* widget and its children.
- void **size_range** (*FL_Widget* *w, int minw, int minh, int maxw=0x7FFFFFFF, int maxh=0x7FFFFFFF)
Set the allowed size range for the give child widget.
- void **size_range** (int index, int minw, int minh, int maxw=0x7FFFFFFF, int maxh=0x7FFFFFFF)
Set the allowed size range for the child at the given index.
- ~**FL_Tile** () **FL_OVERRIDE**
Destructor.

Public Member Functions inherited from *FL_Group*

- *FL_Widget* *& **_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** (*FL_Widget* &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** (*FL_Widget* *o)
See void *FL_Group::add(FL_Widget &w)*
- void **add_resizable** (*FL_Widget* &o)
Adds a widget to the group and makes it the resizable widget.
- *FL_Widget* *const * **array** () const
Returns a pointer to the internal array of children.
- *FL_Group* const * **as_group** () const **FL_OVERRIDE**
- *FL_Group* * **as_group** () **FL_OVERRIDE**
Returns an *FL_Group* pointer if this widget is an *FL_Group*.
- void **begin** ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- *FL_Widget* * **child** (int n) const
Returns the n'th child.
- int **children** () const
Returns how many child widgets the group has.
- void **clear** ()
Deletes all child widgets from memory recursively.
- unsigned int **clip_children** ()
Returns the current clipping mode.
- void **clip_children** (int c)
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- virtual int **delete_child** (int n)
Removes the widget at *index* from the group and deletes it.
- void **end** ()
Exactly the same as *current(this->parent())*.
- int **find** (const *FL_Widget* &o) const
See int *FL_Group::find(const FL_Widget *w)* const.

- `int find (const FL\_Widget *) const`
Searches the children for the widget and returns the index.
- `FL\_Group (int, int, int, int, const char *=0)`
Creates a new [FL_Group](#) widget using the given position, size, and label string.
- `void focus (FL\_Widget *W)`
- `void forms_end ()`
This is for forms compatibility only.
- `void init_sizes ()`
Resets the internal array of widget sizes and positions.
- `void insert (FL\_Widget &, int i)`
The widget is removed from its current group (if any) and then inserted into this group.
- `void insert (FL\_Widget &o, FL\_Widget *before)`
This does insert(w, find(before)).
- `void remove (FL\_Widget &)`
Removes a widget from the group but does not delete it.
- `void remove (FL\_Widget *o)`
Removes the widget o from the group.
- `void remove (int index)`
Removes the widget at index from the group but does not delete it.
- `FL\_Widget *resizable () const`
Returns the group's resizable widget.
- `void resizable (FL\_Widget &o)`
Sets the group's resizable widget.
- `void resizable (FL\_Widget *o)`
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- `virtual ~FL\_Group ()`
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Widget](#)

- `void _clear_fullscreen ()`
- `void _set_fullscreen ()`
- `void activate ()`
Activates the widget.
- `unsigned int active () const`
Returns whether the widget is active.
- `int active_r () const`
Returns whether the widget and all of its parents are active.
- `FL\_Align align () const`
Gets the label alignment.
- `void align (FL\_Align alignment)`
Sets the label alignment.
- `long argument () const`
Gets the current user data (long) argument that is passed to the callback function.
- `void argument (long v)`
Sets the current user data (long) argument that is passed to the callback function.
- `virtual class FL\_Gl\_Window * as_gl_window ()`
Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).
- `virtual class FL\_Gl\_Window const * as_gl_window () const`
- `virtual FL\_Window * as_window ()`
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).

- virtual [FI_Window](#) const * **as_window** () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) **box** () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) **callback** () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [FI_Color](#) **color** () const
Gets the background color of the widget.
- void [color](#) ([FI_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([FI_Color](#) bg, [FI_Color](#) sel)
Sets the background and selection color of the widget.
- [FI_Color](#) **color2** () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [FI_Widget](#) *w) const
Checks if w is a child of this widget.

- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- [uchar](#) [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) ([uchar](#) c)
Sets the damage bits for the widget.
- void [damage](#) ([uchar](#) c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int [damage_resize](#) (int, int, int, int)
Internal use only.
- void [deactivate](#) ()
Deactivates the widget.
- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const

- Checks if this widget is a child of wgt.*

 - int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const

Gets the current label text.
- void `label` (const char *text)

Sets the current label pointer.
- void `label` (FL_Labeltype a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- FL_Color `labelcolor` () const

Gets the label color.
- void `labelcolor` (FL_Color c)

Sets the label color.
- FL_Font `labelfont` () const

Gets the font to use.
- void `labelfont` (FL_Font f)

Sets the font to use.
- FL_Fonsize `labelsize` () const

Gets the font size in pixels.
- void `labelsize` (FL_Fonsize pix)

Sets the font size in pixels.
- FL_Labeltype `labeltype` () const

Gets the label type.
- void `labeltype` (FL_Labeltype a)

Sets the label type.
- void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- FL_Group * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (FL_Group *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- FL_Color `selection_color` () const

Gets the selection color.
- void `selection_color` (FL_Color a)

Sets the selection color.

- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if `MAC_USE_ACCENTS_MENU` flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const

- Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [FL_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) (uchar i)
Sets the flags used to decide when a callback is called.
- [FL_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual [~FL_Widget](#) ()
Destroys the widget.

Protected Member Functions

- [FL_Cursor](#) [cursor](#) (int n)
Returns the cursor for cursor index n.
- int [on_insert](#) ([FL_Widget](#) *, int) [FL_OVERRIDE](#)
Insert a new entry in the size range list.
- int [on_move](#) (int, int) [FL_OVERRIDE](#)
Move the entry in the size range list.
- void [on_remove](#) (int) [FL_OVERRIDE](#)
Remove the entry from the size range list.
- void [request_grow_b](#) (int old_b, int &new_b, [FL_Rect](#) *final_size)
Request for children to change their layout.
- void [request_grow_l](#) (int old_l, int &new_l, [FL_Rect](#) *final_size)
Request for children to change their layout.
- void [request_grow_r](#) (int old_r, int &new_r, [FL_Rect](#) *final_size)
Request for children to change their layout.
- void [request_grow_t](#) (int old_t, int &new_t, [FL_Rect](#) *final_size)
Request for children to change their layout.
- void [request_shrink_b](#) (int old_b, int &new_b, [FL_Rect](#) *final_size)
Request for children to change their layout.
- void [request_shrink_l](#) (int old_l, int &new_l, [FL_Rect](#) *final_size)
Request for children to change their layout.
- void [request_shrink_r](#) (int old_r, int &new_r, [FL_Rect](#) *final_size)
Request for children to change their layout.
- void [request_shrink_t](#) (int old_t, int &new_t, [FL_Rect](#) *final_size)
Request for children to change their layout.
- void [set_cursor](#) (int n)
Set one of four cursors used for dragging etc. . .

Protected Member Functions inherited from [FI_Group](#)

- [FI_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw](#) () [FL_OVERRIDE](#)
Draws the widget.
- void [draw_child](#) ([FI_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [FI_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- int * [sizes](#) ()
Returns the internal array of widget sizes and positions.
- void [update_child](#) ([FI_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FI_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.
- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int [flags](#) () const
Gets the widget flags mask.
- void [h](#) (int v)
Internal use only.
- void [set_flag](#) (unsigned int c)
Sets a flag in the flags mask.
- void [w](#) (int v)
Internal use only.
- void [x](#) (int v)
Internal use only.
- void [y](#) (int v)
Internal use only.

Protected Attributes

- int **cursor_**
current cursor index (0..3)
- [FI_Cursor](#) * **cursors_**
points at the array of 4 cursors (may be overridden)
- int **default_min_h_**
- int **default_min_w_**
- [Size_Range](#) * **size_range_**
- int **size_range_capacity_**
- int **size_range_size_**

Additional Inherited Members

Static Public Member Functions inherited from [FI_Group](#)

- static [FI_Group](#) * **current** ()
Returns the currently active group.
- static void **current** ([FI_Group](#) *g)
Sets the current group.

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [FI_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.147.1 Detailed Description

The [FI_Tile](#) class lets you resize its children by dragging the border between them.

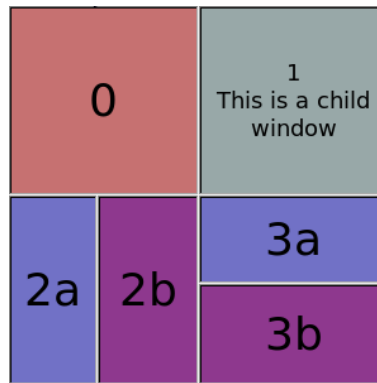


Figure 11.60 FL_Tile

For the tiling to work correctly, the children of an `FL_Tile` **must** cover the entire area of the widget, but **must not** overlap. This means that all children must touch each other at their edges, and no gaps can be left inside the `FL_Tile`.

`FL_Tile` does not normally draw any graphics of its own. The "borders" which can be seen in the snapshot above are actually part of the children. Their boxtypes have been set to `FL_DOWN_BOX` creating the impression of "ridges" where the boxes touch. What you see are actually two adjacent `FL_DOWN_BOX`'s drawn next to each other. All neighboring widgets share the same edge - the widget's thick borders make it appear as though the widgets aren't actually touching, but they are. If the edges of adjacent widgets do not touch, then it will be impossible to drag the corresponding edges.

Note

`FL_Tile` works in two distinctive modes. In classic mode, the range of motion for edges and intersections is controlled using an invisible child that is marked as the `resizable()` widget of the tile group. Classic mode is described in detail a few paragraphs down.

`FL_Tile` size_range mode

By assigning a default minimum size to all children with `FL_Tile::init_size_range(int default_minimum_width, int default_minimum_height)`, or by assigning minimal sizes to individual children with `size_range(FL_Widget *child, int minimum_width, int minimum_height)`, the tile group is put into `size_range` operation mode.

In this mode, the child that is marked `resizable()` will behave as it would in a regular `FL_Group` widget. When dragging edges or intersections with the mouse, `FL_Tile` will ensure that none of the children shrinks to a size that is smaller than requested. When resizing the `FL_Tile` group, size ranges are not enforced by the tile. Instead, the size range of the enclosing window should be limited to a valid range.

Tile does not differentiate between visible and invisible children. If children are created smaller than their assigned minimum size, dragging intersections may cause unexpected jumps in size. Zero width or height widget are not harmful, but should be avoided.

Example for a center document tile and two tool boxes on the left and right

```
FL_Window win(400, 300, "My App");

FL_Tile tile(0, 0, 400, 300);

FL_Box left_tool_box(0, 0, 100, 300, "Tools");
left_tool_box.box(FL_DOWN_BOX);
tile.size_range(&left_tool_box, 50, 50);

FL_Box document(100, 0, 200, 300, "Document");
document.box(FL_DOWN_BOX);
tile.size_range(&document, 100, 50);

FL_Box right_tool_box(300, 0, 400, 300, "More\nTools");
right_tool_box.box(FL_DOWN_BOX);
tile.size_range(&right_tool_box, 50, 50);

tile.end();
tile.resizable(document);

win.end();
win.resizable(tile);
win.show(argc, argv);
win.size_range(200, 50);
```

FL_Tile classic mode

FL_Tile allows objects to be resized to zero dimensions. To prevent this you can use the [resizable\(\)](#) to limit where corners can be dragged to. For more information see note below.

Even though objects can be resized to zero sizes, they must initially have non-zero sizes so the FL_Tile can figure out their layout. If desired, call [position\(\)](#) after creating the children but before displaying the window to set the borders where you want.

Note on [resizable\(FL_Widget &w\)](#): The "resizable" child widget (which should be invisible) limits where the borders can be dragged to. All dragging will be limited inside the resizable widget's borders. If you don't set it, it will be possible to drag the borders right to the edges of the FL_Tile widget, and thus resize objects on the edges to zero width or height. When the entire FL_Tile widget is resized, the [resizable\(\)](#) widget will keep its border distance to all borders the same (this is normal resize behavior), so that you can effectively set a border width that will never change. To ensure correct event delivery to all child widgets the [resizable\(\)](#) widget must be the first child of the FL_Tile widget group. Otherwise some events (e.g. FL_MOVE and FL_ENTER) might be consumed by the [resizable\(\)](#) widget so that they are lost for widgets covered (overlapped) by the [resizable\(\)](#) widget.

Note

You can still resize widgets **inside** the [resizable\(\)](#) to zero width and/or height, i.e. box **2b** above to zero width and box **3a** to zero height.

See also

[void FL_Group::resizable\(FL_Widget &w\)](#)

Example for resizable with 20 pixel border distance:

```
int dx = 20, dy = dx;
FL_Tile tile(50,50,300,300);
// create resizable() box first
FL_Box r(tile.x()+dx,tile.y()+dy,tile.w()-2*dx,tile.h()-2*dy);
tile.resizable(r);
// ... create widgets inside tile (see test/tile.cxx) ...
tile.end();
```

See also the complete example program in test/tile.cxx.

11.147.2 Constructor & Destructor Documentation

11.147.2.1 FL_Tile()

```
FL_Tile::FL_Tile (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new FL_Tile widget using the given position, size, and label string.

The default boxtype is FL_NO_BOX.

The destructor *also deletes all the children*. This allows a whole tree to be deleted at once, without having to keep a pointer to all the children in the user code. A kludge has been done so the FL_Tile and all of its children can be automatic (local) variables, but you must declare the FL_Tile *first*, so that it is destroyed last.

See also

[class FL_Group](#)

11.147.3 Member Function Documentation

11.147.3.1 cursor()

```
FL_Cursor FL_Tile::cursor (
    int n) [inline], [protected]
```

Returns the cursor for cursor index n.

See also

[FL_Tile::set_cursor\(int\)](#)

11.147.3.2 drag_intersection()

```
void Fl_Tile::drag_intersection (
    int oldx,
    int oldy,
    int newx,
    int newy) [virtual]
```

Drags the intersection at (oldx,oldy) to (newx,newy).

See also

[Fl_Tile::move_intersection\(int oldx, int oldy, int newx, int newy\)](#) , but this method does not call [init_sizes\(\)](#) and is used for interactive [children](#) layout using the mouse.

Parameters

in	<i>oldx,oldy</i>	move the intersection at this coordinate, pass zero to disable drag in that direction.
in	<i>newx,newy</i>	move the intersection as close to this new coordinate as possible

11.147.3.3 handle()

```
int Fl_Tile::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Group](#).

11.147.3.4 init_size_range()

```
void Fl_Tile::init_size_range (
    int default_min_w = -1,
    int default_min_h = -1)
```

Initialize the size range mode of [Fl_Tile](#) and set the default minimum width and height.

The default minimum width and height is the size of the mouse pointer grab area at about 4 pixel units.

Parameters

in	<i>default_min_w, default_min_h</i>	default size range for widgets that don't have an individual range assigned

11.147.3.5 move_intersection()

```
void Fl_Tile::move_intersection (
    int oldx,
    int oldy,
    int newx,
    int newy) [virtual]
```

Drags the intersection at (oldx,oldy) to (newx,newy).

This redraws all the necessary children.

If no size ranges are set, the new intersection position is limited to the size of the tile group. The [resizable\(\)](#) option is not taken into account here.

If size ranges are set, the actual new position of the intersection will depend on the size range of every individual child. No child will be smaller than their minw and minh. After the new position is found, [move_intersection\(\)](#) will call [init_sizes\(\)](#). The [resizable\(\)](#) range is ignored.

Parameters

in	<i>oldx,oldy</i>	move the intersection at this coordinate, pass zero to disable drag in that direction.
in	<i>newx,newy</i>	move the intersection as close to this new coordinate as possible

11.147.3.6 on_insert()

```
int Fl_Tile::on_insert (
    Fl_Widget * candidate,
    int index) [protected], [virtual]
```

Insert a new entry in the size range list.

Reimplemented from [Fl_Group](#).

11.147.3.7 on_move()

```
int Fl_Tile::on_move (
    int oldIndex,
    int newIndex) [protected], [virtual]
```

Move the entry in the size range list.

Reimplemented from [Fl_Group](#).

11.147.3.8 on_remove()

```
void Fl_Tile::on_remove (
    int index) [protected], [virtual]
```

Remove the entry from the size range list.

Reimplemented from [Fl_Group](#).

11.147.3.9 position()

```
void Fl_Tile::position (
    int oldx,
    int oldy,
    int newx,
    int newy) [inline]
```

Deprecated "since 1.4.0 - use [move_intersection\(p\)](#) instead"

11.147.3.10 request_grow_b()

```
void Fl_Tile::request_grow_b (
    int old_b,
    int & new_b,
    Fl_Rect * final_size) [protected]
```

Request for children to change their layout.

Parameters

in	<i>old_b</i>	grow all children with this current bottom edge toward the bottom edge of this tile
in, out	<i>new_b</i>	try to grow to this coordinate, return the maximum possible growth (currently maxh is ignored, so we always grow to new_b)
in, out	<i>final_size</i>	write the new position and size of all affected children into this list of Fl_Rect

11.147.3.11 request_grow_l()

```
void Fl_Tile::request_grow_l (
    int old_l,
    int & new_l,
    Fl_Rect * final_size) [protected]
```

Request for children to change their layout.

Parameters

in	<i>old_l</i>	grow all children with this current left edge toward the left edge of this tile
in, out	<i>new_l</i>	try to grow to this coordinate, return the maximum possible growth (currently maxw is ignored, so we always grow to new_l)
in, out	<i>final_size</i>	write the new position and size of all affected children into this list of Fl_Rect

11.147.3.12 request_grow_r()

```
void Fl_Tile::request_grow_r (
    int old_r,
    int & new_r,
    Fl_Rect * final_size) [protected]
```

Request for children to change their layout.

Parameters

in	<i>old_r</i>	grow all children with this current right edge toward the right edge of this tile
in, out	<i>new_r</i>	try to grow to this coordinate, return the maximum possible growth (currently maxw is ignored, so we always grow to new_r)
in, out	<i>final_size</i>	write the new position and size of all affected children into this list of Fl_Rect

11.147.3.13 request_grow_t()

```
void Fl_Tile::request_grow_t (
    int old_t,
    int & new_t,
    Fl_Rect * final_size) [protected]
```

Request for children to change their layout.

Parameters

in	<i>old_t</i>	grow all children with this current top edge toward the top edge of this tile
in, out	<i>new_t</i>	try to grow to this coordinate, return the maximum possible growth (currently maxh is ignored, so we always grow to new_t)
in, out	<i>final_size</i>	write the new position and size of all affected children into this list of Fl_Rect

11.147.3.14 request_shrink_b()

```
void Fl_Tile::request_shrink_b (
    int old_b,
    int & new_b,
    Fl_Rect * final_size) [protected]
```

Request for children to change their layout.

See also

[Fl_Tile::request_shrink_l\(int old_l, int &new_l, Fl_Rect *final_size\)](#)

Parameters

in	<i>old_b</i>	shrink all children with this current bottom edge toward the top edge of this tile
in, out	<i>new_b</i>	try to shrink to this coordinate, return the maximum possible shrinkage
in, out	<i>final_size</i>	if not NULL, write the new position and size of all affected children into this list of Fl_Rect

11.147.3.15 request_shrink_l()

```
void Fl_Tile::request_shrink_l (
    int old_l,
    int & new_l,
    Fl_Rect * final_size) [protected]
```

Request for children to change their layout.

`drag_intersection` requests that all children with the left edge at `old_l` to shrink to `new_l` towards the right side of the tile. If the child can not shrink by that amount, it will ask all other children that touch its right side to shrink by the remainder (recursion). `new_l` will return the the maximum possible value while maintaining minimum width for all children involved.

`request_shrink_r` asks children to shrink toward the left, so that their right edge is as close as possible to `new_r`. `request_shrink_t` and `request_shrink_b` provide the same functionality for vertical resizing.

Parameters

in	<i>old_l</i>	shrink all children with this current left edge
in, out	<i>new_l</i>	try to shrink to this coordinate, return the maximum possible shrinkage
in, out	<i>final_size</i>	if not NULL, write the new position and size of all affected children into this list of Fl_Rect

11.147.3.16 request_shrink_r()

```
void Fl_Tile::request_shrink_r (
    int old_r,
    int & new_r,
    Fl_Rect * final_size) [protected]
```

Request for children to change their layout.

See also

[Fl_Tile::request_shrink_l\(int old_l, int &new_l, Fl_Rect *final_size\)](#)

Parameters

in	<i>old_r</i>	shrink all children with this current right edge toward the left edge of this tile
in, out	<i>new_r</i>	try to shrink to this coordinate, return the maximum possible shrinkage
in, out	<i>final_size</i>	if not NULL, write the new position and size of all affected children into this list of Fl_Rect

11.147.3.17 request_shrink_t()

```
void Fl_Tile::request_shrink_t (
    int old_t,
    int & new_t,
    Fl_Rect * final_size) [protected]
```

Request for children to change their layout.

See also

[Fl_Tile::request_shrink_l\(int old_l, int &new_l, Fl_Rect *final_size\)](#)

Parameters

in	<i>old_t</i>	shrink all children with this current top edge toward the bottom edge of this tile
in, out	<i>new_t</i>	try to shrink to this coordinate, return the maximum possible shrinkage
in, out	<i>final_size</i>	if not NULL, write the new position and size of all affected children into this list of Fl_Rect

11.147.3.18 resize()

```
void Fl_Tile::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Resizes the [Fl_Tile](#) widget and its children.

[Fl_Tile](#) implements its own [resize\(\)](#) method. It does not use [Fl_Group::resize\(\)](#) to resize itself and its children.

In [size_range](#) mode, the child marked [resizable\(\)](#) is resized first. Only if its minimum size is reached, other widgets in the tile will resize too.

In classic mode or when no resizable child is set, enlarging works by moving the lower-right corner and resizing the bottom and right border widgets accordingly.

Shrinking the [Fl_Tile](#) works in the opposite way by shrinking the bottom and right border widgets, unless they are reduced to zero width or height, resp. or to their minimal sizes defined by the [resizable\(\)](#) widget. In this case other widgets will be shrunk as well.

See the [Fl_Tile](#) class documentation about how the [resizable\(\)](#) works.

Reimplemented from [Fl_Group](#).

11.147.3.19 set_cursor()

```
void Fl_Tile::set_cursor (
    int n) [protected]
```

Set one of four cursors used for dragging etc. . .

[Fl_Tile](#) uses an array of four cursors that are set depending on user actions:

- 0: normal cursor
- 1: horizontal dragging
- 2: vertical dragging
- 3: dragging an intersection

This method sets the window cursor for the given index *n*.

11.147.3.20 size_range() [1/2]

```
void Fl_Tile::size_range (
    Fl_Widget * w,
    int minw,
    int minh,
    int maxw = 0x7FFFFFFF,
    int maxh = 0x7FFFFFFF)
```

Set the allowed size range for the give child widget.

[Fl_Tile](#) currently supports only the minimal width and height setting.

Parameters

in	<i>w</i>	set the range for this child widget
in	<i>minw,minh</i>	minimum width and height for that child
in	<i>maxw,maxh</i>	maximum size, defaults to infinite, currently ignored

11.147.3.21 size_range() [2/2]

```
void Fl_Tile::size_range (
    int index,
    int minw,
    int minh,
    int maxw = 0x7FFFFFFF,
    int maxh = 0x7FFFFFFF)
```

Set the allowed size range for the child at the given index.

[Fl_Tile](#) currently supports only the minimal width and height setting.

Parameters

in	<i>index</i>	set the range for the child at this index
in	<i>minw,minh</i>	minimum width and height for that child
in	<i>maxw,maxh</i>	maximum size, defaults to infinite, currently ignored

The documentation for this class was generated from the following files:

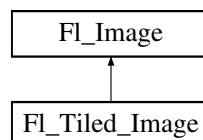
- [Fl_Tile.H](#)
- [Fl_Tile.cxx](#)

11.148 Fl_Tiled_Image Class Reference

This class supports tiling of images over a specified area.

```
#include <Fl_Tiled_Image.H>
```

Inheritance diagram for [Fl_Tiled_Image](#):

**Public Member Functions**

- void [color_average](#) ([Fl_Color](#) c, float i) [FL_OVERRIDE](#)

The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.

- `Fl_Image * copy ()` const
Creates a resized copy of the image.
- `Fl_Image * copy (int W, int H)` const `FL_OVERRIDE`
Creates a resized copy of the image.
- `void desaturate ()` `FL_OVERRIDE`
The `desaturate()` method converts an image to grayscale.
- `void draw (int X, int Y)`
- `void draw (int X, int Y, int W, int H, int cx=0, int cy=0)` `FL_OVERRIDE`
Draws a tiled image.
- `Fl_Tiled_Image (Fl_Image *i, int W=0, int H=0)`
The constructors create a new tiled image containing the specified image.
- `Fl_Image * image ()`
Gets The image that is tiled.
- `virtual ~Fl_Tiled_Image ()`
The destructor frees all memory and server resources that are used by the tiled image.

Public Member Functions inherited from `Fl_Image`

- `virtual class Fl_Shared_Image * as_shared_image ()`
Returns whether an image is an `Fl_Shared_Image` or not.
- `Fl_Image * copy ()` const
Creates a copy of the image in the same size.
- `int count ()` const
Returns the number of data values associated with the image.
- `int d ()` const
Returns the image depth.
- `const char *const * data ()` const
Returns a pointer to the current image data array.
- `int data_h ()` const
Returns the height of the image data.
- `int data_w ()` const
Returns the width of the image data.
- `void draw (int X, int Y)`
Draws the image to the current drawing surface.
- `int fail ()` const
Returns a value that is not 0 if there is currently no image available.
- `Fl_Image (int W, int H, int D)`
The constructor creates an empty image with the specified width, height, and depth.
- `int h ()` const
Returns the current image drawing height in FLTK units.
- `void inactive ()`
The `inactive()` method calls `color_average(FL_BACKGROUND_COLOR, 0.33f)` to produce an image that appears grayed out.
- `virtual void label (Fl_Menu_Item *m)`
This method is an obsolete way to set the image attribute of a menu item.
- `virtual void label (Fl_Widget *w)`
This method is an obsolete way to set the image attribute of a widget or menu item.
- `int ld ()` const
Returns the current line data size in bytes.
- `virtual void release ()`
Releases an `Fl_Image` - the same as `'delete this'`.
- `virtual void scale (int width, int height, int proportional=1, int can_expand=0)`

- *Sets the drawing size of the image.*
- virtual void `uncache` ()
If the image has been cached for display, delete the cache data.
- int `w` () const
Returns the current image drawing width in FLTK units.
- virtual `~FI_Image` ()
The destructor is a virtual method that frees all memory used by the image.

Protected Attributes

- int `alloc_image_`
- `FI_Image *` `image_`

Additional Inherited Members

Static Public Member Functions inherited from `FI_Image`

- static `FI_Labeltype` `define_FL_IMAGE_LABEL` ()
- static `FI_RGB_Scaling` `RGB_scaling` ()
Returns the currently used RGB image scaling method.
- static void `RGB_scaling` (`FI_RGB_Scaling`)
Sets the RGB image scaling method used for copy(int, int).
- static `FI_RGB_Scaling` `scaling_algorithm` ()
Gets what algorithm is used when resizing a source image to draw it.
- static void `scaling_algorithm` (`FI_RGB_Scaling` algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Static Public Attributes inherited from `FI_Image`

- static const int `ERR_FILE_ACCESS` = -2
- static const int `ERR_FORMAT` = -3
- static const int `ERR_MEMORY_ACCESS` = -4
- static const int `ERR_NO_IMAGE` = -1
- static bool `register_images_done` = false
True after `fl_register_images()` was called, false before.

Protected Member Functions inherited from `FI_Image`

- void `d` (int D)
Sets the current image depth.
- void `data` (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void `draw_empty` (int X, int Y)
The protected method `draw_empty()` draws a box with an X in it.
- int `draw_scaled` (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void `h` (int H)
Sets the height of the image data.
- void `ld` (int LD)
Sets the current line data size in bytes.
- void `w` (int W)
Sets the width of the image data.

Static Protected Member Functions inherited from [Fl_Image](#)

- static void **labeltype** (const [Fl_Label](#) *lo, int lx, int ly, int lw, int lh, [Fl_Align](#) la)
- static void **measure** (const [Fl_Label](#) *lo, int &lw, int &lh)

11.148.1 Detailed Description

This class supports tiling of images over a specified area.

The source (tile) image is **not** copied unless you call the [color_average\(\)](#), [desaturate\(\)](#), or [inactive\(\)](#) methods.

11.148.2 Constructor & Destructor Documentation

11.148.2.1 [Fl_Tiled_Image](#)()

```
Fl_Tiled_Image::Fl_Tiled_Image (
    Fl\_Image * i,
    int W = 0,
    int H = 0)
```

The constructors create a new tiled image containing the specified image.

Use a width and height of 0 to tile the whole window/widget.

Note

Due to implementation constraints in FLTK 1.3.3 and later width and height of 0 may not work as expected when used as background image in widgets other than windows. You may need to center and clip the image (label) and set the label type to [FL_NORMAL_LABEL](#). Doing so will let the tiled image fill the whole widget as its background image. Other combinations of label flags may or may not work.

```
#include "bg.xpm"
Fl_Pixmap *bg_xpm = new Fl\_Pixmap(bg_xpm);
Fl\_Tiled\_Image *bg_tiled = new Fl\_Tiled\_Image(bg_xpm, 0, 0);

Fl\_Box *box = new Fl\_Box(40, 40, 300, 100, "");
box->box(FL\_UP\_BOX);
box->labeltype(FL\_NORMAL\_LABEL);
box->align(FL\_ALIGN\_INSIDE | FL\_ALIGN\_CENTER | FL\_ALIGN\_CLIP);
box->image(bg_tiled);
```

Note

Setting an image (label) for a window may not work as expected due to implementation constraints in FLTK 1.3.x and maybe later. The reason is the way [Fl::scheme\(\)](#) initializes the window's label type and image. A possible workaround is to use another [Fl_Group](#) as the only child widget and to set the background image for this group as described above.

Todo Fix [Fl_Tiled_Image](#) as background image for widgets and windows and fix the implementation of [Fl::scheme\(const char *\)](#).

11.148.3 Member Function Documentation

11.148.3.1 [color_average\(\)](#)

```
void Fl\_Tiled\_Image::color\_average (
    Fl\_Color c,
    float i) [virtual]
```

The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.

The first argument specifies the FLTK color to be used.

The second argument specifies the amount of the original image to combine with the color, so a value of 1.0 results in no color blend, and a value of 0.0 results in a constant image of the specified color.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented from [Fl_Image](#).

11.148.3.2 copy()

```
Fl_Image * Fl_Tiled_Image::copy (
    int W,
    int H) const [virtual]
```

Creates a resized copy of the image.

It is recommended not to call this member function to reduce the size of an image to the size of the area where this image will be drawn, and to use [Fl_Image::scale\(\)](#) instead.

The new image should be released when you are done with it.

Note: since FLTK 1.4.0 you can use [Fl_Image::release\(\)](#) for all types of images (i.e. all subclasses of [Fl_Image](#)) instead of operator *delete* for [Fl_Image](#)'s and [Fl_Image::release\(\)](#) for [Fl_Shared_Image](#)'s.

The new image data will be converted to the requested size. RGB images are resized using the algorithm set by [Fl_Image::RGB_scaling\(\)](#).

For the new image the following equations are true:

- `w() == data_w() == W`
- `h() == data_h() == H`

Parameters

in	<i>W,H</i>	Requested width and height of the new image

Note

The returned image can be safely cast to the same image type as that of the source image provided this type is one of [Fl_RGB_Image](#), [Fl_SVG_Image](#), [Fl_Pixmap](#), [Fl_Bitmap](#), [Fl_Tiled_Image](#), [Fl_Anim_GIF_Image](#) and [Fl_Shared_Image](#). Returned objects copied from images of other, derived, image classes belong to the parent class appearing in this list. For example, the copy of an [Fl_GIF_Image](#) is an object of class [Fl_Pixmap](#).

Since FLTK 1.4.0 this method is 'const'. If you derive your own class from [Fl_Image](#) or any subclass your overridden methods of '[Fl_Image::copy\(\) const](#)' and '[Fl_Image::copy\(int, int\) const](#)' **must** also be 'const' for inheritance to work properly. This is different than in FLTK 1.3.x and earlier where these methods have not been 'const'.

Reimplemented from [Fl_Image](#).

11.148.3.3 desaturate()

```
void Fl_Tiled_Image::desaturate () [virtual]
```

The [desaturate\(\)](#) method converts an image to grayscale.

If the image contains an alpha channel (depth = 4), the alpha channel is preserved.

An internal copy is made of the original image data before changes are applied, to avoid modifying the original image data in memory.

Reimplemented from [Fl_Image](#).

11.148.3.4 draw()

```
void Fl_Tiled_Image::draw (
    int X,
    int Y,
    int W,
    int H,
    int cx = 0,
    int cy = 0) [virtual]
```

Draws a tiled image.

Tiled images can be used as background images for widgets and windows. However, due to implementation constraints, you must take care when setting label types and alignment flags. Only certain combinations work as expected, others may yield unexpected results and undefined behavior.

This draw method can draw multiple copies of one image in an area given by X, Y, W, H.

The optional arguments `cx` and `cy` can be used to crop the image starting at offsets (`cx`, `cy`). `cx` and `cy` must be ≥ 0 (negative values are ignored). If one of the values is greater than the image width or height resp. (`cx` $\geq$ `image()->w()` or `cy` $\geq$ `image()->h()`) nothing is drawn, because the resulting image would be empty.

After calculating the resulting image size the image is drawn as often as necessary to fill the given area, starting at the top left corner.

If both `w` and `h` are 0 the image is repeated as often as necessary to fill the entire window, unless there is a valid clip region. If you want to fill only one particular widget's background, then you should either set a clip region in your `draw()` method or use the label alignment flags `FL_ALIGN_INSIDE|FL_ALIGN_CLIP` to make sure the image is clipped.

This may be improved in a later version of the library.

Reimplemented from [Fl_Image](#).

The documentation for this class was generated from the following files:

- `Fl_Tiled_Image.H`
- `Fl_Tiled_Image.cxx`

11.149 Fl_Timeout Class Reference

The internal class [Fl_Timeout](#) handles all timeout related functions.

```
#include <Fl_Timeout.h>
```

Static Public Member Functions

- static void [add_timeout](#) (double time, [Fl_Timeout_Handler](#) cb, void *data)
Adds a one-shot timeout callback.
- static void [do_timeouts](#) ()
Elapse timers and call their callbacks if any timers are expired.
- static void [elapse_timeouts](#) ()
Elapse all timers w/o calling their callbacks.
- static int [has_timeout](#) ([Fl_Timeout_Handler](#) cb, void *data)
Returns true if the timeout exists and has not been called yet.
- static int [remove_next_timeout](#) ([Fl_Timeout_Handler](#) cb, void *data=NULL, void **data_return=NULL)
Remove the next matching timeout callback and return its data pointer.
- static void [remove_timeout](#) ([Fl_Timeout_Handler](#) cb, void *data)
Remove a timeout callback.
- static void [repeat_timeout](#) (double time, [Fl_Timeout_Handler](#) cb, void *data)
Repeats a timeout callback from the expiration of the previous timeout, allowing for more accurate timing.
- static double [time_to_wait](#) (double ttw)
Returns the delay in seconds until the next timer expires, limited by ttw.

Protected Member Functions

- double [delay](#) ()
Get the timer's delay in seconds.
- void [delay](#) (double t)
Set the timer's delay in seconds.
- void [insert](#) ()
Insert this timer entry into the active timer queue.
- void [make_current](#) ()
Remove the timeout from the active timer queue and push it onto the stack of currently running callbacks.
- void [release](#) ()
Remove the top-most timeout from the stack of currently running timeout callbacks and insert it into the list of free timers.

Static Protected Member Functions

- static [FI_Timeout](#) * [current](#) ()
Returns the first (top-most) timeout from the current timeout stack.
- static [FI_Timeout](#) * [get](#) (double time, [FI_Timeout_Handler](#) cb, void *data)
Get an [FI_Timeout](#) instance for further handling.

Protected Attributes

- [FI_Timeout_Handler](#) **callback**
- void * **data**
- [FI_Timeout](#) * **next**
- int **skip**
- double **time**

Static Protected Attributes

- static [FI_Timeout](#) * [current_timeout](#) = 0
The list of current timeouts is used to store the timeout whose callback is called while the callback is executed.
- static [FI_Timeout](#) * [first_timeout](#) = 0
List of active timeouts.
- static [FI_Timeout](#) * [free_timeout](#) = 0
List of free timeouts after use.

11.149.1 Detailed Description

The internal class [FI_Timeout](#) handles all timeout related functions.

All code is platform independent except retrieving a timestamp which requires calling a system driver function and potentially results in different timer resolutions (from milliseconds to microseconds).

Related user documentation:

- [FI_Timeout_Handler](#)
- [FI::add_timeout](#)(double time, [FI_Timeout_Handler](#) cb, void *data)
- [FI::repeat_timeout](#)(double time, [FI_Timeout_Handler](#) cb, void *data)
- [FI::has_timeout](#)([FI_Timeout_Handler](#) cb, void *data)
- [FI::remove_timeout](#)([FI_Timeout_Handler](#) cb, void *data)
- [FI::remove_next_timeout](#)([FI_Timeout_Handler](#) cb, void *data, void **data_return)

11.149.2 Member Function Documentation

11.149.2.1 add_timeout()

```
void FI_Timeout::add_timeout (
    double time,
    FI_Timeout_Handler cb,
    void * data) [static]
```

Adds a one-shot timeout callback.

The callback function `cb` will be called by [FI::wait\(\)](#) at `time` seconds after this function is called.

Parameters

in	<i>time</i>	delta time in seconds until the timer expires
in	<i>cb</i>	callback function
in	<i>data</i>	optional user data (default: NULL)

Implements:

```
void Fl::add_timeout(double time, Fl_Timeout_Handler cb, void *data)
```

See also

[Fl::add_timeout\(double time, Fl_Timeout_Handler cb, void *data\)](#)

11.149.2.2 current()

```
Fl_Timeout * Fl_Timeout::current () [static], [protected]
```

Returns the first (top-most) timeout from the current timeout stack.

This returns a pointer to the timeout but does not remove it from the list of current timeouts. This should be the timeout that is currently executing its callback.

Returns

`Fl_Timeout*` The current timeout whose callback is running.

Return values

<code>NULL</code>	if no callback is currently running.

11.149.2.3 elapse_timeouts()

```
void Fl_Timeout::elapse_timeouts () [static]
```

Elapse all timers w/o calling their callbacks.

All timer values are adjusted by the delta time since the last call. This method does **NOT** call timer callbacks if timers are expired.

This must be called before new timers are added to the timer queue to make sure that the next timer decrement does not count down too much time.

See also

[Fl_Timeout::do_timeouts\(\)](#)

11.149.2.4 get()

```
Fl_Timeout * Fl_Timeout::get (
    double time,
    Fl_Timeout_Handler cb,
    void * data) [static], [protected]
```

Get an [Fl_Timeout](#) instance for further handling.

The timer object will be initialized with the input parameters as given by [Fl::add_timeout\(\)](#) or [Fl::repeat_timeout\(\)](#).

[Fl_Timeout](#) objects are maintained in three queues:

- active timer queue
- list (stack, i.e. LIFO) of currently executing timer callbacks
- free timer entries.

When the FLTK program is launched all queues are empty. Whenever a new timer object is required the [get\(\)](#) method is called and a timer object is either found in the queue of free timer entries or a new timer object is created (operator new).

Active timer entries are inserted into the "active timer queue" until they expire and their callback is called.

Before the callback is called the timer entry is inserted into the list of current timers, i.e. it becomes the `Fl_Timeout::current()` timeout. This can be used in `Fl::repeat_timeout()` to find out if and how long the current timeout has been delayed.

When a timer is no longer used it is popped from the `current` list and inserted into the "free timer" list so it can be reused later.

Timer queue entries are never returned to the system, there's no garbage collection. The total number of timer objects is determined by the largest number of concurrently active timers.

Parameters

in	<i>time</i>	requested delta time
in	<i>cb</i>	timer callback
in	<i>data</i>	userdata for timer callback

Returns

Fl_Timeout* Timer entry

See also

[Fl::add_timeout\(\)](#), [Fl::repeat_timeout\(\)](#)

11.149.2.5 has_timeout()

```
int Fl_Timeout::has_timeout (
    Fl_Timeout_Handler cb,
    void * data) [static]
```

Returns true if the timeout exists and has not been called yet.

Parameters

in	<i>cb</i>	Timer callback (must match)
in	<i>data</i>	Callback user data (must match)

Returns

whether the timer was found in the queue

Return values

0	not found
1	found

Implements:

```
int Fl::has_timeout(Fl_Timeout_Handler cb, void *data)
```

See also

[Fl::has_timeout\(Fl_Timeout_Handler cb, void *data\)](#)

11.149.2.6 insert()

```
void Fl_Timeout::insert () [protected]
```

Insert this timer entry into the active timer queue.

The timer is inserted at the required position so the timer queue is always ordered by due time.

11.149.2.7 make_current()

```
void Fl_Timeout::make_current () [protected]
```

Remove the timeout from the active timer queue and push it onto the stack of currently running callbacks.

This becomes the [current\(\)](#) timeout which can be used in [Fl::repeat_timeout\(\)](#).

See also

[Fl_Timeout::current\(\)](#)

11.149.2.8 release()

```
void Fl_Timeout::release () [protected]
```

Remove the top-most timeout from the stack of currently running timeout callbacks and insert it into the list of free timers.

Typical code in the library would look like:

```
// The timeout \p Fl_Timeout *t has expired, run its callback
t->make_current();
(t->callback)(t->data);
t->release();
```

11.149.2.9 remove_next_timeout()

```
int Fl_Timeout::remove_next_timeout (
    Fl_Timeout_Handler cb,
    void * data = NULL,
    void ** data_return = NULL) [static]
```

Remove the next matching timeout callback and return its `data` pointer.

Implements:

```
int Fl::remove_next_timeout(Fl_Timeout_Handler cb, void *data, void **data_return)
```

Parameters

<code>in</code>	<code>cb</code>	Timer callback to be removed (must match)
<code>in</code>	<code>data</code>	Wildcard if NULL, must match otherwise
<code>in, out</code>	<code>data_return</code>	pointer to void * to receive the data value

Returns

non-zero if a timer was found and removed

Return values

<code>0</code>	no matching timer was found
<code>1</code>	the last matching timeout was found and removed
<code>N > 1</code>	a matching timeout was removed and there are (N - 1) matching timeouts pending

For details

See also

[Fl::remove_next_timeout\(Fl_Timeout_Handler cb, void *data, void **data_return\)](#)

11.149.2.10 remove_timeout()

```
void Fl_Timeout::remove_timeout (
    Fl_Timeout_Handler cb,
    void * data) [static]
```

Remove a timeout callback.

This method removes all matching timeouts, not just the first one. This may change in the future.

Parameters

in	<i>cb</i>	Timer callback to be removed (must match)
in	<i>data</i>	Wildcard if NULL, must match otherwise

Implements:

```
void Fl::remove_timeout(Fl_Timeout_Handler cb, void *data)
```

See also

[Fl::remove_timeout\(Fl_Timeout_Handler cb, void *data\)](#)

11.149.2.11 repeat_timeout()

```
void Fl_Timeout::repeat_timeout (
    double time,
    Fl_Timeout_Handler cb,
    void * data) [static]
```

Repeats a timeout callback from the expiration of the previous timeout, allowing for more accurate timing.

Parameters

in	<i>time</i>	delta time in seconds until the timer expires
in	<i>cb</i>	callback function
in	<i>data</i>	optional user data (default: NULL)

Implements:

```
void Fl::repeat_timeout(double time, Fl_Timeout_Handler cb, void *data)
```

See also

[Fl::repeat_timeout\(double time, Fl_Timeout_Handler cb, void *data\)](#)

11.149.2.12 time_to_wait()

```
double Fl_Timeout::time_to_wait (
    double ttw) [static]
```

Returns the delay in seconds until the next timer expires, limited by `ttw`.

This function calculates the time to wait for the FLTK event queue processing, depending on the given value `ttw`.

If at least one timer is active and its timeout value is smaller than `ttw` then this value is returned. [Fl::wait\(\)](#) will wait no longer than until the next timer expires.

If no timer is active this returns the input value `ttw` unchanged.

If at least one timer is expired this returns 0.0 so the event processing does not wait.

Parameters

in	<i>ttw</i>	time to wait from Fl::wait() etc. (upper limit)

Returns

delay until next timeout or 0.0 (see description)

11.149.3 Member Data Documentation

11.149.3.1 `current_timeout`

`Fl_Timeout * Fl_Timeout::current_timeout = 0 [static], [protected]`

The list of current timeouts is used to store the timeout whose callback is called while the callback is executed.

This is used like a stack, the current timeout is pushed to the front of the list and once the callback is finished, that timeout is removed and entered into the free list.

Background: `Fl::repeat_timeout()` needs to know which timeout triggered it and the exact schedule time and/or the delay of that timeout, i.e. how long the scheduled time was missed before the callback was called. A static, global variable is not sufficient since the user code can call other functions, e.g. dialogs, that run a nested event loop which can run another timeout callback. Hence this list of "current" timeouts is used like a stack (last in, first out).

See also

`Fl_Timeout::push()` Member function (method)

11.149.3.2 `first_timeout`

`Fl_Timeout * Fl_Timeout::first_timeout = 0 [static], [protected]`

List of active timeouts.

These timeouts can be triggered when due, which calls their callbacks. The lifetime of a timeout:

- active, in this queue
- callback running, in queue `current_timeout`
- done, in list of free timeouts, ready to be reused.

11.149.3.3 `free_timeout`

`Fl_Timeout * Fl_Timeout::free_timeout = 0 [static], [protected]`

List of free timeouts after use.

Timeouts can be reused many times.

The documentation for this class was generated from the following files:

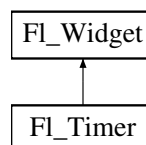
- [Fl_Timeout.h](#)
- [Fl_Timeout.cxx](#)

11.150 `Fl_Timer` Class Reference

This is provided only to emulate the Forms Timer widget.

```
#include <Fl_Timer.H>
```

Inheritance diagram for `Fl_Timer`:



Public Member Functions

- char `direction` () const
Gets or sets the direction of the timer.
- void `direction` (char d)
Gets or sets the direction of the timer.
- `Fl_Timer` (uchar t, int x, int y, int w, int h, const char *l)

- Creates a new *FL_Timer* widget using the given type, position, size, and label string.
- int **handle** (int) **FL_OVERRIDE**
Handles the specified event.
- char **suspended** () const
Gets or sets whether the timer is suspended.
- void **suspended** (char d)
Gets or sets whether the timer is suspended.
- double **value** () const
See void *FL_Timer::value(double)*
- void **value** (double)
Sets the current timer value.
- ~**FL_Timer** ()
Destroys the timer and removes the timeout.

Public Member Functions inherited from *FL_Widget*

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- *FL_Align* **align** () const
Gets the label alignment.
- void **align** (*FL_Align* alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class *FL_Gl_Window* * **as_gl_window** ()
Returns an *FL_Gl_Window* pointer if this widget is an *FL_Gl_Window*.
- virtual class *FL_Gl_Window* const * **as_gl_window** () const
- virtual *FL_Group* * **as_group** ()
Returns an *FL_Group* pointer if this widget is an *FL_Group*.
- virtual *FL_Group* const * **as_group** () const
- virtual *FL_Window* * **as_window** ()
Returns an *FL_Window* pointer if this widget is an *FL_Window*.
- virtual *FL_Window* const * **as_window** () const
- void **bind_deimage** (*FL_Image* *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (*FL_Image* *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- *FL_Boxtype* **box** () const
Gets the box type of the widget.

- void `box` (`FI_Boxtype` new_box)
Sets the box type for the widget.
- `FI_Callback_p` `callback` () const
Gets the current callback function for the widget.
- void `callback` (`FI_Callback` *cb)
Sets the current callback function for the widget.
- void `callback` (`FI_Callback` *cb, `FI_Callback_User_Data` *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void `callback` (`FI_Callback` *cb, void *p)
Sets the current callback function and data for the widget.
- void `callback` (`FI_Callback0` *cb)
Sets the current callback function for the widget.
- void `callback` (`FI_Callback1` *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int `changed` () const
Checks if the widget value changed since the last callback.
- void `clear_active` ()
Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()
Marks the value of the widget as unchanged.
- void `clear_damage` (`uchar` c=0)
Clears or sets the damage flags.
- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FI_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FI_Color` bg)
Sets the background color of the widget.
- void `color` (`FI_Color` bg, `FI_Color` sel)
Sets the background and selection color of the widget.
- `FI_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FI_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

- Internal use only.*

 - void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image &img`)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image *img`)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget *widget`, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget *widget`, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const

Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const

Gets the widget height.
- virtual void `hide` ()

Makes a widget invisible.
- int `horizontal_label_margin` ()

Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)

Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image` ()

Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image * image` () const

Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image &img`)

Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image *img`)

Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const

Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget *wgt`) const

Checks if this widget is a child of wgt.
- int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const

Gets the current label text.
- void `label` (const char *text)

Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.

- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FI_Color labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FI_Color](#) c)
Sets the label color.
- [FI_Font labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FI_Font](#) f)
Sets the font to use.
- [FI_Fonsize labelsiz](#)e () const
Gets the font size in pixels.
- void [labelsiz](#)e ([FI_Fonsize](#) pix)
Sets the font size in pixels.
- [FI_Labeltype labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FI_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FI_Group * parent](#) () const
Returns a pointer to the parent widget.
- void [parent](#) ([FI_Group](#) *p)
Internal use only - "for hacks only".
- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)
Changes the size or position of the widget.
- [FI_Color selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()

- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

Gets the current tooltip text.
- void [tooltip](#) (const char *text)

Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const

Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const

Gets the widget type.
- void [type](#) ([uchar](#) t)

Sets the widget type.
- int [use_accents_menu](#) ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const

Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)

Sets the user data for this widget.
- void [user_data](#) (void *v)

Sets the user data for this widget.
- int [vertical_label_margin](#) ()

Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)

Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const

Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const

Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)

Modifies keyboard focus navigation.
- int [visible_r](#) () const

Returns whether a widget and all its parents are visible.
- int [w](#) () const

Gets the widget width.
- [Fl_When](#) [when](#) () const

Returns the conditions under which the callback is called.

- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **FL_Window** * **window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual ~**FL_Widget** ()
Destroys the widget.

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
Draws the widget.

Protected Member Functions inherited from **FL_Widget**

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FL_Boxtype t, FL_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FL_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FL_Boxtype t, int x, int y, int w, int h, FL_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- **FL_Widget** (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.150.1 Detailed Description

This is provided only to emulate the Forms Timer widget.

It works by making a timeout callback every 1/5 second. This is wasteful and inaccurate if you just want something to happen a fixed time in the future. You should directly call [Fl::add_timeout\(\)](#) instead.

11.150.2 Constructor & Destructor Documentation

11.150.2.1 [Fl_Timer\(\)](#)

```
Fl_Timer::Fl_Timer (
    uchar t,
    int X,
    int Y,
    int W,
    int H,
    const char * l)
```

Creates a new [Fl_Timer](#) widget using the given type, position, size, and label string.

The type parameter can be any of the following symbolic constants:

- [FL_NORMAL_TIMER](#) - The timer just does the callback and displays the string "Timer" in the widget.
- [FL_VALUE_TIMER](#) - The timer does the callback and displays the current timer value in the widget.
- [FL_HIDDEN_TIMER](#) - The timer just does the callback and does not display anything.

11.150.3 Member Function Documentation

11.150.3.1 [direction\(\)](#) [1/2]

```
char Fl_Timer::direction () const [inline]
```

Gets or sets the direction of the timer.

If the direction is zero then the timer will count up, otherwise it will count down from the initial [value\(\)](#).

11.150.3.2 direction() [2/2]

```
void Fl_Timer::direction (
    char d) [inline]
```

Gets or sets the direction of the timer.

If the direction is zero then the timer will count up, otherwise it will count down from the initial [value\(\)](#).

11.150.3.3 draw()

```
void Fl_Timer::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                  // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.150.3.4 handle()

```
int Fl_Timer::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

11.150.3.5 suspended()

```
char Fl_Timer::suspended () const [inline]
```

Gets or sets whether the timer is suspended.

The documentation for this class was generated from the following files:

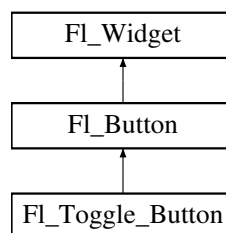
- Fl_Timer.H
- forms_timer.cxx

11.151 Fl_Toggle_Button Class Reference

The toggle button is a push button that needs to be clicked once to toggle on, and one more time to toggle off.

```
#include <Fl_Toggle_Button.H>
```

Inheritance diagram for Fl_Toggle_Button:



Public Member Functions

- [Fl_Toggle_Button](#) (int X, int Y, int W, int H, const char *l=0)
Creates a new [Fl_Toggle_Button](#) widget using the given position, size, and label string.

Public Member Functions inherited from [Fl_Button](#)

- int [clear](#) ()
Same as `value(0)`.
- [uchar compact](#) ()
Return true if buttons are rendered as compact buttons.
- void [compact](#) ([uchar](#) v)
Decide if buttons should be rendered in compact mode.
- [Fl_Boxtype down_box](#) () const
Returns the current down box type, which is drawn when `value()` is non-zero.
- void [down_box](#) ([Fl_Boxtype](#) b)
Sets the down box type.
- [Fl_Color down_color](#) () const
(for backwards compatibility)
- void [down_color](#) (unsigned c)
(for backwards compatibility)
- [Fl_Button](#) (int X, int Y, int W, int H, const char *L=0)
The constructor creates the button using the given position, size, and label.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- int [set](#) ()
Same as `value(1)`.
- void [setonly](#) ()
Turns on this button and turns off all other radio buttons in the group (calling `value(1)` or `set()` does not do this).

- int [shortcut](#) () const
Returns the current shortcut key for the button.
- void **shortcut** (const char *s)
(for backwards compatibility)
- void [shortcut](#) (int s)
Sets the shortcut key to s.
- char **value** () const
Returns the current value of the button (0 or 1).
- int [value](#) (int v)
Sets the current value of the button.

Public Member Functions inherited from [Fl_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [Fl_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([Fl_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()
Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * **as_gl_window** () const
- virtual [Fl_Group](#) * [as_group](#) ()
Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).
- virtual [Fl_Group](#) const * **as_group** () const
- virtual [Fl_Window](#) * [as_window](#) ()
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- virtual [Fl_Window](#) const * **as_window** () const
- void [bind_deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([Fl_Boxtype](#) new_box)
Sets the box type for the widget.
- [Fl_Callback_p](#) [callback](#) () const

- Gets the current callback function for the widget.*

 - void `callback` (`FL_Callback` *cb)
- Sets the current callback function for the widget.*

 - void `callback` (`FL_Callback` *cb, `FL_Callback_User_Data` *p, bool auto_free)
- Sets the current callback function and managed user data for the widget.*

 - void `callback` (`FL_Callback` *cb, void *p)
- Sets the current callback function and data for the widget.*

 - void `callback` (`FL_Callback0` *cb)
- Sets the current callback function for the widget.*

 - void `callback` (`FL_Callback1` *cb, long p=0)
- Sets the current callback function for the widget.*

 - unsigned int `changed` () const
- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()
- Marks the widget as inactive without sending events or changing focus.*

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*

 - void `clear_damage` (`uchar` c=0)
- Clears or sets the damage flags.*

 - void `clear_output` ()
- Sets a widget to accept input.*

 - void `clear_visible` ()
- Hides the widget.*

 - void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*

 - `FL_Color` `color` () const
- Gets the background color of the widget.*

 - void `color` (`FL_Color` bg)
- Sets the background color of the widget.*

 - void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*

 - `FL_Color` `color2` () const
- For back compatibility only.*

 - void `color2` (unsigned a)
- For back compatibility only.*

 - int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*

 - void `copy_label` (const char *new_label)
- Sets the current label.*

 - void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*

 - `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*

 - void `damage` (`uchar` c)
- Sets the damage bits for the widget.*

 - void `damage` (`uchar` c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int `damage_resize` (int, int, int, int)
- Internal use only.*

 - void `deactivate` ()
- Deactivates the widget.*

- [FL_Image](#) * [deimage](#) ()
Gets the image that is used as part of the widget label when in the inactive state.
- const [FL_Image](#) * [deimage](#) () const
Gets the image that is used as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the inactive state.
- void [deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- int [deimage_bound](#) () const
Returns whether the inactive image is managed by the widget.
- void [do_callback](#) ([FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with default arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const

- Gets the label color.*

 - void [labelcolor](#) ([Fl_Color](#) c)
- Sets the label color.*

 - [Fl_Font](#) [labelfont](#) () const
- Gets the font to use.*

 - void [labelfont](#) ([Fl_Font](#) f)
- Sets the font to use.*

 - [Fl_Fontsize](#) [labelsize](#) () const
- Gets the font size in pixels.*

 - void [labelsize](#) ([Fl_Fontsize](#) pix)
- Sets the font size in pixels.*

 - [Fl_Labeltype](#) [labeltype](#) () const
- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const
- Sets width ww and height hh accordingly with the label size.*

 - bool [needs_keyboard](#) () const
- Returns whether this widget needs a keyboard.*

 - void [needs_keyboard](#) (bool needs)
- Sets whether this widget needs a keyboard.*

 - unsigned int [output](#) () const
- Returns if a widget is used for output only.*

 - [Fl_Group](#) * [parent](#) () const
- Returns a pointer to the parent widget.*

 - void [parent](#) ([Fl_Group](#) *p)
- Internal use only - "for hacks only".*

 - void [position](#) (int X, int Y)
- Repositions the window or widget.*

 - void [redraw](#) ()
- Schedules the drawing of the widget.*

 - void [redraw_label](#) ()
- Schedules the drawing of the label.*

 - virtual void [resize](#) (int x, int y, int w, int h)
- Changes the size or position of the widget.*

 - [Fl_Color](#) [selection_color](#) () const
- Gets the selection color.*

 - void [selection_color](#) ([Fl_Color](#) a)
- Sets the selection color.*

 - void [set_active](#) ()
- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

 - void [set_output](#) ()
- Sets a widget to output only.*

 - void [set_visible](#) ()
- Makes the widget visible.*

 - void [set_visible_focus](#) ()
- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const

- Returns a pointer to the nearest parent window up the widget hierarchy.*
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~Fl_Widget** ()
Destroys the widget.

Additional Inherited Members

Static Public Member Functions inherited from **Fl_Widget**

- static void **default_callback** (Fl_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from **Fl_Widget**

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 , **NEEDS_KEYBOARD** = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 , **MAXIMIZED** = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }
flags possible values enumeration.

Protected Member Functions inherited from **Fl_Button**

- void **draw** () **FL_OVERRIDE**
Draws the widget.
- void **simulate_key_action** ()

Protected Member Functions inherited from **Fl_Widget**

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (Fl_Boxtype t, Fl_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (Fl_Boxtype t, int x, int y, int w, int h, Fl_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.

- void `draw_focus` () const
Draws a focus rectangle around the widget.
- void `draw_focus` (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void `draw_focus` (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Static Protected Member Functions inherited from FI_Button

- static void `key_release_timeout` (void *)

Static Protected Attributes inherited from FI_Button

- static FI_Widget_Tracker * `key_release_tracker` = 0

11.151.1 Detailed Description

The toggle button is a push button that needs to be clicked once to toggle on, and one more time to toggle off.

The `FI_Toggle_Button` subclass displays the "on" state by drawing a pushed-in button.

Buttons generate callbacks when they are clicked by the user. You control exactly when and how by changing the values for `type()` and `when()`.

11.151.2 Constructor & Destructor Documentation

11.151.2.1 FI_Toggle_Button()

```
FI_Toggle_Button::FI_Toggle_Button (
    int X,
    int Y,
    int W,
    int H,
    const char * L = 0)
```

Creates a new `FI_Toggle_Button` widget using the given position, size, and label string.

The constructor creates the button using the given position, size, and label.

The inherited destructor deletes the toggle button.

The Button `type()` is set to `FL_TOGGLE_BUTTON`.

Parameters

in	<i>X,Y,W,H</i>	position and size of the widget
in	<i>L</i>	widget label, default is no label

The documentation for this class was generated from the following files:

- Fl_Toggle_Button.H
- Fl_Button.cxx

11.152 Fl_Tooltip Class Reference

The [Fl_Tooltip](#) class provides tooltip support for all FLTK widgets.

```
#include <Fl_Tooltip.H>
```

Static Public Member Functions

- static [Fl_Color](#) [color](#) ()
Gets the background color for tooltips.
- static void [color](#) ([Fl_Color](#) c)
Sets the background color for tooltips.
- static [Fl_Widget](#) * [current](#) ()
Gets the current widget target.
- static void [current](#) ([Fl_Widget](#) *)
Sets the current widget target.
- static [Fl_Window](#) * [current_window](#) (void)
Returns the window that is used for tooltips.
- static float [delay](#) ()
Gets the tooltip delay.
- static void [delay](#) (float f)
Sets the tooltip delay.
- static void [disable](#) ()
Same as enable(0), disables tooltips on all widgets.
- static void [enable](#) (int b=1)
Enables tooltips on all widgets (or disables if b is false).
- static int [enabled](#) ()
Returns non-zero if tooltips are enabled.
- static void [enter_area](#) ([Fl_Widget](#) *w, int X, int Y, int W, int H, const char *tip)
You may be able to use this to provide tooltips for internal pieces of your widget.
- static [Fl_Font](#) [font](#) ()
Gets the typeface for the tooltip text.
- static void [font](#) ([Fl_Font](#) i)
Sets the typeface for the tooltip text.
- static float [hidedelay](#) ()
Gets the time until an open tooltip hides again.
- static void [hidedelay](#) (float f)
Sets the time until an open tooltip hides again.
- static float [hoverdelay](#) ()
Gets the tooltip hover delay, the delay between tooltips.
- static void [hoverdelay](#) (float f)
Sets the tooltip hover delay, the delay between tooltips.
- static int [margin_height](#) ()

- Gets the amount of extra space above and below the tooltip's text.*
 - static void `margin_height` (int v)
 - Sets the amount of extra space above and below the tooltip's text.*
- static int `margin_width` ()
 - Gets the amount of extra space left/right of the tooltip's text.*
- static void `margin_width` (int v)
 - Sets the amount of extra space left/right of the tooltip's text.*
- static `Fl_Fontsize` `size` ()
 - Gets the size of the tooltip text.*
- static void `size` (`Fl_Fontsize` s)
 - Sets the size of the tooltip text.*
- static `Fl_Color` `textcolor` ()
 - Gets the color of the text in the tooltip.*
- static void `textcolor` (`Fl_Color` c)
 - Sets the color of the text in the tooltip.*
- static int `wrap_width` ()
 - Gets the maximum width for tooltip's text before it word wraps.*
- static void `wrap_width` (int v)
 - Sets the maximum width for tooltip's text before it word wraps.*

Static Public Attributes

- static void(* `enter`)(`Fl_Widget` *w) = nothing
- static void(* `exit`)(`Fl_Widget` *w) = nothing

Friends

- class `Fl_TooltipBox`
- void `Fl_Widget::copy_tooltip` (const char *)
- void `Fl_Widget::tooltip` (const char *)

11.152.1 Detailed Description

The `Fl_Tooltip` class provides tooltip support for all FLTK widgets. It contains only static methods.

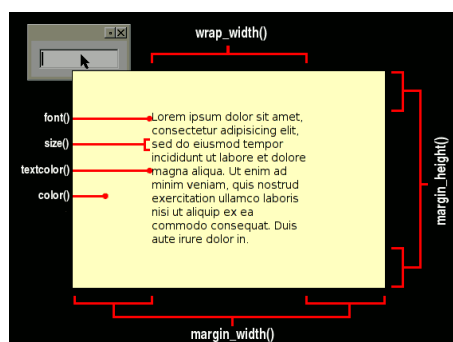


Figure 11.61 `Fl_Tooltip` Options

11.152.2 Member Function Documentation

11.152.2.1 `color()` [1/2]

```
static Fl_Color Fl_Tooltip::color () [inline], [static]
```

Gets the background color for tooltips.

The default background color is a pale yellow.

11.152.2.2 color() [2/2]

```
static void Fl_Tooltip::color (
    Fl_Color c) [inline], [static]
```

Sets the background color for tooltips.

The default background color is a pale yellow.

11.152.2.3 current()

```
void Fl_Tooltip::current (
    Fl_Widget * w) [static]
```

Sets the current widget target.

Acts as though `enter(widget)` was done but does not pop up a tooltip. This is useful to prevent a tooltip from reappearing when a modal overlapping window is deleted. FLTK does this automatically when you click the mouse button.

11.152.2.4 delay() [1/2]

```
static float Fl_Tooltip::delay () [inline], [static]
```

Gets the tooltip delay.

The default delay is 1.0 seconds.

11.152.2.5 delay() [2/2]

```
static void Fl_Tooltip::delay (
    float f) [inline], [static]
```

Sets the tooltip delay.

The default delay is 1.0 seconds.

11.152.2.6 disable()

```
static void Fl_Tooltip::disable () [inline], [static]
```

Same as `enable(0)`, disables tooltips on all widgets.

11.152.2.7 enable()

```
static void Fl_Tooltip::enable (
    int b = 1) [inline], [static]
```

Enables tooltips on all widgets (or disables if *b* is false).

11.152.2.8 enabled()

```
static int Fl_Tooltip::enabled () [inline], [static]
```

Returns non-zero if tooltips are enabled.

11.152.2.9 enter_area()

```
void Fl_Tooltip::enter_area (
    Fl_Widget * wid,
    int x,
    int y,
```

```
int w,
int h,
const char * t) [static]
```

You may be able to use this to provide tooltips for internal pieces of your widget.

Call this after setting [Fl::belowmouse\(\)](#) to your widget (because that calls the above `enter()` method). Then figure out what thing the mouse is pointing at, and call this with the widget (this pointer is used to remove the tooltip if the widget is deleted or hidden, and to locate the tooltip), the rectangle surrounding the area, relative to the top-left corner of the widget (used to calculate where to put the tooltip), and the text of the tooltip (which must be a pointer to static data as it is not copied).

11.152.2.10 font() [1/2]

```
static Fl_Font Fl_Tooltip::font () [inline], [static]
```

Gets the typeface for the tooltip text.

11.152.2.11 font() [2/2]

```
static void Fl_Tooltip::font (
    Fl_Font i) [inline], [static]
```

Sets the typeface for the tooltip text.

11.152.2.12 hidedelay() [1/2]

```
static float Fl_Tooltip::hidedelay () [inline], [static]
```

Gets the time until an open tooltip hides again.

The default delay is 12.0 seconds.

11.152.2.13 hidedelay() [2/2]

```
static void Fl_Tooltip::hidedelay (
    float f) [inline], [static]
```

Sets the time until an open tooltip hides again.

The default delay is 12.0 seconds.

11.152.2.14 hoverdelay() [1/2]

```
static float Fl_Tooltip::hoverdelay () [inline], [static]
```

Gets the tooltip hover delay, the delay between tooltips.

The default delay is 0.2 seconds.

11.152.2.15 hoverdelay() [2/2]

```
static void Fl_Tooltip::hoverdelay (
    float f) [inline], [static]
```

Sets the tooltip hover delay, the delay between tooltips.

The default delay is 0.2 seconds.

11.152.2.16 margin_height() [1/2]

```
static int Fl_Tooltip::margin_height () [inline], [static]
```

Gets the amount of extra space above and below the tooltip's text.

Default is 3.

11.152.2.17 margin_height() [2/2]

```
static void Fl_Tooltip::margin_height (  
    int v) [inline], [static]
```

Sets the amount of extra space above and below the tooltip's text.
Default is 3.

11.152.2.18 margin_width() [1/2]

```
static int Fl_Tooltip::margin_width () [inline], [static]
```

Gets the amount of extra space left/right of the tooltip's text.
Default is 3.

11.152.2.19 margin_width() [2/2]

```
static void Fl_Tooltip::margin_width (  
    int v) [inline], [static]
```

Sets the amount of extra space left/right of the tooltip's text.
Default is 3.

11.152.2.20 size() [1/2]

```
static Fl_Fontsize Fl_Tooltip::size () [inline], [static]
```

Gets the size of the tooltip text.

11.152.2.21 size() [2/2]

```
static void Fl_Tooltip::size (  
    Fl_Fontsize s) [inline], [static]
```

Sets the size of the tooltip text.

11.152.2.22 textcolor() [1/2]

```
static Fl_Color Fl_Tooltip::textcolor () [inline], [static]
```

Gets the color of the text in the tooltip.
The default is black.

11.152.2.23 textcolor() [2/2]

```
static void Fl_Tooltip::textcolor (  
    Fl_Color c) [inline], [static]
```

Sets the color of the text in the tooltip.
The default is black.

11.152.2.24 wrap_width() [1/2]

```
static int Fl_Tooltip::wrap_width () [inline], [static]
```

Gets the maximum width for tooltip's text before it word wraps.
Default is 400.

11.152.2.25 wrap_width() [2/2]

```
static void Fl_Tooltip::wrap_width (  
    int v) [inline], [static]
```

Sets the maximum width for tooltip's text before it word wraps.
Default is 400.

The documentation for this class was generated from the following files:

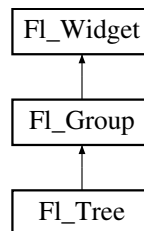
- [Fl_Tooltip.H](#)
- [Fl.cxx](#)
- [Fl_Tooltip.cxx](#)

11.153 Fl_Tree Class Reference

Tree widget.

```
#include <Fl_Tree.H>
```

Inheritance diagram for Fl_Tree:



Public Member Functions

- [Fl_Tree_Item](#) * [add](#) (const char *path, [Fl_Tree_Item](#) *newitem=0)
Adds a new item, given a menu style 'path'.
- [Fl_Tree_Item](#) * [add](#) ([Fl_Tree_Item](#) *parent_item, const char *name)
Add a new child item labeled 'name' to the specified 'parent_item'.
- void [calc_dimensions](#) ()
Recalculate widget dimensions and scrollbar visibility, normally managed automatically.
- void [calc_tree](#) ()
Recalculates the tree's sizes and scrollbar visibility, normally managed automatically.
- [Fl_Tree_Item](#) * [callback_item](#) ()
Gets the item that caused the callback.
- void [callback_item](#) ([Fl_Tree_Item](#) *item)
Sets the item that was changed for this callback.
- [Fl_Tree_Reason](#) [callback_reason](#) () const
Gets the reason for this callback.
- void [callback_reason](#) ([Fl_Tree_Reason](#) reason)
Sets the reason for this callback.
- void [clear](#) ()
Clear the entire tree's children, including the root.
- void [clear_children](#) ([Fl_Tree_Item](#) *item)
Clear all the children for 'item'.
- int [close](#) (const char *path, int docallback=1)
Closes the item specified by 'path'.
- int [close](#) ([Fl_Tree_Item](#) *item, int docallback=1)
Closes the specified 'item'.
- [Fl_Image](#) * [closeicon](#) () const
Returns the icon to be used as the 'close' icon.
- void [closeicon](#) ([Fl_Image](#) *val)
Sets the icon to be used as the 'close' icon.
- [Fl_Color](#) [connectorcolor](#) () const
Get the connector color used for tree connection lines.

- void **connectorcolor** (FL_Color val)
Set the connector color used for tree connection lines.
- FL_Tree_Connector **connectorstyle** () const
Returns the line drawing style for inter-connecting items.
- void **connectorstyle** (FL_Tree_Connector val)
Sets the line drawing style for inter-connecting items.
- int **connectorwidth** () const
Gets the width of the horizontal connection lines (in pixels) that appear to the left of each tree item's label.
- void **connectorwidth** (int val)
Sets the width of the horizontal connection lines (in pixels) that appear to the left of each tree item's label.
- int **deselect** (const char *path, int docallback=1)
Deselect an item specified by 'path'.
- int **deselect** (FL_Tree_Item *item, int docallback=1)
Deselect the specified item.
- int **deselect_all** (FL_Tree_Item *item=0, int docallback=1)
Deselect 'item' and all its children.
- void **display** (FL_Tree_Item *item)
Displays 'item', scrolling the tree as necessary.
- int **displayed** (FL_Tree_Item *item)
See if 'item' is currently displayed on-screen (visible within the widget).
- int **extend_selection** (FL_Tree_Item *from, FL_Tree_Item *to, int val=1, bool visible=false)
Extend a selection between 'from' and 'to' depending on 'visible'.
- int **extend_selection_dir** (FL_Tree_Item *from, FL_Tree_Item *to, int dir, int val, bool visible)
Extend the selection between and including 'from' and 'to' depending on direction 'dir', 'val', and 'visible'.
- FL_Tree_Item * **find_clicked** (int yonly=0)
Non-const version of FL_Tree::find_clicked(int yonly) const.
- const FL_Tree_Item * **find_clicked** (int yonly=0) const
Find the item that was last clicked on.
- FL_Tree_Item * **find_item** (const char *path)
*Non-const version of FL_Tree::find_item(const char *path) const.*
- const FL_Tree_Item * **find_item** (const char *path) const
Find the item, given a menu style path, e.g.
- FL_Tree_Item * **first** ()
Returns the first item in the tree, or 0 if none.
- FL_Tree_Item * **first_selected_item** ()
Returns the first selected item in the tree.
- FL_Tree_Item * **first_visible** ()
Returns the first open(), visible item in the tree, or 0 if none.
- FL_Tree_Item * **first_visible_item** ()
Returns the first open(), visible item in the tree, or 0 if none.
- FL_Tree (int X, int Y, int W, int H, const char *L=0)
Constructor.
- FL_Tree_Item * **get_item_focus** () const
Get the item that currently has keyboard focus.
- int **get_selected_items** (FL_Tree_Item_Array &items)
Returns the currently selected items as an array of 'ret_items'.
- int **handle** (int e) FL_OVERRIDE
Standard FLTK event handler for this widget.
- int **hposition** () const
Returns the horizontal scroll position as a pixel offset.

- void [hposition](#) (int pos)
Sets the horizontal scroll offset to position 'pos'.
- [FI_Tree_Item](#) * [insert](#) ([FI_Tree_Item](#) *item, const char *name, int pos)
Insert a new item 'name' into 'item's children at position 'pos'.
- [FI_Tree_Item](#) * [insert_above](#) ([FI_Tree_Item](#) *above, const char *name)
Inserts a new item 'name' above the specified [FI_Tree_Item](#) 'above'.
- int [is_close](#) (const char *path) const
See if item specified by 'path' is closed.
- int [is_close](#) ([FI_Tree_Item](#) *item) const
See if the specified 'item' is closed.
- int [is_hscroll_visible](#) () const
See if the horizontal scrollbar is currently visible.
- int [is_open](#) (const char *path) const
See if item specified by 'path' is open.
- int [is_open](#) ([FI_Tree_Item](#) *item) const
See if 'item' is open.
- int [is_scrollbar](#) ([FI_Widget](#) *w)
See if widget 'w' is one of the [FI_Tree](#) widget's scrollbars.
- int [is_selected](#) (const char *path)
See if item specified by 'path' is selected.
- int [is_selected](#) ([FI_Tree_Item](#) *item) const
See if the specified 'item' is selected.
- int [is_vscroll_visible](#) () const
See if the vertical scrollbar is currently visible.
- [FI_Tree_Item](#) * [item_clicked](#) ()
Return the item that was last clicked.
- [FI_Tree_Item_Draw_Mode](#) [item_draw_mode](#) () const
Get the 'item draw mode' used for the tree.
- void [item_draw_mode](#) ([FI_Tree_Item_Draw_Mode](#) mode)
Set the 'item draw mode' used for the tree to 'mode'.
- void [item_draw_mode](#) (int mode)
Set the 'item draw mode' used for the tree to integer 'mode'.
- void [item_labelbgcolor](#) ([FI_Color](#) val)
Set the default label background color used for creating new items.
- [FI_Color](#) [item_labelbgcolor](#) (void) const
Get the default label background color used for creating new items.
- void [item_labelfgcolor](#) ([FI_Color](#) val)
Set the default label foreground color used for creating new items.
- [FI_Color](#) [item_labelfgcolor](#) (void) const
Get the default label foreground color used for creating new items.
- [FI_Font](#) [item_labelfont](#) () const
Get the default font face used for creating new items.
- void [item_labelfont](#) ([FI_Font](#) val)
Set the default font face used for creating new items.
- [FI_Fontsize](#) [item_labelsize](#) () const
Get the default label fontsize used for creating new items.
- void [item_labelsize](#) ([FI_Fontsize](#) val)
Set the default label font size used for creating new items.
- int [item_pathname](#) (char *pathname, int pathnamelen, const [FI_Tree_Item](#) *item) const
Return 'pathname' of size 'pathnamelen' for the specified 'item'.
- [FI_Tree_Item_Reselect_Mode](#) [item_reselect_mode](#) () const

- Returns the current item re/selection mode.*

 - void **item_reselect_mode** ([Fl_Tree_Item_Reselect_Mode](#) mode)

Sets the item re/selection mode.
- int **labelmarginleft** () const

Get the amount of white space (in pixels) that should appear to the left of the label text.
- void **labelmarginleft** (int val)

Set the amount of white space (in pixels) that should appear to the left of the label text.
- [Fl_Tree_Item](#) * **last** ()

Returns the last item in the tree.
- [Fl_Tree_Item](#) * **last_selected_item** ()

Returns the last selected item in the tree.
- [Fl_Tree_Item](#) * **last_visible** ()

Returns the last [open\(\)](#), visible item in the tree.
- [Fl_Tree_Item](#) * **last_visible_item** ()

Returns the last [open\(\)](#), visible item in the tree.
- int **linespacing** () const

Get the amount of white space (in pixels) that should appear between items in the tree.
- void **linespacing** (int val)

Sets the amount of white space (in pixels) that should appear between items in the tree.
- void **load** (class [Fl_Preferences](#) &)

Load FLTK preferences.
- int **marginbottom** () const

Get the amount of white space (in pixels) that should appear below the last visible item when the vertical scroller is scrolled to the bottom.
- void **marginbottom** (int val)

Sets the amount of white space (in pixels) that should appear below the last visible item when the vertical scroller is scrolled to the bottom.
- int **marginleft** () const

Get the amount of white space (in pixels) that should appear between the widget's left border and the tree's contents.
- void **marginleft** (int val)

Set the amount of white space (in pixels) that should appear between the widget's left border and the left side of the tree's contents.
- int **margintop** () const

Get the amount of white space (in pixels) that should appear between the widget's top border and the top of the tree's contents.
- void **margintop** (int val)

Sets the amount of white space (in pixels) that should appear between the widget's top border and the top of the tree's contents.
- [Fl_Tree_Item](#) * **next** ([Fl_Tree_Item](#) *item=0)

Return the next item after 'item', or 0 if no more items.
- [Fl_Tree_Item](#) * **next_item** ([Fl_Tree_Item](#) *item, int dir=[FL_Down](#), bool visible=false)

Returns next item after 'item' in direction 'dir' depending on 'visible'.
- [Fl_Tree_Item](#) * **next_selected_item** ([Fl_Tree_Item](#) *item=0, int dir=[FL_Down](#))

Returns the next selected item above or below 'item', depending on 'dir'.
- [Fl_Tree_Item](#) * **next_visible_item** ([Fl_Tree_Item](#) *start, int dir)

Returns next [open\(\)](#), visible item above (dir==[FL_Up](#)) or below (dir==[FL_Down](#)) the specified 'item', or 0 if no more items.
- int **open** (const char *path, int docallback=1)

Opens the item specified by 'path'.
- int **open** ([Fl_Tree_Item](#) *item, int docallback=1)

Open the specified 'item'.
- void **open_toggle** ([Fl_Tree_Item](#) *item, int docallback=1)

- Toggle the open state of 'item'.*
- int **openchild_marginbottom** () const
Get the amount of white space (in pixels) that should appear below an open child tree's contents.
- void **openchild_marginbottom** (int val)
Set the amount of white space (in pixels) that should appear below an open child tree's contents.
- [Fl_Image](#) * **openicon** () const
Returns the icon to be used as the 'open' icon.
- void **openicon** ([Fl_Image](#) *val)
Sets the icon to be used as the 'open' icon.
- const [Fl_Tree_Prefs](#) & **prefs** () const
- [Fl_Tree_Item](#) * **prev** ([Fl_Tree_Item](#) *item=0)
Return the previous item before 'item', or 0 if no more items.
- void **recalc_tree** ()
Schedule tree to recalc the entire tree size.
- int **remove** ([Fl_Tree_Item](#) *item)
Remove the specified 'item' from the tree.
- void **resize** (int, int, int, int) **FL_OVERRIDE**
Resizes the [Fl_Group](#) widget and all of its children.
- [Fl_Tree_Item](#) * **root** ()
Returns the root item.
- void **root** ([Fl_Tree_Item](#) *newitem)
Sets the root item to 'newitem'.
- void **root_label** (const char *new_label)
Set the label for the root item to 'new_label'.
- int **scrollbar_size** () const
Gets the default size of scrollbars' troughs for this widget in pixels.
- void **scrollbar_size** (int size)
Sets the pixel size of the scrollbars' troughs to 'size' for this widget, in pixels.
- int **select** (const char *path, int docallback=1)
Select the item specified by 'path'.
- int **select** ([Fl_Tree_Item](#) *item, int docallback=1)
Select the specified 'item'.
- int **select_all** ([Fl_Tree_Item](#) *item=0, int docallback=1)
Select 'item' and all its children.
- int **select_only** ([Fl_Tree_Item](#) *selitem, int docallback=1)
Select only the specified item, deselecting all others that might be selected.
- void **select_toggle** ([Fl_Tree_Item](#) *item, int docallback=1)
Toggle the select state of the specified 'item'.
- [Fl_Boxtype](#) **selectbox** () const
Sets the style of box used to draw selected items.
- void **selectbox** ([Fl_Boxtype](#) val)
Gets the style of box used to draw selected items.
- [Fl_Tree_Select](#) **selectmode** () const
Gets the tree's current selection mode.
- void **selectmode** ([Fl_Tree_Select](#) val)
Sets the tree's selection mode.
- void **set_item_focus** ([Fl_Tree_Item](#) *item)
Set the item that currently should have keyboard focus.
- void **show_item** ([Fl_Tree_Item](#) *item)
Adjust the vertical scrollbar to show 'item' at the top of the display IF it is currently off-screen (for instance [show_item_top\(\)](#)).

- void [show_item](#) ([FI_Tree_Item](#) *item, int yoff)
Adjust the vertical scrollbar so that 'item' is visible 'yoff' pixels from the top of the [FI_Tree](#) widget's display.
- void [show_item_bottom](#) ([FI_Tree_Item](#) *item)
Adjust the vertical scrollbar so that 'item' is at the bottom of the display.
- void [show_item_middle](#) ([FI_Tree_Item](#) *item)
Adjust the vertical scrollbar so that 'item' is in the middle of the display.
- void [show_item_top](#) ([FI_Tree_Item](#) *item)
Adjust the vertical scrollbar so that 'item' is at the top of the display.
- void [show_self](#) ()
Print the tree as 'ascii art' to stdout.
- int [showcollapse](#) () const
Returns 1 if the collapse icon is enabled, 0 if not.
- void [showcollapse](#) (int val)
Set if we should show the collapse icon or not.
- int [showroot](#) () const
Returns 1 if the root item is to be shown, or 0 if not.
- void [showroot](#) (int val)
Set if the root item should be shown or not.
- [FI_Tree_Sort](#) sortorder () const
Set the default sort order used when items are added to the tree.
- void [sortorder](#) ([FI_Tree_Sort](#) val)
Gets the sort order used to add items to the tree.
- [FI_Image](#) * [usericon](#) () const
Returns the [FI_Image](#) being used as the default user icon for all newly created items.
- void [usericon](#) ([FI_Image](#) *val)
Sets the [FI_Image](#) to be used as the default user icon for all newly created items.
- int [usericonmarginleft](#) () const
Get the amount of white space (in pixels) that should appear to the left of the usericon.
- void [usericonmarginleft](#) (int val)
Set the amount of white space (in pixels) that should appear to the left of the usericon.
- int [vposition](#) () const
Returns the vertical scroll position as a pixel offset.
- void [vposition](#) (int pos)
Sets the vertical scroll offset to position 'pos'.
- int [widgetmarginleft](#) () const
Get the amount of white space (in pixels) that should appear to the left of the child fltk widget (if any).
- void [widgetmarginleft](#) (int val)
Set the amount of white space (in pixels) that should appear to the left of the child fltk widget (if any).
- ~[FI_Tree](#) ()
Destructor.

Public Member Functions inherited from [FI_Group](#)

- [FI_Widget](#) *& [_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void [add](#) ([FI_Widget](#) &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void [add](#) ([FI_Widget](#) *o)
See void [FI_Group::add\(FI_Widget &w\)](#)
- void [add_resizable](#) ([FI_Widget](#) &o)
Adds a widget to the group and makes it the resizable widget.

- `FL_Widget *const * array () const`
Returns a pointer to the internal array of children.
- `FL_Group const * as_group () const FL_OVERRIDE`
- `FL_Group * as_group () FL_OVERRIDE`
Returns an `FL_Group` pointer if this widget is an `FL_Group`.
- `void begin ()`
Sets the current group so you can build the widget tree by just constructing the widgets.
- `FL_Widget * child (int n) const`
Returns the *n*'th child.
- `int children () const`
Returns how many child widgets the group has.
- `void clear ()`
Deletes all child widgets from memory recursively.
- `unsigned int clip_children ()`
Returns the current clipping mode.
- `void clip_children (int c)`
Controls whether the group widget clips the drawing of child widgets to its bounding box.
- `virtual int delete_child (int n)`
Removes the widget at *index* from the group and deletes it.
- `void end ()`
Exactly the same as `current(this->parent())`.
- `int find (const FL_Widget &o) const`
See `int FL_Group::find(const FL_Widget *w) const`.
- `int find (const FL_Widget *) const`
Searches the children for the widget and returns the index.
- `FL_Group (int, int, int, int, const char * = 0)`
Creates a new `FL_Group` widget using the given position, size, and label string.
- `void focus (FL_Widget *W)`
- `void forms_end ()`
This is for forms compatibility only.
- `void init_sizes ()`
Resets the internal array of widget sizes and positions.
- `void insert (FL_Widget &, int i)`
The widget is removed from its current group (if any) and then inserted into this group.
- `void insert (FL_Widget &o, FL_Widget *before)`
This does `insert(w, find(before))`.
- `void remove (FL_Widget &)`
Removes a widget from the group but does not delete it.
- `void remove (FL_Widget *o)`
Removes the widget *o* from the group.
- `void remove (int index)`
Removes the widget at *index* from the group but does not delete it.
- `FL_Widget * resizable () const`
Returns the group's resizable widget.
- `void resizable (FL_Widget &o)`
Sets the group's resizable widget.
- `void resizable (FL_Widget *o)`
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
- `virtual ~FL_Group ()`
The destructor also deletes all the children.

Public Member Functions inherited from [FL_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FL_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FL_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FL_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FL_Gl_Window](#) pointer if this widget is an [FL_Gl_Window](#).
- virtual class [FL_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FL_Window](#) * [as_window](#) ()
Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).
- virtual [FL_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FL_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FL_Boxtype](#) new_box)
Sets the box type for the widget.
- [FL_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FL_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FL_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FL_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()

- Marks the widget as inactive without sending events or changing focus.*

 - void `clear_changed` ()
- Marks the value of the widget as unchanged.*

 - void `clear_damage` (uchar c=0)
- Clears or sets the damage flags.*

 - void `clear_output` ()
- Sets a widget to accept input.*

 - void `clear_visible` ()
- Hides the widget.*

 - void `clear_visible_focus` ()
- Disables keyboard focus navigation with this widget.*

 - `FL_Color` `color` () const
- Gets the background color of the widget.*

 - void `color` (`FL_Color` bg)
- Sets the background color of the widget.*

 - void `color` (`FL_Color` bg, `FL_Color` sel)
- Sets the background and selection color of the widget.*

 - `FL_Color` `color2` () const
- For back compatibility only.*

 - void `color2` (unsigned a)
- For back compatibility only.*

 - int `contains` (const `FL_Widget` *w) const
- Checks if w is a child of this widget.*

 - void `copy_label` (const char *new_label)
- Sets the current label.*

 - void `copy_tooltip` (const char *text)
- Sets the current tooltip text.*

 - `uchar` `damage` () const
- Returns non-zero if `draw()` needs to be called.*

 - void `damage` (uchar c)
- Sets the damage bits for the widget.*

 - void `damage` (uchar c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*

 - int `damage_resize` (int, int, int, int)
- Internal use only.*

 - void `deactivate` ()
- Deactivates the widget.*

 - `FL_Image` * `deimage` ()
- Gets the image that is used as part of the widget label when in the inactive state.*

 - const `FL_Image` * `deimage` () const
- Gets the image that is used as part of the widget label when in the inactive state.*

 - void `deimage` (`FL_Image` &img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - void `deimage` (`FL_Image` *img)
- Sets the image to use as part of the widget label when in the inactive state.*

 - int `deimage_bound` () const
- Returns whether the inactive image is managed by the widget.*

 - void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with default arguments.*

 - void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
- Calls the widget callback function with arbitrary arguments.*

- void `do_callback` (`Fl_Widget *widget`, `void *arg=0`, `Fl_Callback_Reason reason=FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (`int`, `int`, `int`, `int`, `Fl_Align`) `const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () `const`
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (`int px`)
Set the spacing between the label and the horizontal edge of the widget.
- `Fl_Image * image` ()
Gets the image that is used as part of the widget label when in the active state.
- `const Fl_Image * image` () `const`
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`Fl_Image &img`)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`Fl_Image *img`)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () `const`
Returns whether the image is managed by the widget.
- int `inside` (`const Fl_Widget *wgt`) `const`
Checks if this widget is a child of wgt.
- int `is_label_copied` () `const`
Returns whether the current label was assigned with `copy_label()`.
- `const char * label` () `const`
Gets the current label text.
- void `label` (`const char *text`)
Sets the current label pointer.
- void `label` (`Fl_Labeltype a`, `const char *b`)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (`int gap`)
Set the gap between the label and the image in pixels.
- `Fl_Color labelcolor` () `const`
Gets the label color.
- void `labelcolor` (`Fl_Color c`)
Sets the label color.
- `Fl_Font labelfont` () `const`
Gets the font to use.
- void `labelfont` (`Fl_Font f`)
Sets the font to use.
- `Fl_Fontsize labelsz` () `const`
Gets the font size in pixels.
- void `labelsize` (`Fl_Fontsize pix`)
Sets the font size in pixels.
- `Fl_Labeltype labeltype` () `const`
Gets the label type.
- void `labeltype` (`Fl_Labeltype a`)

- Sets the label type.*

 - void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FI_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FI_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.
- `FI_Color` `selection_color` () const

Gets the selection color.
- void `selection_color` (`FI_Color` a)

Sets the selection color.
- void `set_active` ()

Marks the widget as active without sending events or changing focus.
- void `set_changed` ()

Marks the value of the widget as changed.
- void `set_output` ()

Sets a widget to output only.
- void `set_visible` ()

Makes the widget visible.
- void `set_visible_focus` ()

Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void `show` ()

Makes a widget visible.
- void `size` (int W, int H)

Changes the size of the widget.
- int `take_focus` ()

Gives the widget the keyboard focus.
- unsigned int `takeevents` () const

Returns if the widget is able to take events.
- int `test_shortcut` ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const

Gets the current tooltip text.
- void `tooltip` (const char *text)

Sets the current tooltip text.

- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) ([uchar](#) i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual ~[Fl_Widget](#) ()
Destroys the widget.

Protected Member Functions

- void [do_callback_for_item](#) ([Fl_Tree_Item](#) *item, [Fl_Tree_Reason](#) reason)
Do the callback for the specified 'item' using 'reason', setting the [callback_item\(\)](#) and [callback_reason\(\)](#).
- void [draw](#) () [FL_OVERRIDE](#)
Standard FLTK [draw\(\)](#) method, handles drawing the tree widget.
- void [item_clicked](#) ([Fl_Tree_Item](#) *val)
Set the item that was last clicked.

Protected Member Functions inherited from [FI_Group](#)

- [FI_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw_child](#) ([FI_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void [draw_outside_label](#) (const [FI_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) ([FI_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int [on_move](#) (int, int)
Allow derived groups to act when a widget is moved within the group.
- virtual void [on_remove](#) (int)
Allow derived groups to act when a child widget is removed from the group.
- int * [sizes](#) ()
Returns the internal array of widget sizes and positions.
- void [update_child](#) ([FI_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FI_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [drawBackdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.
- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int [flags](#) () const
Gets the widget flags mask.
- void [h](#) (int v)
Internal use only.
- void [set_flag](#) (unsigned int c)
Sets a flag in the flags mask.
- void [w](#) (int v)

Internal use only.

- void [x](#) (int v)

Internal use only.

- void [y](#) (int v)

Internal use only.

Protected Attributes

- [Fl_Scrollbar](#) * [_hscroll](#)

Horizontal scrollbar.

- int [_tih](#)

Tree widget inner xywh dimension: inside borders + scrollbars.

- int [_tiw](#)

- int [_tix](#)

- int [_tiy](#)

- int [_toh](#)

Tree widget outer xywh dimension: outside scrollbars, inside widget border.

- int [_tow](#)

- int [_tox](#)

- int [_toy](#)

- int [_tree_h](#)

the calculated height of the entire tree hierarchy. See [calc_tree\(\)](#)

- int [_tree_w](#)

the calculated width of the entire tree hierarchy. See [calc_tree\(\)](#)

- [Fl_Scrollbar](#) * [_vscroll](#)

Vertical scrollbar.

Friends

- class [Fl_Tree_Item](#)

Additional Inherited Members

Static Public Member Functions inherited from [Fl_Group](#)

- static [Fl_Group](#) * [current](#) ()

Returns the currently active group.

- static void [current](#) ([Fl_Group](#) *g)

Sets the current group.

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)

The default callback for all widgets that don't set a callback.

- static unsigned int [label_shortcut](#) (const char *t)

Returns the Unicode value of the '&x' shortcut in a given text.

- static int [test_shortcut](#) (const char *, const bool require_alt=false)

Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.153.1 Detailed Description

Tree widget.



Figure 11.62 [Fl_Tree](#) example program

```
Fl_Tree                                     // Top level widget
|--- Fl_Tree_Item                         // Items in the tree
|--- Fl_Tree_Prefs                       // Preferences for the tree
    |--- Fl_Tree_Connector (enum)        // Connection modes
    |--- Fl_Tree_Select (enum)          // Selection modes
    |--- Fl_Tree_Sort (enum)            // Sort behavior
```

Similar to [Fl_Browser](#), [Fl_Tree](#) is a browser of [Fl_Tree_Item](#)'s arranged in a parented hierarchy, or 'tree'. Subtrees can be expanded or closed. Items can be added, deleted, inserted, sorted and re-ordered.

The tree items may also contain other FLTK widgets, like buttons, input fields, or even "custom" widgets.

The [callback\(\)](#) is invoked depending on the value of [when\(\)](#):

- [FL_WHEN_RELEASE](#) – callback invoked when left mouse button is released on an item
- [FL_WHEN_CHANGED](#) – callback invoked when left mouse changes selection state

The simple way to define a tree:

```
#include <FL/Fl_Tree.H>
[... ]
Fl_Tree tree(X,Y,W,H);
tree.begin();
tree.add("Flintstones/Fred");
tree.add("Flintstones/Wilma");
tree.add("Flintstones/Pebbles");
tree.add("Simpsons/Homer");
tree.add("Simpsons/Marge");
```

```
tree.add("Simpsons/Bart");
tree.add("Simpsons/Lisa");
tree.end();
```

FEATURES

Items can be added with [add\(\)](#), removed with [remove\(\)](#), completely cleared with [clear\(\)](#), inserted with [insert\(\)](#) and [insert_above\(\)](#), selected/deselected with [select\(\)](#) and [deselect\(\)](#), open/closed with [open\(\)](#) and [close\(\)](#), positioned on the screen with [show_item_top\(\)](#), [show_item_middle\(\)](#) and [show_item_bottom\(\)](#), item children can be swapped around with [Fl_Tree_Item::swap_children\(\)](#), items can be moved around with [Fl_Tree_Item::move\(\)](#), an item's children can be walked with [Fl_Tree_Item::first\(\)](#) and [Fl_Tree_Item::next\(\)](#), an item's children can be indexed directly with [Fl_Tree_Item::child\(\)](#) and [Fl_Tree_Item::children\(\)](#), items can be moved from one subtree to another with [Fl_Tree_Item::deparent\(\)](#) and [Fl_Tree_Item::reparent\(\)](#), sorting can be controlled when items are [add\(\)](#)ed via [sortorder\(\)](#). You can walk the entire tree with [first\(\)](#) and [next\(\)](#). You can walk visible items with [first_visible_item\(\)](#) and [next_visible_item\(\)](#). You can walk selected items with [first_selected_item\(\)](#) and [next_selected_item\(\)](#). Items can be found by their pathname using [find_item\(const char*\)](#), and an item's pathname can be found with [item_pathname\(\)](#). The selected items' colors are controlled by [selection_color\(\)](#) (inherited from [Fl_Widget](#)). A hook is provided to allow you to redefine how item's labels are drawn via [Fl_Tree::item_draw_callback\(\)](#). Items can be interactively dragged using [FL_TREE_SELECT_SINGLE_DRAGGABLE](#).

SELECTION OF ITEMS

The tree can have different selection behaviors controlled by [selectmode\(\)](#). The background color used for selected items is the [Fl_Tree::selection_color\(\)](#). The foreground color for selected items is controlled internally with [fl_contrast\(\)](#).

CHILD WIDGETS

FLTK widgets (including custom widgets) can be assigned to tree items via [Fl_Tree_Item::widget\(\)](#).

When an [Fl_Tree_Item::widget\(\)](#) is defined, the default behavior is for the [widget\(\)](#) to be shown in place of the item's label (if it has one). Only the [widget\(\)](#)'s width will be used; the [widget\(\)](#)'s x() and y() position will be managed by the tree, and the h() will track the item's height. This default behavior can be altered (ABI 1.3.1): Setting [Fl_Tree::item_draw_mode\(\)](#)'s [FL_TREE_ITEM_DRAW_LABEL_AND_WIDGET](#) flag causes the label + widget to be displayed together in that order, and adding the [FL_TREE_ITEM_HEIGHT_FROM_WIDGET](#) flag causes widget's height to define the [widget\(\)](#)'s height.

ICONS

The tree's open/close icons can be redefined with [Fl_Tree::openicon\(\)](#), [Fl_Tree::closeicon\(\)](#). User icons can either be changed globally with [Fl_Tree::usericon\(\)](#), or on a per-item basis with [Fl_Tree_Item::usericon\(\)](#).

Various default preferences can be globally manipulated via [Fl_Tree_Prefs](#), including colors, margins, icons, connection lines, etc.

FONTS AND COLORS

When adding new items to the tree, the new items get the defaults for fonts and colors from:

- [Fl_Tree::item_labelfont\(\)](#) – The default item label font (default: FL_HELVETICA)
- [Fl_Tree::item_labelsize\(\)](#) – The default item label size (default: FL_NORMAL_SIZE)
- [Fl_Tree::item_labelfgcolor\(\)](#) – The default item label foreground color (default: FL_FOREGROUND_COLOR)
- [Fl_Tree::item_labelbgcolor\(\)](#) – The default item label background color (default: 0xffffffff, which tree uses as 'transparent')

Each item ([Fl_Tree_Item](#)) inherits a copy of these font/color attributes when created, and each item has its own methods to let the app change these values on a per-item basis using methods of the same name:

- [Fl_Tree_Item::labelfont\(\)](#) – The item's label font (default: FL_HELVETICA)
- [Fl_Tree_Item::labelsizesize\(\)](#) – The item's label size (default: FL_NORMAL_SIZE)
- [Fl_Tree_Item::labelfgcolor\(\)](#) – The item's label foreground color (default: FL_FOREGROUND_COLOR)
- [Fl_Tree_Item::labelbgcolor\(\)](#) – The item's label background color (default: 0xffffffff, which uses the tree's own bg color)

CALLBACKS

The tree's [callback\(\)](#) will be invoked when items change state or are open/closed. [when\(\)](#) controls when mouse/keyboard events invoke the callback. [callback_item\(\)](#) and [callback_reason\(\)](#) can be used to determine the cause of the callback. e.g.

```
void MyTreeCallback(Fl_Widget *w, void *data) {
    Fl_Tree *tree = (Fl_Tree*)w;
    Fl_Tree_Item *item = (Fl_Tree_Item*)tree->callback_item();    // get selected item
    switch ( tree->callback_reason() ) {
        case FL_TREE_REASON_SELECTED: [..]
        case FL_TREE_REASON_DESELECTED: [..]
        case FL_TREE_REASON_RESELECTED: [..]
        case FL_TREE_REASON_OPENED: [..]
        case FL_TREE_REASON_CLOSED: [..]
    }
    :
}
```

SIMPLE EXAMPLES

To find all the selected items:

```
for ( Fl_Tree_Item *i=first_selected_item(); i; i=next_selected_item(i) )
    printf("Item %s is selected\n", i->label());
```

To get an item's full menu pathname, use [Fl_Tree::item_pathname\(\)](#), e.g.

```
[..]
char pathname[256] = "???";
tree->item_pathname(pathname, sizeof(pathname), item);    // eg. "Parent/Child/Item"
[..]
```

To walk all the items of the tree from top to bottom:

```
// Walk all the items in the tree, and print their labels
for ( Fl_Tree_Item *item = tree->first(); item; item = tree->next(item) ) {
    printf("Item: %s\n", item->label());
}
```

To recursively walk all the children of a particular item, define a function that uses recursion:

```
// Find all of the item's children and print an indented report of their labels
void my_print_all_children(Fl_Tree_Item *item, int indent=0) {
    for ( int t=0; t<item->children(); t++) {
        printf("%%s Item: %%s\\n", indent, "", item->child(t)->label());
        my_print_all_children(item->child(t), indent+4);    // recurse
    }
}
```

To change the default label font and color when creating new items:

```
tree = new Fl_Tree(..);
tree->item_labelfont(FL_COURIER);           // Use Courier font for all new items
tree->item_labelbgcolor(FL_RED);             // Use red color for labels of all new items
[..]
// Now create the items in the tree using the above defaults.
tree->add("Aaa");
tree->add("Bbb");
```

To change the font and color of all existing items in the tree:

```
// Change the font and color of all items currently in the tree
for ( FL_Tree_Item *item = tree->first(); item; item = tree->next(item) ) {
    item->labelfont(FL_COURIER);
    item->labelcolor(FL_RED);
}
```

DISPLAY DESCRIPTION

The following image shows the tree's various visual elements and the methods that control them:

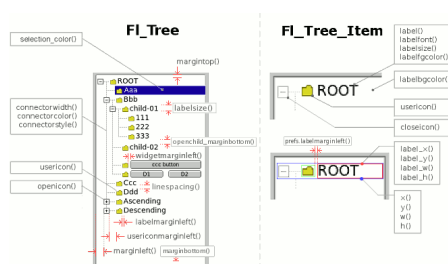


Figure 11.63 Fl_Tree elements

The following shows the protected dimension variables 'tree inner' (tix..) and 'tree outer' (tox..):

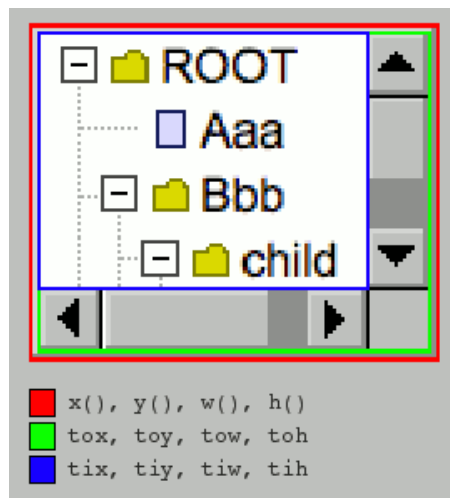


Figure 11.64 FL_Tree inner/outer dimensions

KEYBOARD BINDINGS

The following table lists keyboard bindings for navigating the tree:

Keyboard	FL_TREE_SELECT↵ _MULTI	FL_TREE_SELECT↵ _SINGLE	FL_TREE_SELECT↵ _NONE
Ctrl-A (Linux/Windows)	Select all items	N/A	N/A
Command-A (Mac)	Select all items	N/A	N/A
Space	Selects item	Selects item	N/A
Ctrl-Space	Toggle item	Toggle item	N/A
Shift-Space	Extends selection	Selects item	N/A
Enter	Toggles open/close	Toggles open/close	Toggles open/close
Ctrl-Enter	Toggles open/close	Toggles open/close	Toggles open/close
Shift-Enter	Toggles open/close	Toggles open/close	Toggles open/close
Right / Left	Open/Close item	Open/Close item	Open/Close item
Up / Down	Move focus box up/down	Move focus box up/down	N/A
Shift-Up / Shift-Down	Extend selection up/down	Move focus up/down	N/A
Home / End	Move to top/bottom of tree	Move to top/bottom of tree	Move to top/bottom of tree
PageUp / PageDown	Page up/down	Page up/down	Page up/down

11.153.2 Member Function Documentation

11.153.2.1 add() [1/2]

```
Fl_Tree_Item * Fl_Tree::add (
    const char * path,
    Fl_Tree_Item * item = 0)
```

Adds a new item, given a menu style 'path'.

Any parent nodes that don't already exist are created automatically. Adds the item based on the value of [sortorder\(\)](#). If 'item' is NULL, a new item is created.

To specify items or submenus that contain slashes ('/' or '\') use an escape character to protect them, e.g.

```
:
tree->add("/Holidays/Photos/12\\25\\2010");           // Adds item "12/25/2010"
tree->add("/Pathnames/c:\\\\Program Files\\\\MyApp"); // Adds item "c:\Program Files\MyApp"
:
```

Parameters

in	<i>path</i>	The path to the item, e.g. "Flintstone/Fred".
in	<i>item</i>	The new item to be added. If NULL, a new item is created with a name that is the last element in 'path'.

Returns

The new item added, or 0 on error.

Version

1.3.3

11.153.2.2 add() [2/2]

```
Fl_Tree_Item * Fl_Tree::add (
    Fl_Tree_Item * parent_item,
    const char * name)
```

Add a new child item labeled 'name' to the specified 'parent_item'.

Parameters

in	<i>parent_item</i>	The parent item the new child item will be added to. Must not be NULL.
in	<i>name</i>	The label for the new item

Returns

The new item added.

Version

1.3.0 release

11.153.2.3 calc_dimensions()

```
void Fl_Tree::calc_dimensions ()
```

Recalculate widget dimensions and scrollbar visibility, normally managed automatically.

Low overhead way to update the tree widget's outer/inner dimensions and re-determine scrollbar visibility based on these changes without recalculating the entire size of the tree data.

Assumes that either the tree's size in `_tree_w/_tree_h` are correct so that scrollbar visibility can be calculated easily, or are both zero indicating scrollbar visibility can't be calculated yet.

This method is called when the widget is [resize\(\)](#)ed or if the scrollbar's sizes are changed (affects tree widget's inner dimensions `tix/y/w/h`), and also used by [calc_tree\(\)](#).

Version

1.3.3 ABI feature

11.153.2.4 `calc_tree()`

```
void Fl_Tree::calc_tree ()
```

Recalculates the tree's sizes and scrollbar visibility, normally managed automatically.

On return:

- `_tree_w` will be the overall pixel width of the entire viewable tree
- `_tree_h` will be the overall pixel height ""
- scrollbar visibility and pan sizes are updated
- internal `_tix/_tiy/_tiw/_tih` dimensions are updated

`_tree_w/_tree_h` include the tree's margins (e.g. `marginleft()`), whether items are open or closed, label contents and font sizes, etc.

The tree hierarchy's size is managed separately from the widget's size as an optimization; this way `resize()` on the widget doesn't involve recalculating the tree's hierarchy needlessly, as widget size has no bearing on the tree hierarchy.

The tree hierarchy's size only changes when items are added/removed, open/closed, label contents or font sizes changed, margins changed, etc.

This calculation involves walking the *entire* tree from top to bottom, potentially a slow calculation if the tree has many items (potentially hundreds of thousands), and should therefore be called sparingly.

For this reason, `recalc_tree()` is used as a way to /schedule/ calculation when changes affect the tree hierarchy's size.

Apps may want to call this method directly if the app makes changes to the tree's geometry, then immediately needs to work with the tree's new dimensions before an actual redraw (and recalc) occurs. (This use by an app should only rarely be needed)

11.153.2.5 `callback_item()` [1/2]

```
Fl_Tree_Item * Fl_Tree::callback_item ()
```

Gets the item that caused the callback.

The `callback()` can use this value to see which item changed.

11.153.2.6 `callback_item()` [2/2]

```
void Fl_Tree::callback_item (
    Fl_Tree_Item * item)
```

Sets the item that was changed for this callback.

Used internally to pass the item that invoked the callback.

11.153.2.7 `callback_reason()` [1/2]

```
Fl_Tree_Reason Fl_Tree::callback_reason () const
```

Gets the reason for this callback.

The `callback()` can use this value to see why it was called. Example:

```

:
void MyTreeCallback(Fl_Widget *w, void *userdata) {
    Fl_Tree *tree = (Fl_Tree*)w;
    Fl_Tree_Item *item = tree->callback_item();    // the item changed (can be NULL if more than one
item was changed!)
    switch ( tree->callback_reason() ) {           // reason callback was invoked
        case FL_TREE_REASON_OPENED: ..item was opened..
        case FL_TREE_REASON_CLOSED: ..item was closed..
        case FL_TREE_REASON_SELECTED: ..item was selected..
        case FL_TREE_REASON_RESELECTED: ..item was reselected (double-clicked, etc)..
        case FL_TREE_REASON_DESELECTED: ..item was deselected..
    }
}
:

```

See also

`item_reselect_mode()` – enables `FL_TREE_REASON_RESELECTED` events

11.153.2.8 callback_reason() [2/2]

```
void Fl_Tree::callback_reason (
    Fl_Tree_Reason reason)
```

Sets the reason for this callback.

Used internally to pass the reason the callback was invoked.

11.153.2.9 clear()

```
void Fl_Tree::clear (
    void )
```

Clear the entire tree's children, including the root.

The tree will be left completely empty.

11.153.2.10 clear_children()

```
void Fl_Tree::clear_children (
    Fl_Tree_Item * item)
```

Clear all the children for 'item'.

Item may not be NULL.

11.153.2.11 close() [1/2]

```
int Fl_Tree::close (
    const char * path,
    int docallback = 1)
```

Closes the item specified by 'path'.

Invokes the callback depending on the value of optional parameter 'docallback'.

Handles calling [redraw\(\)](#) if anything changed.

Items or submenus that themselves contain slashes ('/' or '\') should be escaped, e.g. `close("Holidays/12\25\2010")`.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>path</i>	– the tree item's pathname (e.g. "Flintstones/Fred")
in	<i>docallback</i>	– A flag that determines if the callback() is invoked or not: • 0 - callback() is not invoked • 1 - callback() is invoked if item changed (default), callback_reason() will be <code>FL_TREE_REASON_CLOSED</code>

Returns

- 1 – OK: item closed
- 0 – OK: item was already closed, no change
- -1 – ERROR: item was not found

See also

[open\(\)](#), [close\(\)](#), [is_open\(\)](#), [is_close\(\)](#), [callback_item\(\)](#), [callback_reason\(\)](#)

11.153.2.12 close() [2/2]

```
int Fl_Tree::close (
    Fl_Tree_Item * item,
    int docallback = 1)
```

Closes the specified 'item'.

Invokes the callback depending on the value of optional parameter 'docallback'.

Handles calling [redraw\(\)](#) if anything changed.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>item</i>	– the item to be closed. Must not be NULL.
in	<i>docallback</i>	– A flag that determines if the callback() is invoked or not: • 0 - callback() is not invoked • 1 - callback() is invoked if item changed (default), callback_reason() will be FL_TREE_REASON_CLOSED

Returns

- 1 – item was closed
- 0 – item was already closed, no change

See also

[open\(\)](#), [close\(\)](#), [is_open\(\)](#), [is_close\(\)](#), [callback_item\(\)](#), [callback_reason\(\)](#)

11.153.2.13 [closeicon\(\)](#) [1/2]

```
Fl_Image * Fl_Tree::closeicon () const
```

Returns the icon to be used as the 'close' icon.

If none was set, the internal default is returned, a simple '[-]' icon.

11.153.2.14 [closeicon\(\)](#) [2/2]

```
void Fl_Tree::closeicon (
    Fl_Image * val)
```

Sets the icon to be used as the 'close' icon.

This overrides the built in default '[-]' icon.

Parameters

in	<i>val</i>	– The new image, or zero to use the default '[-]' icon.

11.153.2.15 [connectorstyle\(\)](#)

```
void Fl_Tree::connectorstyle (
    Fl_Tree_Connector val)
```

Sets the line drawing style for inter-connecting items.

See [Fl_Tree_Connector](#) for possible values.

11.153.2.16 [deselect\(\)](#) [1/2]

```
int Fl_Tree::deselect (
    const char * path,
    int docallback = 1)
```

Deselect an item specified by 'path'.

Invokes the callback depending on the value of optional parameter 'docallback'.

Handles calling [redraw\(\)](#) if anything changed.

Items or submenus that themselves contain slashes ('/' or '\') should be escaped, e.g. `deselect("← Holidays/12\25\2010")`.

The callback can use `callback_item()` and `callback_reason()` respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>path</i>	– the tree item's pathname (e.g. "Flintstones/Fred")
in	<i>docallback</i>	– A flag that determines if the <code>callback()</code> is invoked or not: • 0 - the <code>callback()</code> is not invoked • 1 - the <code>callback()</code> is invoked if item changed state (default), <code>callback_reason()</code> will be <code>FL_TREE_REASON_DESELECTED</code>

Returns

- 1 - OK: item's state was changed
- 0 - OK: item was already deselected, no change was made
- -1 - ERROR: item was not found

11.153.2.17 `deselect()` [2/2]

```
int Fl_Tree::deselect (
    Fl_Tree_Item * item,
    int docallback = 1)
```

Deselect the specified `item`.

Invokes the callback depending on the value of optional parameter '`docallback`'.

Handles calling `redraw()` if anything changed.

The callback can use `callback_item()` and `callback_reason()` respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>item</i>	– the item to be deselected. Must not be NULL.
in	<i>docallback</i>	– A flag that determines if the <code>callback()</code> is invoked or not: • 0 - the <code>callback()</code> is not invoked • 1 - the <code>callback()</code> is invoked if item changed state (default), <code>callback_reason()</code> will be <code>FL_TREE_REASON_DESELECTED</code>

Returns

- 0 - item was already deselected, no change was made
- 1 - item's state was changed

11.153.2.18 `deselect_all()`

```
int Fl_Tree::deselect_all (
    Fl_Tree_Item * item = 0,
    int docallback = 1)
```

Deselect '`item`' and all its children.

If item is NULL, `first()` is used.

Invokes the callback depending on the value of optional parameter '`docallback`'.

Handles calling `redraw()` if anything changed.

The callback can use `callback_item()` and `callback_reason()` respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>item</i>	The item that will be deselected (along with all its children). If NULL, first() is used.
in	<i>docalldback</i>	– A flag that determines if the callback() is invoked or not: • 0 - the callback() is not invoked • 1 - the callback() is invoked for each item that changed state (default), callback_reason() will be FL_TREE_REASON_DESELECTED

Returns

Count of how many items were actually changed to the deselected state.

11.153.2.19 display()

```
void Fl_Tree::display (
    Fl_Tree_Item * item)
```

Displays 'item', scrolling the tree as necessary.

Parameters

in	<i>item</i>	The item to be displayed. If NULL, first() is used.

11.153.2.20 displayed()

```
int Fl_Tree::displayed (
    Fl_Tree_Item * item)
```

See if 'item' is currently displayed on-screen (visible within the widget).

This can be used to detect if the item is scrolled off-screen. Checks to see if the item's vertical position is within the top and bottom edges of the display window. This does NOT take into account the [hide\(\)](#) / [show\(\)](#) or [open\(\)](#) / [close\(\)](#) status of the item.

Parameters

in	<i>item</i>	The item to be checked. If NULL, first() is used.

Returns

1 if displayed, 0 if scrolled off screen or no items are in tree.

11.153.2.21 draw()

```
void Fl_Tree::draw (
    void ) [protected], [virtual]
```

Standard FLTK [draw\(\)](#) method, handles drawing the tree widget.

Reimplemented from [Fl_Group](#).

11.153.2.22 extend_selection()

```
int Fl_Tree::extend_selection (
    Fl_Tree_Item * from,
    Fl_Tree_Item * to,
    int val = 1,
    bool visible = false)
```

Extend a selection between 'from' and 'to' depending on 'visible'.

Similar to the more efficient `extend_selection_dir(Fl_Tree_Item*, Fl_Tree_Item*, int dir, int val, bool vis)` method, but direction (up or down) doesn't need to be known.

We're less efficient because we search the tree for to/from, then operate on items in between. The more efficient method avoids the "search", but necessitates a direction to be specified to find 'to'.

Used by SHIFT-click to extend a selection between two items inclusive.

Handles calling `redraw()` if anything changed.

Parameters

in	<i>from</i>	Starting item
in	<i>to</i>	Ending item
in	<i>val</i>	Select or deselect items (0=deselect, 1=select, 2=toggle)
in	<i>visible</i>	true=affect only <code>open()</code> , visible items, false=affect open or closed items (default)

Returns

The number of items whose selection states were changed, if any.

Version

1.3.3 ABI feature

11.153.2.23 extend_selection_dir()

```
int Fl_Tree::extend_selection_dir (
    Fl_Tree_Item * from,
    Fl_Tree_Item * to,
    int dir,
    int val,
    bool visible)
```

Extend the selection between and including 'from' and 'to' depending on direction 'dir', 'val', and 'visible'.

Efficient: does not walk entire tree; starts with 'from' and stops at 'to' while moving in direction 'dir'. Dir must be specified though.

If dir cannot be known in advance, such as during SHIFT-click operations, the method `extend_selection(Fl_Tree_Item*, Fl_Tree_Item*, int val, bool visible)` should be used.

Handles calling `redraw()` if anything changed.

Parameters

in	<i>from</i>	Starting item
in	<i>to</i>	Ending item
in	<i>dir</i>	Direction to extend selection (FL_Up or FL_Down)
in	<i>val</i>	0=deselect, 1=select, 2=toggle
in	<i>visible</i>	true=affect only <code>open()</code> , visible items, false=affect open or closed items (default)

Returns

The number of items whose selection states were changed, if any.

Version

1.3.3

11.153.2.24 find_clicked()

```
const Fl_Tree_Item * Fl_Tree::find_clicked (
    int yonly = 0) const
```

Find the item that was last clicked on.

You should use [callback_item\(\)](#) instead, which is fast, and is meant to be used within a callback to determine the item clicked.

This method walks the entire tree looking for the first item that is under the mouse. (The value of the 'yonly' flag affects whether both x and y events are checked, or just y)

Use this method /only/ if you've subclassed [Fl_Tree](#), and are receiving events before [Fl_Tree](#) has been able to process and update [callback_item\(\)](#).

Parameters

in	yonly	– 0: check both event's X and Y values. – 1: only check event's Y value, don't care about X.

Returns

The item clicked, or NULL if no item was under the current event.

Version

1.3.0

1.3.3 ABI feature: added yonly parameter

11.153.2.25 find_item()

```
const Fl_Tree_Item * Fl_Tree::find_item (
    const char * path) const
```

Find the item, given a menu style path, e.g.

"/Parent/Child/item". There is both a const and non-const version of this method. Const version allows pure const methods to use this method to do lookups without causing compiler errors.

To specify items or submenus that contain slashes ('/' or '\') use an escape character to protect them, e.g.

```
:
tree->add("/Holidays/Photos/12\\/25\\/2010"); // Adds item "12/25/2010"
tree->add("/Pathnames/c:\\\\Program Files\\\\MyApp"); // Adds item "c:\Program Files\MyApp"
:
```

Parameters

in	path	– the tree item's pathname to be found (e.g. "Flintstones/Fred")

Returns

The item, or NULL if not found.

See also

[item_pathname\(\)](#)

11.153.2.26 first()

```
Fl_Tree_Item * Fl_Tree::first ()
```

Returns the first item in the tree, or 0 if none.

Use this to walk the tree in the forward direction, e.g.


```

:
for ( Fl_Tree_Item *item = tree->first(); item; item = tree->next(item) )
    printf("Item: %s\n", item->label());
:

```

Returns

First item in tree, or 0 if none (tree empty).

See also

[first\(\)](#), [next\(\)](#), [last\(\)](#), [prev\(\)](#)

11.153.2.27 first_selected_item()

```
Fl_Tree_Item * Fl_Tree::first_selected_item ()
```

Returns the first selected item in the tree.

Use this to walk the tree from top to bottom looking for all the selected items, e.g.

```

:
// Walk tree forward, from top to bottom
for ( Fl_Tree_Item *i=tree->first_selected_item(); i; i=tree->next_selected_item(i) )
    printf("Selected item: %s\n", i->label());
:

```

Returns

The first selected item, or 0 if none.

See also

[first_selected_item\(\)](#), [last_selected_item\(\)](#), [next_selected_item\(\)](#)

11.153.2.28 first_visible()

```
Fl_Tree_Item * Fl_Tree::first_visible ()
```

Returns the first [open\(\)](#), visible item in the tree, or 0 if none.

Deprecated in 1.3.3 ABI – use [first_visible_item\(\)](#) instead.

11.153.2.29 first_visible_item()

```
Fl_Tree_Item * Fl_Tree::first_visible_item ()
```

Returns the first [open\(\)](#), visible item in the tree, or 0 if none.

Returns

First visible item in tree, or 0 if none.

See also

[first_visible_item\(\)](#), [last_visible_item\(\)](#), [next_visible_item\(\)](#)

Version

1.3.3

11.153.2.30 get_selected_items()

```
int Fl_Tree::get_selected_items (
    Fl_Tree_Item_Array & ret_items)
```

Returns the currently selected items as an array of 'ret_items'.

Example:

```
:
// Get selected items as an array
Fl_Tree_Item_Array items;
tree->get_selected_items(items);
// Manipulate the returned array
for ( int t=0; t<items.total(); t++ ) {
    Fl_Tree_Item &item = items[t];
    ..do stuff with each selected item..
}
:
```

Parameters

out	<i>ret_items</i>	The returned array of selected items.

Returns

The number of items in the returned array.

See also

[first_selected_item\(\)](#), [next_selected_item\(\)](#)

Version

1.3.3 ABI feature

11.153.2.31 handle()

```
int Fl_Tree::handle (
    int e) [virtual]
```

Standard FLTK event handler for this widget.

Todo add [Fl_Widget_Tracker](#) (see [Fl_Browser_.cxx::handle\(\)](#))

Reimplemented from [Fl_Group](#).

11.153.2.32 hposition() [1/2]

```
int Fl_Tree::hposition () const
```

Returns the horizontal scroll position as a pixel offset.

The position returned is how many pixels of the tree are scrolled off the left edge of the screen.

See also

[hposition\(int\)](#), [vposition\(\)](#), [vposition\(int\)](#)

Note

Must be using FLTK ABI 1.3.3 or higher for this to be effective.

11.153.2.33 hposition() [2/2]

```
void Fl_Tree::hposition (
    int pos)
```

Sets the horizontal scroll offset to position 'pos'.

The position is how many pixels of the tree are scrolled off the left edge of the screen.

Parameters

in	<i>pos</i>	The vertical position (in pixels) to scroll the tree to.

See also

[hposition\(\)](#), [vposition\(\)](#), [vposition\(int\)](#)

Note

Must be using FLTK ABI 1.3.3 or higher for this to be effective.

11.153.2.34 insert()

```
Fl_Tree_Item * Fl_Tree::insert (
    Fl_Tree_Item * item,
    const char * name,
    int pos)
```

Insert a new item 'name' into 'item's children at position 'pos'.

If pos is out of range the new item is

- prepended if pos < 0 or
- appended if pos > item->[children\(\)](#).

Note: pos == [children\(\)](#) is not considered out of range: the item is appended to the child list.

Example:

```
:
tree->add("Aaa/000");           // "000" is index 0 in Aaa's children
tree->add("Aaa/111");           // "111" is index 1 in Aaa's children
tree->add("Aaa/222");           // "222" is index 2 in Aaa's children
:
// How to use insert() to insert a new item between Aaa/111 + Aaa/222
Fl_Tree_Item *item = tree->find_item("Aaa"); // get parent item Aaa
if (item) tree->insert(item, "New item", 2); // insert as a child of Aaa at index #2
:
```

Parameters

in	<i>item</i>	The existing item to insert new child into. Must not be NULL.
in	<i>name</i>	The label for the new item
in	<i>pos</i>	The position of the new item in the child list

Returns

The new item added.

See also

[insert_above\(\)](#)

11.153.2.35 insert_above()

```
Fl_Tree_Item * Fl_Tree::insert_above (
    Fl_Tree_Item * above,
    const char * name)
```

Inserts a new item 'name' above the specified [Fl_Tree_Item](#) 'above'.

Example:

```

:
tree->add("Aaa/000");      // "000" is index 0 in Aaa's children
tree->add("Aaa/111");      // "111" is index 1 in Aaa's children
tree->add("Aaa/222");      // "222" is index 2 in Aaa's children
..
// How to use insert_above() to insert a new item above Aaa/222
Fl_Tree_Item *item = tree->find_item("Aaa/222"); // get item Aaa/222
if (item) tree->insert_above(item, "New item"); // insert new item above it
:

```

Parameters

in	<i>above</i>	– the item above which to insert the new item. Must not be NULL.
in	<i>name</i>	– the name of the new item

Returns

The new item added, or 0 if 'above' could not be found.

See also

[insert\(\)](#)

11.153.2.36 is_close() [1/2]

```

int Fl_Tree::is_close (
    const char * path) const

```

See if item specified by 'path' is closed.

Items or submenus that themselves contain slashes ('/' or '\') should be escaped, e.g. `is_close("Holidays/12\25\2010")`.

Parameters

in	<i>path</i>	– the tree item's pathname (e.g. "Flintstones/Fred")

Returns

- 1 - OK: item is closed
- 0 - OK: item is open
- -1 - ERROR: item was not found

11.153.2.37 is_close() [2/2]

```

int Fl_Tree::is_close (
    Fl_Tree_Item * item) const

```

See if the specified 'item' is closed.

Parameters

in	<i>item</i>	– the item to be tested. Must not be NULL.

Returns

- 1 : item is closed
- 0 : item is open

11.153.2.38 is_hscroll_visible()

```
int Fl_Tree::is_hscroll_visible () const
```

See if the horizontal scrollbar is currently visible.

Returns

1 if scrollbar visible, 0 if not.

Note

Must be using FLTK ABI 1.3.3 or higher for this to be effective.

11.153.2.39 is_open() [1/2]

```
int Fl_Tree::is_open (
    const char * path) const
```

See if item specified by 'path' is open.

Items or submenus that themselves contain slashes ('/' or '\') should be escaped, e.g. `is_open("← Holidays/12\25\2010")`.

Items that are 'open' are themselves not necessarily visible; one of the item's parents might be closed.

Parameters

in	<i>path</i>	– the tree item's pathname (e.g. "Flintstones/Fred")

Returns

- 1 - OK: item is open
- 0 - OK: item is closed
- -1 - ERROR: item was not found

See also

[Fl_Tree_Item::visible_r\(\)](#)

11.153.2.40 is_open() [2/2]

```
int Fl_Tree::is_open (
    Fl\_Tree\_Item * item) const
```

See if 'item' is open.

Items that are 'open' are themselves not necessarily visible; one of the item's parents might be closed.

Parameters

in	<i>item</i>	– the item to be tested. Must not be NULL.

Returns

- 1 : item is open
- 0 : item is closed

11.153.2.41 is_scrollbar()

```
int Fl_Tree::is_scrollbar (
    Fl_Widget * w)
```

See if widget 'w' is one of the [Fl_Tree](#) widget's scrollbars.

Use this to skip over the scrollbars when walking the [child\(\)](#) array. Example:

```
:
for ( int i=0; i<tree->children(); i++ ) {    // walk children
    Fl_Widget *w = tree->child(i);
    if ( tree->is_scrollbar(w) ) continue;    // skip scrollbars
    ..do work here..
}
:
```

Parameters

in	w	Widget to test

Returns

1 if w is a scrollbar, 0 if not.

Todo should be const

11.153.2.42 is_selected() [1/2]

```
int Fl_Tree::is_selected (
    const char * path)
```

See if item specified by 'path' is selected.

Items or submenus that themselves contain slashes (/ or \) should be escaped, e.g. `is_selected("Holidays/12\25\2010")`.

Parameters

in	path	– the tree item's pathname (e.g. "Flintstones/Fred")

Returns

- 1 : item selected
- 0 : item deselected
- -1 : item was not found

11.153.2.43 is_selected() [2/2]

```
int Fl_Tree::is_selected (
    Fl_Tree_Item * item) const
```

See if the specified 'item' is selected.

Parameters

in	item	– the item to be tested. Must not be NULL.

Returns

- 1 : item selected
- 0 : item deselected

11.153.2.44 is_vscroll_visible()

```
int Fl_Tree::is_vscroll_visible () const
```

See if the vertical scrollbar is currently visible.

Returns

1 if scrollbar visible, 0 if not.

11.153.2.45 item_clicked() [1/2]

```
Fl_Tree_Item * Fl_Tree::item_clicked ()
```

Return the item that was last clicked.

Valid only from within the [callback\(\)](#).

Returns

The item clicked, or 0 if none. 0 may also be used to indicate several items were clicked/changed.

Deprecated in 1.3.3 ABI – use [callback_item\(\)](#) instead.

11.153.2.46 item_clicked() [2/2]

```
void Fl_Tree::item_clicked (
    Fl_Tree_Item * item) [protected]
```

Set the item that was last clicked.

Should only be used by subclasses needing to change this value. Normally [Fl_Tree](#) manages this value.

Deprecated in 1.3.3 ABI – use [callback_item\(\)](#) instead.

11.153.2.47 item_draw_mode() [1/3]

```
Fl_Tree_Item_Draw_Mode Fl_Tree::item_draw_mode () const
```

Get the 'item draw mode' used for the tree.

Version

1.3.1 ABI feature

11.153.2.48 item_draw_mode() [2/3]

```
void Fl_Tree::item_draw_mode (
    Fl_Tree_Item_Draw_Mode mode)
```

Set the 'item draw mode' used for the tree to 'mode'.

This affects how items in the tree are drawn, such as when a [widget\(\)](#) is defined. See [Fl_Tree_Item_Draw_Mode](#) for possible values.

Version

1.3.1 ABI feature

11.153.2.49 item_draw_mode() [3/3]

```
void Fl_Tree::item_draw_mode (
    int mode)
```

Set the 'item draw mode' used for the tree to integer 'mode'.

This affects how items in the tree are drawn, such as when a [widget\(\)](#) is defined. See [Fl_Tree_Item_Draw_Mode](#) for possible values.

Version

1.3.1 ABI feature

11.153.2.50 item_labelbgcolor() [1/2]

```
void Fl_Tree::item_labelbgcolor (
    Fl_Color val)
```

Set the default label background color used for creating new items.

A special case is made for color 0xffffffff (default) which is treated as 'transparent'. To change the background color on a per-item basis, use [Fl_Tree_Item::labelbgcolor\(Fl_Color\)](#)

11.153.2.51 item_labelbgcolor() [2/2]

```
Fl_Color Fl_Tree::item_labelbgcolor (
    void ) const
```

Get the default label background color used for creating new items.

If the color is 0xffffffff, it is 'transparent'.

11.153.2.52 item_labelfgcolor()

```
void Fl_Tree::item_labelfgcolor (
    Fl_Color val)
```

Set the default label foreground color used for creating new items.

To change the foreground color on a per-item basis, use [Fl_Tree_Item::labelfgcolor\(Fl_Color\)](#)

11.153.2.53 item_labelfont()

```
void Fl_Tree::item_labelfont (
    Fl_Font val)
```

Set the default font face used for creating new items.

To change the font face on a per-item basis, use [Fl_Tree_Item::labelfont\(Fl_Font\)](#)

11.153.2.54 item_labelsize()

```
void Fl_Tree::item_labelsize (
    Fl_Fontsize val)
```

Set the default label font size used for creating new items.

To change the font size on a per-item basis, use [Fl_Tree_Item::labelsize\(Fl_Fontsize\)](#)

11.153.2.55 item_pathname()

```
int Fl_Tree::item_pathname (
    char * pathname,
    int pathnamelen,
    const Fl_Tree_Item * item) const
```

Return 'pathname' of size 'pathnamelen' for the specified 'item'.

If 'item' is NULL, [root\(\)](#) is used.

The tree's root will be included in the pathname if [showroot\(\)](#) is on.

Menu items or submenus that contain slashes ('/' or '\') in their names will be escaped with a backslash. This is symmetrical with the [add\(\)](#) function which uses the same escape pattern to set names.

Parameters

out	<i>pathname</i>	The string to use to return the pathname
in	<i>pathnamelen</i>	The maximum length of the string (including NULL). Must not be zero.
in	<i>item</i>	The item whose pathname is to be returned.

Returns

- 0 : OK (`pathname` returns the item's pathname)
- -1 : item not found (`pathname=""`)
- -2 : pathname not large enough (`pathname=""`)

See also

[find_item\(\)](#)

11.153.2.56 item_reselect_mode() [1/2]

```
Fl_Tree_Item_Reselect_Mode Fl_Tree::item_reselect_mode () const
```

Returns the current item re/selection mode.

Version

1.3.1 ABI feature

11.153.2.57 item_reselect_mode() [2/2]

```
void Fl_Tree::item_reselect_mode (
    Fl_Tree_Item_Reselect_Mode mode)
```

Sets the item re/selection mode.

See [Fl_Tree_Item_Reselect_Mode](#) for possible values.

Version

1.3.1 ABI feature

11.153.2.58 last()

```
Fl_Tree_Item * Fl_Tree::last ()
```

Returns the last item in the tree.

This can be used to walk the tree in reverse, e.g.

```
for ( Fl_Tree_Item *item = tree->last(); item; item = tree->prev() )
    printf("Item: %s\n", item->label());
```

Returns

Last item in the tree, or 0 if none (tree empty).

See also

[first\(\)](#), [next\(\)](#), [last\(\)](#), [prev\(\)](#)

11.153.2.59 last_selected_item()

```
Fl_Tree_Item * Fl_Tree::last_selected_item ()
```

Returns the last selected item in the tree.

Use this to walk the tree in reverse from bottom to top looking for all the selected items, e.g.

```
:
// Walk tree in reverse, from bottom to top
for ( Fl_Tree_Item *i=tree->last_selected_item(); i; i=tree->next_selected_item(i, FL_Up) )
    printf("Selected item: %s\n", i->label());
:
```

Returns

The last selected item, or 0 if none.

See also

[first_selected_item\(\)](#), [last_selected_item\(\)](#), [next_selected_item\(\)](#)

Version

1.3.3

11.153.2.60 last_visible()

`Fl_Tree_Item * Fl_Tree::last_visible ()`

Returns the last [open\(\)](#), visible item in the tree.

Deprecated in 1.3.3 – use [last_visible_item\(\)](#) instead.

11.153.2.61 last_visible_item()

`Fl_Tree_Item * Fl_Tree::last_visible_item ()`

Returns the last [open\(\)](#), visible item in the tree.

Returns

Last visible item in the tree, or 0 if none.

See also

[first_visible_item\(\)](#), [last_visible_item\(\)](#), [next_visible_item\(\)](#)

Version

1.3.3

11.153.2.62 load()

```
void Fl_Tree::load (
    class Fl_Preferences & prefs)
```

Load FLTK preferences.

Read a preferences database into the tree widget.

A preferences database is a hierarchical collection of data which can be directly loaded into the tree view for inspection.

Parameters

<code>in</code>	<code>prefs</code>	the Fl_Preferences database

11.153.2.63 next()

```
Fl_Tree_Item * Fl_Tree::next (
    Fl_Tree_Item * item = 0)
```

Return the next item after 'item', or 0 if no more items.

Use this code to walk the entire tree:

```
:
for ( Fl_Tree_Item *i = tree->first(); i; i = tree->next(i) )
    printf("Item: %s\n", i->label());
:
```

Parameters

in	<i>item</i>	The item to use to find the next item. If NULL, returns 0.

Returns

Next item in tree, or 0 if at last item.

See also

[first\(\)](#), [next\(\)](#), [last\(\)](#), [prev\(\)](#)

11.153.2.64 next_item()

```
Fl_Tree_Item * Fl_Tree::next_item (
    Fl_Tree_Item * item,
    int dir = FL_Down,
    bool visible = false)
```

Returns next item after 'item' in direction 'dir' depending on 'visible'.

Next item will be above (if dir==FL_Up) or below (if dir==FL_Down). If 'visible' is true, only items whose parents are [open\(\)](#) will be returned. If 'visible' is false, even items whose parents are [close\(\)](#)ed will be returned.

If item is 0, the return value will be the result of this truth table:

	visible=true	visible=false
	-----	-----
dir=FL_Up:	last_visible_item()	last()
dir=FL_Down:	first_visible_item()	first()

Example use:

```
:
// Walk down the tree showing open(), visible items
for ( Fl_Tree_Item *i=tree->first_visible_item(); i; i=tree->next_item(i, FL_Down, true) )
    printf("Item: %s\n", i->label());

// Walk up the tree showing open(), visible items
for ( Fl_Tree_Item *i=tree->last_visible_item(); i; i=tree->next_item(i, FL_Up, true) )
    printf("Item: %s\n", i->label());

// Walk down the tree showing all items (open or closed)
for ( Fl_Tree_Item *i=tree->first(); i; i=tree->next_item(i, FL_Down, false) )
    printf("Item: %s\n", i->label());

// Walk up the tree showing all items (open or closed)
for ( Fl_Tree_Item *i=tree->last(); i; i=tree->next_item(i, FL_Up, false) )
    printf("Item: %s\n", i->label());
:
```

Parameters

in	<i>item</i>	The item to use to find the next item. If NULL, returns 0.
in	<i>dir</i>	Can be FL_Up or FL_Down (default=FL_Down or 'next')
in	<i>visible</i>	true=return only open() , visible items, false=return open or closed items (default)

Returns

Next item in tree in the direction and visibility specified, or 0 if no more items of specified visibility in that direction.

See also

[first\(\)](#), [last\(\)](#), [next\(\)](#),
[first_visible_item\(\)](#), [last_visible_item\(\)](#), [next_visible_item\(\)](#),
[first_selected_item\(\)](#), [last_selected_item\(\)](#), [next_selected_item\(\)](#)

Version

1.3.3

11.153.2.65 next_selected_item()

```
Fl_Tree_Item * Fl_Tree::next_selected_item (
    Fl_Tree_Item * item = 0,
    int dir = FL_Down)
```

Returns the next selected item above or below 'item', depending on 'dir'.

If 'item' is 0, search starts at either [first\(\)](#) or [last\(\)](#), depending on 'dir': [first\(\)](#) if 'dir' is FL_Down (default), [last\(\)](#) if 'dir' is FL_Up.

Use this to walk the tree looking for all the selected items, e.g.

```
:
// Walk down the tree (forwards)
for ( Fl_Tree_Item *i=tree->first_selected_item(); i; i=tree->next_selected_item(i, FL_Down) )
    printf("Item: %s\n", i->label());

// Walk up the tree (backwards)
for ( Fl_Tree_Item *i=tree->last_selected_item(); i; i=tree->next_selected_item(i, FL_Up) )
    printf("Item: %s\n", i->label());
:
```

Parameters

in	<i>item</i>	The item above or below which we'll find the next selected item. If NULL, first() is used if FL_Down, last() if FL_Up. (default=NULL)
in	<i>dir</i>	The direction to go. FL_Up for moving up the tree, FL_Down for down the tree (default)

Returns

The next selected item, or 0 if there are no more selected items.

See also

[first_selected_item\(\)](#), [last_selected_item\(\)](#), [next_selected_item\(\)](#)

Version

1.3.3

11.153.2.66 next_visible_item()

```
Fl_Tree_Item * Fl_Tree::next_visible_item (
    Fl_Tree_Item * item,
    int dir)
```

Returns next [open\(\)](#), visible item above (dir==FL_Up) or below (dir==FL_Down) the specified 'item', or 0 if no more items.

If 'item' is 0, returns [last\(\)](#) if 'dir' is FL_Up, or [first\(\)](#) if dir is FL_Down.

```
:
// Walk down the tree (forwards)
for ( Fl_Tree_Item *i=tree->first_visible_item(); i; i=tree->next_visible_item(i, FL_Down) )
    printf("Item: %s\n", i->label());

// Walk up the tree (backwards)
for ( Fl_Tree_Item *i=tree->last_visible_item(); i; i=tree->next_visible_item(i, FL_Up) )
    printf("Item: %s\n", i->label());
:
```

Parameters

in	<i>item</i>	The item above/below which we'll find the next visible item
in	<i>dir</i>	The direction to search. Can be FL_Up or FL_Down.

Returns

The item found, or 0 if there's no visible items above/below the specified *item*.

Version

1.3.3

11.153.2.67 open() [1/2]

```
int Fl_Tree::open (
    const char * path,
    int docallback = 1)
```

Opens the item specified by 'path'.

This causes the item's children (if any) to be shown.

Invokes the callback depending on the value of optional parameter 'docallback'.

Handles calling [redraw\(\)](#) if anything changed.

Items or submenus that themselves contain slashes ('/' or '\') should be escaped, e.g. `open("Holidays/12\25\2010")`.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>path</i>	– the tree item's pathname (e.g. "Flintstones/Fred")
in	<i>docallback</i>	– A flag that determines if the callback() is invoked or not: • 0 - callback() is not invoked • 1 - callback() is invoked if item changed (default), callback_reason() will be FL_TREE_REASON_OPENED

Returns

- 1 – OK: item opened
- 0 – OK: item was already open, no change
- -1 – ERROR: item was not found

See also

[open\(\)](#), [close\(\)](#), [is_open\(\)](#), [is_close\(\)](#), [callback_item\(\)](#), [callback_reason\(\)](#)

11.153.2.68 open() [2/2]

```
int Fl_Tree::open (
    Fl_Tree_Item * item,
    int docallback = 1)
```

Open the specified 'item'.

This causes the item's children (if any) to be shown.

Invokes the callback depending on the value of optional parameter 'docallback'.

Handles calling [redraw\(\)](#) if anything changed.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>item</i>	– the item to be opened. Must not be NULL.
in	<i>docalldback</i>	– A flag that determines if the callback() is invoked or not: • 0 - callback() is not invoked • 1 - callback() is invoked if item changed (default), callback_reason() will be FL_TREE_REASON_OPENED

Returns

- 1 – item was opened
- 0 – item was already open, no change

See also

[open\(\)](#), [close\(\)](#), [is_open\(\)](#), [is_close\(\)](#), [callback_item\(\)](#), [callback_reason\(\)](#)

11.153.2.69 open_toggle()

```
void Fl_Tree::open_toggle (
    Fl_Tree_Item * item,
    int docalldback = 1)
```

Toggle the open state of '*item*'.

Invokes the callback depending on the value of optional parameter '*docalldback*'.

Handles calling [redraw\(\)](#) if anything changed.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>item</i>	– the item whose open state is to be toggled. Must not be NULL.
in	<i>docalldback</i>	– A flag that determines if the callback() is invoked or not: • 0 - callback() is not invoked • 1 - callback() is invoked (default), callback_reason() will be either FL_TREE_REASON_OPENED or FL_TREE_REASON_CLOSED

See also

[open\(\)](#), [close\(\)](#), [is_open\(\)](#), [is_close\(\)](#), [callback_item\(\)](#), [callback_reason\(\)](#)

11.153.2.70 openicon() [1/2]

```
Fl_Image * Fl_Tree::openicon () const
```

Returns the icon to be used as the 'open' icon.

If none was set, the internal default is returned, a simple '[+]' icon.

11.153.2.71 openicon() [2/2]

```
void Fl_Tree::openicon (
    Fl_Image * val)
```

Sets the icon to be used as the 'open' icon.

This overrides the built in default '[+]' icon.

Parameters

<i>in</i>	<i>val</i>	– The new image, or zero to use the default [+] icon.

11.153.2.72 prev()

```
Fl_Tree_Item * Fl_Tree::prev (
    Fl_Tree_Item * item = 0)
```

Return the previous item before 'item', or 0 if no more items.

This can be used to walk the tree in reverse, e.g.

```
:
for ( Fl_Tree_Item *item = tree->first(); item; item = tree->prev(item) )
    printf("Item: %s\n", item->label());
:
```

Parameters

<i>in</i>	<i>item</i>	The item to use to find the previous item. If NULL, returns 0.

Returns

Previous item in tree, or 0 if at first item.

See also

[first\(\)](#), [next\(\)](#), [last\(\)](#), [prev\(\)](#)

11.153.2.73 recalc_tree()

```
void Fl_Tree::recalc_tree ()
```

Schedule tree to recalc the entire tree size.

Note

Must be using FLTK ABI 1.3.3 or higher for this to be effective.

11.153.2.74 remove()

```
int Fl_Tree::remove (
    Fl_Tree_Item * item)
```

Remove the specified 'item' from the tree.

item may not be NULL. If it has children, all those are removed too. If item being removed has focus, no item will have focus.

Returns

0 if done, -1 if 'item' not found.

11.153.2.75 resize()

```
void Fl_Tree::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Resizes the [Fl_Group](#) widget and all of its children.

The [Fl_Group](#) widget first resizes itself, and then it moves and resizes all its children according to the rules documented for [Fl_Group::resizable\(Fl_Widget*\)](#)

See also

[Fl_Group::resizable\(Fl_Widget*\)](#)
[Fl_Group::resizable\(\)](#)
[Fl_Widget::resize\(int,int,int,int\)](#)

Reimplemented from [Fl_Group](#).

11.153.2.76 root()

```
void Fl_Tree::root (
    Fl_Tree_Item * newitem)
```

Sets the root item to 'newitem'.

If a root item already exists, [clear\(\)](#) is called first to clear it before replacing it with newitem.

Use this to install a custom item (derived from [Fl_Tree_Item](#)) as the root of the tree. This allows the derived class to implement custom drawing by overriding [Fl_Tree_Item::draw_item_content\(\)](#).

Version

1.3.3

11.153.2.77 root_label()

```
void Fl_Tree::root_label (
    const char * new_label)
```

Set the label for the root item to 'new_label'.

Makes an internally managed copy of 'new_label'.

11.153.2.78 scrollbar_size() [1/2]

```
int Fl_Tree::scrollbar_size () const
```

Gets the default size of scrollbars' troughs for this widget in pixels.

If this value is zero (default), this widget will use the global [Fl::scrollbar_size\(\)](#) value as the scrollbar's width.

Returns

Scrollbar size in pixels, or 0 if the global [Fl::scrollbar_size\(\)](#) is being used.

See also

[Fl::scrollbar_size\(int\)](#)

11.153.2.79 scrollbar_size() [2/2]

```
void Fl_Tree::scrollbar_size (
    int size)
```

Sets the pixel size of the scrollbars' troughs to 'size' for this widget, in pixels.

Normally you should not need this method, and should use the global [Fl::scrollbar_size\(int\)](#) instead to manage the size of ALL your widgets' scrollbars. This ensures your application has a consistent UI, and is the default behavior. Normally this is what you want.

Only use this method if you really need to override just THIS instance of the widget's scrollbar size. (This need should be rare.)

Setting `size` to the special value of 0 causes the widget to track the global [Fl::scrollbar_size\(\)](#), which is the default.

Parameters

<code>in</code>	<code>size</code>	Sets the scrollbar size in pixels. If 0 (default), scrollbar size tracks the global Fl::scrollbar_size()

See also

[Fl::scrollbar_size\(\)](#)

11.153.2.80 select() [1/2]

```
int Fl_Tree::select (
    const char * path,
    int dcallback = 1)
```

Select the item specified by 'path'.

Invokes the callback depending on the value of optional parameter 'dcallback'.

Handles calling [redraw\(\)](#) if anything changed.

Items or submenus that themselves contain slashes ('/' or '\') should be escaped, e.g. `select("← Holidays/12\\25\\2010")`.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>path</i>	– the tree item's pathname (e.g. "Flintstones/Fred")
in	<i>dcallback</i>	– A flag that determines if the callback() is invoked or not: • 0 - the callback() is not invoked • 1 - the callback() is invoked if item changed state (default), callback_reason() will be FL_TREE_REASON_SELECTED

Returns

- 1 : OK: item's state was changed
- 0 : OK: item was already selected, no change was made
- -1 : ERROR: item was not found

11.153.2.81 select() [2/2]

```
int Fl_Tree::select (
    Fl_Tree_Item * item,
    int dcallback = 1)
```

Select the specified 'item'.

Use '[deselect\(\)](#)' to deselect it.

Invokes the callback depending on the value of optional parameter *dcallback*.

Handles calling [redraw\(\)](#) if anything changed.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>item</i>	– the item to be selected. Must not be NULL.
in	<i>dcallback</i>	– A flag that determines if the callback() is invoked or not: • 0 - the callback() is not invoked • 1 - the callback() is invoked if item changed state, callback_reason() will be FL_TREE_REASON_SELECTED

Returns

- 1 - item's state was changed
- 0 - item was already selected, no change was made

11.153.2.82 select_all()

```
int Fl_Tree::select_all (
    Fl_Tree_Item * item = 0,
    int docallback = 1)
```

Select 'item' and all its children.

If item is NULL, [first\(\)](#) is used.

Invokes the callback depending on the value of optional parameter 'docallback'.

Handles calling [redraw\(\)](#) if anything changed.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>item</i>	The item that will be selected (along with all its children). If NULL, first() is used.
in	<i>docallback</i>	– A flag that determines if the callback() is invoked or not: • 0 - the callback() is not invoked • 1 - the callback() is invoked for each item that changed state (default), callback_reason() will be FL_TREE_REASON_SELECTED

Returns

Count of how many items were actually changed to the selected state.

11.153.2.83 select_only()

```
int Fl_Tree::select_only (
    Fl_Tree_Item * selitem,
    int docallback = 1)
```

Select only the specified item, deselecting all others that might be selected.

If 'selitem' is 0, [first\(\)](#) is used.

Invokes the callback depending on the value of optional parameter 'docallback'.

Handles calling [redraw\(\)](#) if anything changed.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>selitem</i>	The item to be selected. If NULL, first() is used.
in	<i>docallback</i>	– A flag that determines if the callback() is invoked or not: • 0 - the callback() is not invoked • 1 - the callback() is invoked for each item that changed state (default), callback_reason() will be either FL_TREE_REASON_SELECTED or FL_TREE_REASON_DESELECTED

Returns

The number of items whose selection states were changed, if any.

11.153.2.84 select_toggle()

```
void Fl_Tree::select_toggle (
    Fl_Tree_Item * item,
    int docallback = 1)
```

Toggle the select state of the specified 'item'.

Invokes the callback depending on the value of optional parameter 'docallback'.

Handles calling [redraw\(\)](#) if anything changed.

The callback can use [callback_item\(\)](#) and [callback_reason\(\)](#) respectively to determine the item changed and the reason the callback was called.

Parameters

in	<i>item</i>	– the item to be selected. Must not be NULL.
in	<i>docallback</i>	– A flag that determines if the callback() is invoked or not: • 0 - the callback() is not invoked • 1 - the callback() is invoked (default), callback_reason() will be either FL_TREE_REASON_SELECTED or FL_TREE_REASON_DESELECTED

11.153.2.85 selectbox() [1/2]

```
Fl_Boxtype Fl_Tree::selectbox () const
```

Sets the style of box used to draw selected items.

This is an fltk [Fl_Boxtype](#). The default is influenced by FLTK's current [Fl::scheme\(\)](#)

11.153.2.86 selectbox() [2/2]

```
void Fl_Tree::selectbox (
    Fl_Boxtype val)
```

Gets the style of box used to draw selected items.

This is an fltk [Fl_Boxtype](#). The default is influenced by FLTK's current [Fl::scheme\(\)](#)

11.153.2.87 selectmode() [1/2]

```
Fl_Tree_Select Fl_Tree::selectmode () const
```

Gets the tree's current selection mode.

See [Fl_Tree_Select](#) for possible values.

11.153.2.88 selectmode() [2/2]

```
void Fl_Tree::selectmode (
    Fl_Tree_Select val)
```

Sets the tree's selection mode.

See [Fl_Tree_Select](#) for possible values.

11.153.2.89 set_item_focus()

```
void Fl_Tree::set_item_focus (
    Fl_Tree_Item * item)
```

Set the item that currently should have keyboard focus.

Handles calling [redraw\(\)](#) to update the focus box (if it is visible).

Parameters

in	<i>item</i>	The item that should take focus. If NULL, none will have focus.

11.153.2.90 show_item() [1/2]

```
void Fl_Tree::show_item (  
    Fl_Tree_Item * item)
```

Adjust the vertical scrollbar to show 'item' at the top of the display IF it is currently off-screen (for instance [show_item_top\(\)](#)).

If it is already on-screen, no change is made.

Parameters

in	<i>item</i>	The item to be shown. If NULL, first() is used.

See also

[show_item_top\(\)](#), [show_item_middle\(\)](#), [show_item_bottom\(\)](#)

11.153.2.91 show_item() [2/2]

```
void Fl_Tree::show_item (  
    Fl_Tree_Item * item,  
    int yoff)
```

Adjust the vertical scrollbar so that 'item' is visible 'yoff' pixels from the top of the [Fl_Tree](#) widget's display.

For instance, yoff=0 will position the item at the top.

If yoff is larger than the vertical scrollbar's limit, the value will be clipped. So if yoff=100, but scrollbar's max is 50, then 50 will be used.

Parameters

in	<i>item</i>	The item to be shown. If NULL, first() is used.
in	<i>yoff</i>	The pixel offset from the top for the displayed position.

See also

[show_item_top\(\)](#), [show_item_middle\(\)](#), [show_item_bottom\(\)](#)

11.153.2.92 show_item_bottom()

```
void Fl_Tree::show_item_bottom (  
    Fl_Tree_Item * item)
```

Adjust the vertical scrollbar so that 'item' is at the bottom of the display.

Parameters

in	<i>item</i>	The item to be shown. If NULL, first() is used.

11.153.2.93 show_item_middle()

```
void Fl_Tree::show_item_middle (  
    Fl_Tree_Item * item)
```

Adjust the vertical scrollbar so that 'item' is in the middle of the display.

Parameters

in	<i>item</i>	The item to be shown. If NULL, first() is used.

11.153.2.94 show_item_top()

```
void Fl_Tree::show_item_top (
    Fl_Tree_Item * item)
```

Adjust the vertical scrollbar so that 'item' is at the top of the display.

Parameters

in	item	The item to be shown. If NULL, first() is used.

11.153.2.95 show_self()

```
void Fl_Tree::show_self ()
```

Print the tree as 'ascii art' to stdout.
Used mainly for debugging.

Todo should be const

Version

1.3.0

11.153.2.96 showcollapse() [1/2]

```
int Fl_Tree::showcollapse () const
```

Returns 1 if the collapse icon is enabled, 0 if not.

See also

[showcollapse\(int\)](#)

11.153.2.97 showcollapse() [2/2]

```
void Fl_Tree::showcollapse (
    int val)
```

Set if we should show the collapse icon or not.

If collapse icons are disabled, the user will not be able to interactively collapse items in the tree, unless the application provides some other means via [open\(\)](#) and [close\(\)](#).

Parameters

in	val	1: shows collapse icons (default), 0: hides collapse icons.

11.153.2.98 showroot()

```
void Fl_Tree::showroot (
    int val)
```

Set if the root item should be shown or not.

Parameters

in	val	1 – show the root item (default) 0 – hide the root item.

11.153.2.99 sortorder()

```
Fl_Tree_Sort Fl_Tree::sortorder () const
```

Set the default sort order used when items are added to the tree.

See [Fl_Tree_Sort](#) for possible values.

11.153.2.100 usericon() [1/2]

```
Fl_Image * Fl_Tree::usericon () const
```

Returns the [Fl_Image](#) being used as the default user icon for all newly created items.

Returns zero if no icon has been set, which is the default.

11.153.2.101 usericon() [2/2]

```
void Fl_Tree::usericon (
    Fl_Image * val)
```

Sets the [Fl_Image](#) to be used as the default user icon for all newly created items.

If you want to specify user icons on a per-item basis, use [Fl_Tree_Item::usericon\(\)](#) instead.

Parameters

in	val	– The new image to be used, or zero to disable user icons.

11.153.2.102 vposition() [1/2]

```
int Fl_Tree::vposition () const
```

Returns the vertical scroll position as a pixel offset.

The position returned is how many pixels of the tree are scrolled off the top edge of the screen.

See also

[vposition\(int\)](#), [hposition\(\)](#), [hposition\(int\)](#)

11.153.2.103 vposition() [2/2]

```
void Fl_Tree::vposition (
    int pos)
```

Sets the vertical scroll offset to position 'pos'.

The position is how many pixels of the tree are scrolled off the top edge of the screen.

Parameters

in	pos	The vertical position (in pixels) to scroll the tree to.

See also

[vposition\(\)](#), [hposition\(\)](#), [hposition\(int\)](#)

The documentation for this class was generated from the following files:

- [Fl_Tree.H](#)
- [Fl_Tree.cxx](#)

11.154 Fl_Tree_Item Class Reference

Tree widget item.

```
#include <Fl_Tree_Item.H>
```

Public Member Functions

- void `activate` (int val=1)
Change the item's activation state to the optionally specified 'val'.
- `Fl_Tree_Item` * `add` (const `Fl_Tree_Prefs` &prefs, char **arr)
Descend into the path specified by 'arr', and add a new child there.
- `Fl_Tree_Item` * `add` (const `Fl_Tree_Prefs` &prefs, char **arr, `Fl_Tree_Item` *newitem)
Descend into path specified by 'arr' and add 'newitem' there.
- `Fl_Tree_Item` * `add` (const `Fl_Tree_Prefs` &prefs, const char *new_label)
Add a new child to this item with the name 'new_label' and defaults from 'prefs'.
- `Fl_Tree_Item` * `add` (const `Fl_Tree_Prefs` &prefs, const char *new_label, `Fl_Tree_Item` *newitem)
Add 'item' as immediate child with 'new_label' and defaults from 'prefs'.
- `Fl_Tree_Item` * `child` (int index)
Return the child item for the given 'index'.
- const `Fl_Tree_Item` * `child` (int t) const
Return the const child item for the given 'index'.
- int `children` () const
Return the number of children this item has.
- void `clear_children` ()
Clear all the children for this item.
- void `close` ()
Close this item and all its children.
- void `deactivate` ()
Deactivate the item; the callback() won't be invoked when clicked.
- `Fl_Tree_Item` * `deparent` (int index)
Deparent child at index position 'pos'.
- int `depth` () const
Returns how many levels deep this item is in the hierarchy.
- void `deselect` ()
Disable the item's selection state.
- int `deselect_all` ()
Deselect item and all its children.
- void `draw` (int X, int &Y, int W, `Fl_Tree_Item` *itemfocus, int &tree_item_xmax, int lastchild=1, int render=1)
Draw this item and its children.
- virtual int `draw_item_content` (int render)
Draw the item content.
- int `event_on_collapse_icon` (const `Fl_Tree_Prefs` &prefs) const
Was the event on the 'collapse' button of this item?
- int `event_on_item` (const `Fl_Tree_Prefs` &prefs) const
Was event anywhere on the item?
- int `event_on_label` (const `Fl_Tree_Prefs` &prefs) const
Was event on the label() of this item?
- int `event_on_user_icon` (const `Fl_Tree_Prefs` &prefs) const
Was the event on the 'user icon' of this item, if any?
- int `find_child` (const char *name)
Return the index of the immediate child of this item that has the label 'name'.
- int `find_child` (`Fl_Tree_Item` *item)
Find the index number for the specified 'item' in the current item's list of children.
- `Fl_Tree_Item` * `find_child_item` (char **arr)
Non-const version of `Fl_Tree_Item::find_child_item(char **arr)` const.
- const `Fl_Tree_Item` * `find_child_item` (char **arr) const

- Find child item by descending array 'arr' of names.*

 - [FI_Tree_Item](#) * **find_child_item** (const char *name)

*Non-const version of [FI_Tree_Item::find_child_item\(const char *name\)](#) const.*

 - const [FI_Tree_Item](#) * **find_child_item** (const char *name) const

Return the /immediate/ child of current item that has the label 'name'.

 - [FI_Tree_Item](#) * **find_clicked** (const [FI_Tree_Prefs](#) &prefs, int yonly=0)

Non-const version of [FI_Tree_Item::find_clicked\(const FI_Tree_Prefs&,int\)](#) const.

 - const [FI_Tree_Item](#) * **find_clicked** (const [FI_Tree_Prefs](#) &prefs, int yonly=0) const

Find the item that the last event was over.

 - [FI_Tree_Item](#) * **find_item** (char **arr)

*Non-const version of [FI_Tree_Item::find_item\(char **names\)](#) const.*

 - const [FI_Tree_Item](#) * **find_item** (char **arr) const

Find item by descending array of 'names'.

 - [FI_Tree_Item](#) (const [FI_Tree_Item](#) *o)

Copy constructor.

 - [FI_Tree_Item](#) (const [FI_Tree_Prefs](#) &prefs)

Constructor.

 - [FI_Tree_Item](#) ([FI_Tree](#) *tree)

Constructor.

 - int **h** () const

The item's height.

 - int **has_children** () const

See if this item has children.

 - [FI_Tree_Item](#) * **insert** (const [FI_Tree_Prefs](#) &prefs, const char *new_label, int pos=0)

Insert a new item named 'new_label' into current item's children at a specified position 'pos'.

 - [FI_Tree_Item](#) * **insert_above** (const [FI_Tree_Prefs](#) &prefs, const char *new_label)

Insert a new item named 'new_label' above this item.

 - char **is_activated** () const

See if the item is activated.

 - char **is_active** () const

See if the item is activated. Alias for [is_activated\(\)](#).

 - int **is_close** () const

See if the item is 'closed'.

 - int **is_open** () const

See if the item is 'open'.

 - int **is_root** () const

Is this item the root of the tree?

 - char **is_selected** () const

See if the item is selected.

 - int **is_visible** () const

See if the item is visible.

 - int **is_visible_r** () const

See if item and all its parents are [open\(\)](#) and [visible\(\)](#).

 - const char * **label** () const

Return the label.

 - void **label** (const char *val)

Set the label to 'name'.

 - int **label_h** () const

The item's label height.

 - int **label_w** () const

The item's maximum label width to right edge of [FI_Tree](#)'s inner width within scrollbars.

- int **label_x** () const
The item's label x position relative to the window.
- int **label_y** () const
The item's label y position relative to the window.
- **FI_Color labelbgcolor** () const
Return item's label background text color.
- void **labelbgcolor** (**FI_Color** val)
Set item's label background color.
- **FI_Color labelcolor** () const
*Return item's label text color. Alias for **labelfgcolor**() const).*
- void **labelcolor** (**FI_Color** val)
*Set item's label text color. Alias for **labelfgcolor**(FI_Color).*
- **FI_Color labelfgcolor** () const
Return item's label foreground text color.
- void **labelfgcolor** (**FI_Color** val)
Set item's label foreground text color.
- **FI_Font labelfont** () const
Get item's label font face.
- void **labelfont** (**FI_Font** val)
Set item's label font face.
- **FI_Fonsize labelsize** () const
Get item's label font size.
- void **labelsize** (**FI_Fonsize** val)
Set item's label font size.
- int **move** (**FI_Tree_Item** *item, int op=0, int pos=0)
Move the current item above/below/into the specified 'item', where 'op' determines the type of move:
- int **move** (int to, int from)
Move an item within its parent using index numbers.
- int **move_above** (**FI_Tree_Item** *item)
Move the current item above the specified 'item'.
- int **move_below** (**FI_Tree_Item** *item)
Move the current item below the specified 'item'.
- int **move_into** (**FI_Tree_Item** *item, int pos=0)
Parent the current item as a child of the specified 'item'.
- **FI_Tree_Item** * **next** ()
Return the next item in the tree.
- **FI_Tree_Item** * **next_displayed** (**FI_Tree_Prefs** &prefs)
*Same as **next_visible**()*.
- **FI_Tree_Item** * **next_sibling** ()
Return this item's next sibling.
- **FI_Tree_Item** * **next_visible** (**FI_Tree_Prefs** &prefs)
*Return the next **open**(), **visible**() item.*
- void **open** ()
Open this item and all its children.
- void **open_toggle** ()
Toggle the item's open/closed state.
- **FI_Tree_Item** * **parent** ()
Return the parent for this item. Returns NULL if we are the root.
- const **FI_Tree_Item** * **parent** () const
Return the const parent for this item. Returns NULL if we are the root.
- void **parent** (**FI_Tree_Item** *val)

- Set the parent for this item.*

 - `const FI_Tree_Prefs & prefs () const`

Return the parent tree's prefs.
- `FI_Tree_Item * prev ()`

Return the previous item in the tree.
- `FI_Tree_Item * prev_displayed (FI_Tree_Prefs &prefs)`

Same as `prev_visible()`.
- `FI_Tree_Item * prev_sibling ()`

Return this item's previous sibling.
- `FI_Tree_Item * prev_visible (FI_Tree_Prefs &prefs)`

Return the previous `open()`, `visible()` item.
- `int remove_child (const char *new_label)`

Remove immediate child (and its children) by its label 'name'.
- `int remove_child (FI_Tree_Item *item)`

Remove 'item' from the current item's children.
- `int reparent (FI_Tree_Item *newchild, int index)`

Reparent specified item as a child of ourself at position 'pos'.
- `FI_Tree_Item * replace (FI_Tree_Item *new_item)`

Replace the current item with a new item.
- `FI_Tree_Item * replace_child (FI_Tree_Item *olditem, FI_Tree_Item *newitem)`

Replace existing child 'olditem' with 'newitem'.
- `void select (int val=1)`

Change the item's selection state to the optionally specified 'val'.
- `int select_all ()`

Select item and all its children.
- `void select_toggle ()`

Toggle the item's selection state.
- `void show_self (const char *indent="") const`

Print the tree as 'ascii art' to stdout.
- `int swap_children (FI_Tree_Item *a, FI_Tree_Item *b)`

Swap two of our immediate children, given item pointers.
- `void swap_children (int ax, int bx)`

Swap two of our children, given two child index values 'ax' and 'bx'.
- `FI_Tree * tree ()`

Return the tree for this item.
- `const FI_Tree * tree () const`

Return the tree for this item.
- `void update_prev_next (int index)`

Update our `_prev_sibling` and `_next_sibling` pointers to point to neighbors given `index` as being our current position in the parent's item array.
- `void * user_data () const`

Retrieve the user-data value that has been assigned to the item.
- `void user_data (void *data)`

Set a user-data value for the item.
- `FI_Image * userdeicon () const`

Return the deactivated version of the user icon, if any.
- `void userdeicon (FI_Image *val)`

Set the usericon to draw when the item is deactivated.
- `FI_Image * usericon () const`

Get the item's user icon as an `FI_Image`. Returns '0' if disabled.
- `void usericon (FI_Image *val)`

- Set the item's user icon to an [Fl_Image](#).
- `int visible () const`

See if the item is visible. Alias for [is_visible\(\)](#).
- `int visible_r () const`

See if item and all its parents are [open\(\)](#) and [visible\(\)](#).
- `int w () const`

The entire item's width to right edge of [Fl_Tree](#)'s inner width within scrollbars.
- `Fl_Widget * widget () const`

Return FLTK widget assigned to this item.
- `void widget (Fl_Widget *val)`

Assign an FLTK widget to this item.
- `int x () const`

The item's x position relative to the window.
- `int y () const`

The item's y position relative to the window.

Protected Member Functions

- `void _Init (const Fl_Tree_Prefs &prefs, Fl_Tree *tree)`
- `int calc_item_height (const Fl_Tree_Prefs &prefs) const`
- Return the item's 'visible' height.
- `void draw_horizontal_connector (int x1, int x2, int y, const Fl_Tree_Prefs &prefs)`
- Internal: Horizontal connector line based on preference settings.
- `void draw_vertical_connector (int x, int y1, int y2, const Fl_Tree_Prefs &prefs)`
- Internal: Vertical connector line based on preference settings.
- `Fl_Color drawbgcolor () const`
- Returns the recommended background color used for drawing this item.
- `Fl_Color drawfgcolor () const`
- Returns the recommended foreground color used for drawing this item.
- `void hide_widgets ()`
- Internal: Hide the FLTK [widget\(\)](#) for this item and all children.
- `int is_flag (unsigned short val) const`
- See if flag set. Returns 0 or 1.
- `void recalc_tree ()`
- Call this when our geometry is changed.
- `void set_flag (unsigned short flag, int val)`
- Set a flag to an on or off value. val is 0 or 1.
- `void show_widgets ()`
- Internal: Show the FLTK [widget\(\)](#) for this item and all children.

11.154.1 Detailed Description

Tree widget item.

This class is a single tree item, and manages all of the item's attributes. [Fl_Tree_Item](#) is used by [Fl_Tree](#), which is comprised of many instances of [Fl_Tree_Item](#).

[Fl_Tree_Item](#) is hierarchical; it dynamically manages an [Fl_Tree_Item_Array](#) of children that are themselves instances of [Fl_Tree_Item](#). Each item can have zero or more children. When an item has children, [close\(\)](#) and [open\(\)](#) can be used to hide or show them.

Items have their own attributes; font size, face, color. Items maintain their own hierarchy of children.

When you make changes to items, you'll need to tell the tree to [redraw\(\)](#) for the changes to show up.

New 1.3.3 ABI feature: You can define custom items by either adding a custom widget to the item with [Fl_Tree_Item::widget\(\)](#), or override the [draw_item_content\(\)](#) method if you want to just redefine how the label is drawn.

The following shows the `Fl_Tree_Item`'s dimensions, useful when overriding the `draw_item_content()` method:

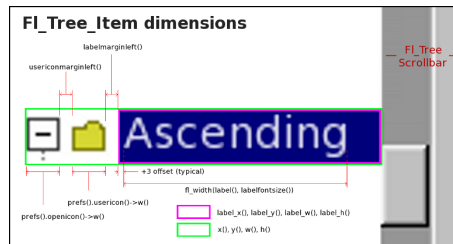


Figure 11.65 `Fl_Tree_Item`'s internal dimensions.

11.154.2 Constructor & Destructor Documentation

11.154.2.1 `Fl_Tree_Item()` [1/2]

```
Fl_Tree_Item::Fl_Tree_Item (
    const Fl_Tree_Prefs & prefs)
```

Constructor.

Makes a new instance of `Fl_Tree_Item` using defaults from 'prefs'.

Deprecated in 1.3.3 ABI – you must use `Fl_Tree_Item(Fl_Tree*)` for proper horizontal scrollbar behavior.

11.154.2.2 `Fl_Tree_Item()` [2/2]

```
Fl_Tree_Item::Fl_Tree_Item (
    Fl_Tree * tree)
```

Constructor.

Makes a new instance of `Fl_Tree_Item` for 'tree'.

This must be used instead of the older, deprecated `Fl_Tree_Item(Fl_Tree_Prefs)` constructor for proper horizontal scrollbar calculation.

Version

1.3.3 ABI feature

11.154.3 Member Function Documentation

11.154.3.1 `activate()`

```
void Fl_Tree_Item::activate (
    int val = 1) [inline]
```

Change the item's activation state to the optionally specified 'val'.

When deactivated, the item will be 'grayed out'; the `callback()` won't be invoked if the user clicks on the label. If a `widget()` is associated with the item, its activation state will be changed as well.

If 'val' is not specified, the item will be activated.

11.154.3.2 `add()` [1/4]

```
Fl_Tree_Item * Fl_Tree_Item::add (
    const Fl_Tree_Prefs & prefs,
    char ** arr)
```

Descend into the path specified by 'arr', and add a new child there.

Should be used only by `Fl_Tree`'s internals. Adds the item based on the value of `prefs.sortorder()`.

Returns

the item added.

Version

1.3.0 release

11.154.3.3 add() [2/4]

```
Fl_Tree_Item * Fl_Tree_Item::add (
    const Fl_Tree_Prefs & prefs,
    char ** arr,
    Fl_Tree_Item * newitem)
```

Descend into path specified by 'arr' and add 'newitem' there.

Should be used only by [Fl_Tree](#)'s internals. If item is NULL, a new item is created. Adds the item based on the value of prefs.sortorder().

Returns

the item added.

Version

1.3.3 ABI feature

11.154.3.4 add() [3/4]

```
Fl_Tree_Item * Fl_Tree_Item::add (
    const Fl_Tree_Prefs & prefs,
    const char * new_label)
```

Add a new child to this item with the name 'new_label' and defaults from 'prefs'.

An internally managed copy is made of the label string. Adds the item based on the value of prefs.sortorder().

Returns

the item added

Version

1.3.0 release

11.154.3.5 add() [4/4]

```
Fl_Tree_Item * Fl_Tree_Item::add (
    const Fl_Tree_Prefs & prefs,
    const char * new_label,
    Fl_Tree_Item * item)
```

Add 'item' as immediate child with 'new_label' and defaults from 'prefs'.

If 'item' is NULL, a new item is created. An internally managed copy is made of the label string. Adds the item based on the value of prefs.sortorder().

Returns

the item added

Version

1.3.3

11.154.3.6 calc_item_height()

```
int Fl_Tree_Item::calc_item_height (
    const Fl_Tree_Prefs & prefs) const [protected]
```

Return the item's 'visible' height.

Takes into account the item's:

- visibility (if `lis_visible()`, returns 0)

- [labelfont\(\)](#) height: if [label\(\)](#) != NULL
- [widget\(\)](#) height: if [widget\(\)](#) != NULL
- [openicon\(\)](#) height (if has children)
- [usericon\(\)](#) height (if not NULL) Does NOT include [Fl_Tree::linespacing\(\)](#);

Returns

maximum pixel height

11.154.3.7 child()

```
const Fl_Tree_Item * Fl_Tree_Item::child (
    int t) const
```

Return the const child item for the given 'index'.

Return const child item for the specified 'index'.

11.154.3.8 deactivate()

```
void Fl_Tree_Item::deactivate () [inline]
```

Deactivate the item; the callback() won't be invoked when clicked.

Same as activate(0)

11.154.3.9 deparent()

```
Fl_Tree_Item * Fl_Tree_Item::deparent (
    int pos)
```

Deparent child at index position 'pos'.

This creates an "orphaned" item that is still allocated, but has no parent or siblings. Normally the caller would want to immediately reparent the orphan elsewhere.

A successfully orphaned item will have its [parent\(\)](#) and [prev_sibling\(\)/next_sibling\(\)](#) set to NULL.

Returns

- pointer to orphaned item on success
- NULL on error (could not deparent the item)

See also

[reparent\(\)](#)

11.154.3.10 depth()

```
int Fl_Tree_Item::depth () const
```

Returns how many levels deep this item is in the hierarchy.

For instance; root has a depth of zero, and its immediate children would have a depth of 1, and so on. Use e.g. for determining the horizontal indent of this item during drawing.

11.154.3.11 deselect_all()

```
int Fl_Tree_Item::deselect_all () [inline]
```

Deselect item and all its children.

Returns count of how many items were in the 'selected' state, ie. how many items were "changed".

11.154.3.12 draw()

```
void Fl_Tree_Item::draw (
    int X,
    int & Y,
    int W,
    Fl_Tree_Item * itemfocus,
    int & tree_item_xmax,
    int lastchild = 1,
    int render = 1)
```

Draw this item and its children.

Parameters

in	<i>X</i>	Horizontal position for item being drawn
in, out	<i>Y</i>	Vertical position for item being drawn, returns new position for next item
in	<i>W</i>	Recommended width for item
in	<i>itemfocus</i>	The tree's current focus item (if any)
in, out	<i>tree_item_xmax</i>	The tree's running xmax (right-most edge so far). Mainly used by parent tree when render==0 to calculate tree's max width.
in	<i>lastchild</i>	Is this item the last child in a subtree?
in	<i>render</i>	Whether or not to render the item: 0: no rendering, just calculate size w/out drawing. 1: render item as well as size calc

Version

1.3.3 ABI feature: modified parameters

11.154.3.13 draw_horizontal_connector()

```
void Fl_Tree_Item::draw_horizontal_connector (
    int x1,
    int x2,
    int y,
    const Fl_Tree_Prefs & prefs) [protected]
```

Internal: Horizontal connector line based on preference settings.

Parameters

in	<i>x1</i>	The left hand X position of the horizontal connector
in	<i>x2</i>	The right hand X position of the horizontal connector
in	<i>y</i>	The vertical position of the horizontal connector
in	<i>prefs</i>	The Fl_Tree prefs

11.154.3.14 draw_item_content()

```
int Fl_Tree_Item::draw_item_content (
    int render) [virtual]
```

Draw the item content.

This method can be overridden to implement custom drawing by filling the `label_[xywh]()` area with content.

A minimal example of how to override `draw_item_content()` and draw just a normal item's background and label ourselves:

```
class MyTreeItem : public Fl_Tree_Item {
public:
    MyTreeItem() { }
    ~MyTreeItem() { }
    // DRAW OUR CUSTOM CONTENT FOR THE ITEM
```

```

int draw_item_content(int render) {
    // Our item's dimensions + text content
    int X=label_x(), Y=label_y(), W=label_w(), H=label_h();
    const char *text = label() ? label() : "";
    // Rendering? Do any drawing that's needed
    if ( render ) {
        // Draw bg -- a filled rectangle
        fl_color(drawbgcolor()); fl_rectf(X,Y,W,H);
        // Draw label
        fl_font(labelfont(), labelsizesize()); // use item's label font/size
        fl_color(drawfgcolor()); // use recommended fg color
        fl_draw(text, X,Y,W,H, FL_ALIGN_LEFT); // draw the item's label
    }
    // Rendered or not, we must calculate content's max X position
    int lw=0, lh=0;
    fl_measure(text, lw, lh); // get width of label text
    return X + lw; // return X + label width
};

```

You can draw anything you want inside `draw_item_content()` using any of the `fl_draw.H` functions, as long as it's within the label's xywh area.

To add instances of your custom item to the tree, you can use:

```

// Example #1: using add()
MyTreeItem *bart = new MyTreeItem(..); // class derived from Fl_Tree_Item
tree->add("/Simpsons/Bart", bart); // Add item as /Simpsons/Bart

```

..or you can insert or replace existing items:

```

// Example #2: using replace()
MyTreeItem *marge = new MyTreeItem(..); // class derived from Fl_Tree_Item
item = tree->add("/Simpsons/Marge"); // create item
item->replace(mi); // replace it with our own

```

Parameters

in	<i>render</i>	Whether we should render content (1), or just tally the geometry (0). <code>Fl_Tree</code> may want only to find the widest item in the tree for scrollbar calculations.

Returns

the right-most X coordinate, or 'xmax' of content we drew, i.e. the "scrollable" content. The tree uses the largest xmax to determine the maximum width of the tree's content (needed for e.g. computing the horizontal scrollbar's size).

Version

1.3.3 ABI feature

11.154.3.15 draw_vertical_connector()

```

void Fl_Tree_Item::draw_vertical_connector (
    int x,
    int y1,
    int y2,
    const Fl_Tree_Prefs & prefs) [protected]

```

Internal: Vertical connector line based on preference settings.

Parameters

in	<i>x</i>	The x position of the vertical connector
in	<i>y1</i>	The top of the vertical connector
in	<i>y2</i>	The bottom of the vertical connector
in	<i>prefs</i>	The <code>Fl_Tree</code> prefs

11.154.3.16 drawbgcolor()

```
Fl_Color Fl_Tree_Item::drawbgcolor () const [protected]
```

Returns the recommended background color used for drawing this item.

See also

[draw_item_content\(\)](#)

Version

1.3.3 ABI

11.154.3.17 drawfgcolor()

```
Fl_Color Fl_Tree_Item::drawfgcolor () const [protected]
```

Returns the recommended foreground color used for drawing this item.

See also

[draw_item_content\(\)](#)

Version

1.3.3 ABI ABI

11.154.3.18 find_child() [1/2]

```
int Fl_Tree_Item::find_child (  
    const char * name)
```

Return the index of the immediate child of this item that has the label 'name'.

Returns

index of found item, or -1 if not found.

Version

1.3.0 release

11.154.3.19 find_child() [2/2]

```
int Fl_Tree_Item::find_child (  
    Fl_Tree_Item * item)
```

Find the index number for the specified 'item' in the current item's list of children.

Returns

the index, or -1 if not found.

11.154.3.20 find_child_item() [1/2]

```
const Fl_Tree_Item * Fl_Tree_Item::find_child_item (  
    char ** arr) const
```

Find child item by descending array 'arr' of names.

Does not include self in search. Only [Fl_Tree](#) should need this method.

Returns

item, or 0 if not found

Version

1.3.0 release

11.154.3.21 find_child_item() [2/2]

```
const Fl_Tree_Item * Fl_Tree_Item::find_child_item (
    const char * name) const
```

Return the /immediate/ child of current item that has the label 'name'.

Returns

const found item, or 0 if not found.

Version

1.3.3

11.154.3.22 find_clicked()

```
const Fl_Tree_Item * Fl_Tree_Item::find_clicked (
    const Fl_Tree_Prefs & prefs,
    int yonly = 0) const
```

Find the item that the last event was over.

If 'yonly' is 1, only check event's y value, don't care about x.

Parameters

in	<i>prefs</i>	The parent tree's Fl_Tree_Prefs
in	<i>yonly</i>	– 0: check both event's X and Y values. – 1: only check event's Y value, don't care about X.

Returns

pointer to clicked item, or NULL if none found

Version

1.3.3 ABI feature

11.154.3.23 find_item()

```
const Fl_Tree_Item * Fl_Tree_Item::find_item (
    char ** names) const
```

Find item by descending array of 'names'.

Includes self in search. Only [Fl_Tree](#) should need this method. Use [Fl_Tree::find_item\(\)](#) instead.

Returns

const item, or 0 if not found

11.154.3.24 hide_widgets()

```
void Fl_Tree_Item::hide_widgets () [protected]
```

Internal: Hide the [FLTK widget\(\)](#) for this item and all children.

Used by [close\(\)](#) to hide widgets.

11.154.3.25 insert()

```
Fl_Tree_Item * Fl_Tree_Item::insert (
    const Fl_Tree_Prefs & prefs,
    const char * new_label,
    int pos = 0)
```

Insert a new item named 'new_label' into current item's children at a specified position 'pos'.

If pos is out of range the new item is

- prepended if `pos < 0` or
- appended if `pos > item->children()`.

Returns

the new item inserted

See also

[Fl_Tree::insert\(\)](#)

11.154.3.26 insert_above()

```
Fl_Tree_Item * Fl_Tree_Item::insert_above (
    const Fl_Tree_Prefs & prefs,
    const char * new_label)
```

Insert a new item named 'new_label' above this item.

Returns

the new item inserted, or 0 if an error occurred.

11.154.3.27 is_visible_r()

```
int Fl_Tree_Item::is_visible_r () const
```

See if item and all its parents are [open\(\)](#) and [visible\(\)](#).

Returns

1 – item and its parents are [open\(\)](#) and [visible\(\)](#) 0 – item or one of its parents are either not [visible\(\)](#) or [close\(\)](#)ed.

11.154.3.28 label()

```
void Fl_Tree_Item::label (
    const char * name)
```

Set the label to 'name'.

Makes and manages an internal copy of 'name'.

11.154.3.29 label_h()

```
int Fl_Tree_Item::label_h () const [inline]
```

The item's label height.

Version

1.3.3

11.154.3.30 label_w()

```
int Fl_Tree_Item::label_w () const [inline]
```

The item's maximum label width to right edge of [Fl_Tree](#)'s inner width within scrollbars.

Version

1.3.3

11.154.3.31 `label_x()`

```
int Fl_Tree_Item::label_x () const [inline]
```

The item's label x position relative to the window.

Version

1.3.3

11.154.3.32 `label_y()`

```
int Fl_Tree_Item::label_y () const [inline]
```

The item's label y position relative to the window.

Version

1.3.3

11.154.3.33 `labelbgcolor()` [1/2]

```
Fl_Color Fl_Tree_Item::labelbgcolor () const [inline]
```

Return item's label background text color.

If the color is 0xffffffff, the default behavior is the parent tree's bg color will be used. (An overloaded [draw_item_content\(\)](#) can override this behavior.)

11.154.3.34 `labelbgcolor()` [2/2]

```
void Fl_Tree_Item::labelbgcolor (
    Fl_Color val) [inline]
```

Set item's label background color.

A special case is made for color 0xffffffff which uses the parent tree's bg color.

11.154.3.35 `move()` [1/2]

```
int Fl_Tree_Item::move (
    Fl_Tree_Item * item,
    int op = 0,
    int pos = 0)
```

Move the current item above/below/into the specified 'item', where 'op' determines the type of move:

- 0: move above 'item' ('pos' ignored)
- 1: move below 'item' ('pos' ignored)
- 2: move into 'item' as a child (at optional position 'pos')

..and 'pos' determines an optional index position after the move.

Returns

0 on success. a negative number on error:

- -1: one of the items has no parent
- -2: item's index could not be determined
- -3: bad 'op'
- -4: index range error
- -5: could not deparent
- -6: could not reparent at 'pos'
- (Other return values reserved for future use.)

See also

[move_above\(\)](#), [move_below\(\)](#), [move_into\(\)](#), [move\(int,int\)](#)

11.154.3.36 move() [2/2]

```
int Fl_Tree_Item::move (
    int to,
    int from)
```

Move an item within its parent using index numbers.

Item is moved 'to' its new position 'from' its old position.

Returns

- 0: Success
- -1: range error (e.g. if 'to' or 'from' out of range).
- (Other return values reserved for future use)

See also

[move_above\(\)](#), [move_below\(\)](#), [move_into\(\)](#), [move\(Fl_Tree_Item*,int,int\)](#)

11.154.3.37 move_above()

```
int Fl_Tree_Item::move_above (
    Fl_Tree_Item * item)
```

Move the current item above the specified 'item'.

This is the equivalent of calling `move(item,0,0)`.

Returns

0 on success.

On error returns a negative value; see [move\(Fl_Tree_Item*,int,int\)](#) for possible error codes.

See also

[move_below\(\)](#), [move_into\(\)](#), [move\(int,int\)](#), [move\(Fl_Tree_Item*,int,int\)](#)

11.154.3.38 move_below()

```
int Fl_Tree_Item::move_below (
    Fl_Tree_Item * item)
```

Move the current item below the specified 'item'.

This is the equivalent of calling `move(item,1,0)`.

Returns

0 on success.

On error returns a negative value; see [move\(Fl_Tree_Item*,int,int\)](#) for possible error codes.

See also

[move_above\(\)](#), [move_into\(\)](#), [move\(int,int\)](#), [move\(Fl_Tree_Item*,int,int\)](#)

11.154.3.39 move_into()

```
int Fl_Tree_Item::move_into (
    Fl_Tree_Item * item,
    int pos = 0)
```

Parent the current item as a child of the specified 'item'.

This is the equivalent of calling `move(item,2,pos)`.

Returns

0 on success.

On error returns a negative value; see [move\(Fl_Tree_Item*,int,int\)](#) for possible error codes.

See also

[move_above\(\)](#), [move_below\(\)](#), [move\(int,int\)](#), [move\(Fl_Tree_Item*,int,int\)](#)

11.154.3.40 next()

```
Fl_Tree_Item * Fl_Tree_Item::next ()
```

Return the next item in the tree.

This method can be used to walk the tree forward. For an example of how to use this method, see [Fl_Tree::first\(\)](#).

Returns

the next item in the tree, or 0 if there's no more items.

11.154.3.41 next_displayed()

```
Fl_Tree_Item * Fl_Tree_Item::next_displayed (
    Fl_Tree_Prefs & prefs)
```

Same as [next_visible\(\)](#).

Deprecated in 1.3.3 for confusing name, use [next_visible\(\)](#) instead

11.154.3.42 next_sibling()

```
Fl_Tree_Item * Fl_Tree_Item::next_sibling ()
```

Return this item's next sibling.

Moves to the next item below us at the same level (sibling). Use this to move down the tree without changing [depth\(\)](#). effectively skipping over this item's children/descendents.

Returns

item's next sibling, or 0 if none.

11.154.3.43 next_visible()

```
Fl_Tree_Item * Fl_Tree_Item::next_visible (
    Fl_Tree_Prefs & prefs)
```

Return the next [open\(\)](#), [visible\(\)](#) item.

(If this item has children and is closed, children are skipped)

This method can be used to walk the tree forward, skipping items that are not currently open/visible to the user.

Returns

the next [open\(\)](#) [visible\(\)](#) item below us, or 0 if there's no more items.

Version

1.3.3

11.154.3.44 parent()

```
void Fl_Tree_Item::parent (
    Fl_Tree_Item * val) [inline]
```

Set the parent for this item.

Should only be used by [Fl_Tree](#)'s internals.

11.154.3.45 prefs()

```
const Fl_Tree_Prefs & Fl_Tree_Item::prefs () const
```

Return the parent tree's prefs.

Returns

a reference to the parent tree's [Fl_Tree_Prefs](#)

Version

1.3.3 ABI feature

11.154.3.46 prev()

```
Fl_Tree_Item * Fl_Tree_Item::prev ()
```

Return the previous item in the tree.

This method can be used to walk the tree backwards. For an example of how to use this method, see [Fl_Tree::last\(\)](#).

Returns

the previous item in the tree, or 0 if there's no item above this one (hit the root).

11.154.3.47 prev_displayed()

```
Fl_Tree_Item * Fl_Tree_Item::prev_displayed (  
    Fl_Tree_Prefs & prefs)
```

Same as [prev_visible\(\)](#).

Deprecated in 1.3.3 for confusing name, use [prev_visible\(\)](#)

11.154.3.48 prev_sibling()

```
Fl_Tree_Item * Fl_Tree_Item::prev_sibling ()
```

Return this item's previous sibling.

Moves to the previous item above us at the same level (sibling). Use this to move up the tree without changing [depth\(\)](#).

Returns

This item's previous sibling, or 0 if none.

11.154.3.49 prev_visible()

```
Fl_Tree_Item * Fl_Tree_Item::prev_visible (  
    Fl_Tree_Prefs & prefs)
```

Return the previous [open\(\)](#), [visible\(\)](#) item.

(If this item above us has children and is closed, its children are skipped)

This method can be used to walk the tree backward, skipping items that are not currently open/visible to the user.

Returns

the previous [open\(\)](#) [visible\(\)](#) item above us, or 0 if there's no more items.

11.154.3.50 recalc_tree()

```
void Fl_Tree_Item::recalc_tree () [protected]
```

Call this when our geometry is changed.

(Font size, label contents, etc) Schedules tree to recalculate itself, as changes to us may affect tree widget's scrollbar visibility and tab sizes.

Version

1.3.3 ABI

11.154.3.51 remove_child() [1/2]

```
int Fl_Tree_Item::remove_child (
    const char * name)
```

Remove immediate child (and its children) by its label 'name'.

If more than one item matches 'name', only the first matching item is removed.

Parameters

in	<i>name</i>	The label name of the immediate child to remove

Returns

0 if removed, -1 if not found.

Version

1.3.3

11.154.3.52 remove_child() [2/2]

```
int Fl_Tree_Item::remove_child (
    Fl_Tree_Item * item)
```

Remove 'item' from the current item's children.

Returns

0 if removed, -1 if item not an immediate child.

11.154.3.53 reparent()

```
int Fl_Tree_Item::reparent (
    Fl_Tree_Item * newchild,
    int pos)
```

Reparent specified item as a child of ourself at position 'pos'.

Typically 'newchild' was recently orphaned with [deparent\(\)](#).

Returns

- 0: on success
- -1: on error (e.g. if 'pos' out of range) with no changes made.

See also

[deparent\(\)](#)

11.154.3.54 replace()

```
Fl_Tree_Item * Fl_Tree_Item::replace (
    Fl_Tree_Item * newitem)
```

Replace the current item with a new item.

The current item is destroyed if successful. No checks are made to see if an item with the same name exists.

This method can be used to, for example, install 'custom' items into the tree derived from [Fl_Tree_Item](#); see [draw_item_content\(\)](#).

Parameters

in	<i>newitem</i>	The new item to replace the current item

Returns

newitem on success, NULL if could not be replaced.

See also

[Fl_Tree_Item::draw_item_content\(\)](#), [Fl_Tree::root\(Fl_Tree_Item*\)](#)

Version

1.3.3 ABI feature

11.154.3.55 replace_child()

```
Fl_Tree_Item * Fl_Tree_Item::replace_child (
    Fl_Tree_Item * olditem,
    Fl_Tree_Item * newitem)
```

Replace existing child 'olditem' with 'newitem'.

The 'olditem' is destroyed if successful. Can be used to put custom items (derived from [Fl_Tree_Item](#)) into the tree. No checks are made to see if an item with the same name exists.

Parameters

in	<i>olditem</i>	The item to be found and replaced
in	<i>newitem</i>	The new item to take the place of 'olditem'

Returns

newitem on success and 'olditem' is destroyed. NULL on error if 'olditem' was not found as an immediate child.

See also

[replace\(\)](#), [Fl_Tree_Item::draw\(\)](#)

Version

1.3.3 ABI feature

11.154.3.56 select()

```
void Fl_Tree_Item::select (
    int val = 1) [inline]
```

Change the item's selection state to the optionally specified 'val'.

If 'val' is not specified, the item will be selected.

11.154.3.57 select_all()

```
int Fl_Tree_Item::select_all () [inline]
```

Select item and all its children.

Returns count of how many items were in the 'deselected' state, ie. how many items were "changed".

11.154.3.58 show_self()

```
void Fl_Tree_Item::show_self (
    const char * indent = "") const
```

Print the tree as 'ascii art' to stdout.

Used mainly for debugging.

11.154.3.59 show_widgets()

```
void Fl_Tree_Item::show_widgets () [protected]
```

Internal: Show the FLTK [widget\(\)](#) for this item and all children.

Used by [open\(\)](#) to re-show widgets that were hidden by a previous [close\(\)](#)

11.154.3.60 swap_children() [1/2]

```
int Fl_Tree_Item::swap_children (
    Fl_Tree_Item * a,
    Fl_Tree_Item * b)
```

Swap two of our immediate children, given item pointers.

Use e.g. for sorting.

This method is SLOW because it involves linear lookups.

For speed, use [swap_children\(int,int\)](#) instead.

Parameters

in	<i>a,b</i>	The item ptrs of the two items to swap. Both must be immediate children of the current item.

Returns

- 0 : OK
- -1 : failed: item 'a' or 'b' is not our child.

11.154.3.61 swap_children() [2/2]

```
void Fl_Tree_Item::swap_children (
    int ax,
    int bx)
```

Swap two of our children, given two child index values 'ax' and 'bx'.

Use e.g. for sorting.

This method is FAST, and does not involve lookups.

No range checking is done on either index value.

Parameters

in	<i>ax,bx</i>	the index of the items to swap

11.154.3.62 tree() [1/2]

```
Fl_Tree * Fl_Tree_Item::tree () [inline]
```

Return the tree for this item.

Version

1.3.4

11.154.3.63 tree() [2/2]

```
const Fl_Tree * Fl_Tree_Item::tree () const [inline]
```

Return the tree for this item.

Version

1.3.3

11.154.3.64 update_prev_next()

```
void Fl_Tree_Item::update_prev_next (
    int index)
```

Update our `_prev_sibling` and `_next_sibling` pointers to point to neighbors given `index` as being our current position in the parent's item array.

Call this whenever items in the array are added/removed/moved/swapped/etc.

Parameters

in	<i>index</i>	Our index# in the parent. Special case if <code>index=-1</code> : become an orphan; null out all parent/sibling associations.

11.154.3.65 userdeicon() [1/2]

```
Fl_Image * Fl_Tree_Item::userdeicon () const [inline]
```

Return the deactivated version of the user icon, if any.

Returns 0 if none.

11.154.3.66 userdeicon() [2/2]

```
void Fl_Tree_Item::userdeicon (
    Fl_Image * val) [inline]
```

Set the usericon to draw when the item is deactivated.

Use '0' to disable. No internal copy is made; caller must manage icon's memory.

To create a typical 'grayed out' version of your usericon image, you can do the following:

```
// Create tree + usericon for items
Fl_Tree *tree = new Fl_Tree(..);
Fl_Image *usr_icon = new Fl_Pixmap(..); // your usericon
Fl_Image *de_icon = usr_icon->copy(); // make a copy, and..
de_icon->inactive(); // make it 'grayed out'
...
for ( .. ) { // item loop..
    item = tree->add("..."); // create new item
    item->usericon(usr_icon); // assign usericon to items
    item->userdeicon(de_icon); // assign userdeicon to items
    ..
}
```

In the above example, the app should 'delete' the two icons when they're no longer needed (e.g. after the tree is destroyed)

Version

1.3.4

11.154.3.67 usericon()

```
void Fl_Tree_Item::usericon (
    Fl_Image * val) [inline]
```

Set the item's user icon to an [Fl_Image](#).

Use '0' to disable. No internal copy is made; caller must manage icon's memory.

Note, if you expect your items to be deactivated(), use [userdeicon\(Fl_Image*\)](#) to set up a 'grayed out' version of your icon to be used for display.

See also

[userdeicon\(Fl_Image*\)](#)

11.154.3.68 visible_r()

```
int Fl_Tree_Item::visible_r () const [inline]
```

See if item and all its parents are [open\(\)](#) and [visible\(\)](#).

Alias for [is_visible_r\(\)](#).

Returns

1 – item and its parents are [open\(\)](#) and [visible\(\)](#) 0 – item (or one of its parents) are not visible or [close\(\)](#)ed.

The documentation for this class was generated from the following files:

- [FI_Tree_Item.H](#)
- [FI_Tree_Item.cxx](#)

11.155 FI_Tree_Item_Array Class Reference

Manages an array of [FI_Tree_Item](#) pointers.

```
#include <FI_Tree_Item_Array.H>
```

Public Member Functions

- void [add](#) ([FI_Tree_Item](#) *val)
Add an item to the end of the array.*
- void [clear](#) ()
Clear the entire array.
- int [deparent](#) (int pos)
Deparent item at 'pos' from our list of children.
- [FI_Tree_Item_Array](#) (const [FI_Tree_Item_Array](#) *o)
Copy constructor. Makes new copy of array, with new instances of each item.
- [FI_Tree_Item_Array](#) (int new_chunksize=10)
Constructor; creates an empty array.
- void [insert](#) (int pos, [FI_Tree_Item](#) *new_item)
Insert an item at index position pos.
- int [manage_item_destroy](#) () const
- void [manage_item_destroy](#) (int val)
Option to control if FI_Tree_Item_Array's destructor will also destroy the FI_Tree_Item's.
- int [move](#) (int to, int from)
Move item at 'from' to new position 'to' in the array.
- [FI_Tree_Item](#) * [operator\[\]](#) (int i)
Return the item and index i.
- const [FI_Tree_Item](#) * [operator\[\]](#) (int i) const
Const version of operator[](int i)
- int [remove](#) ([FI_Tree_Item](#) *item)
Remove the item from the array.
- void [remove](#) (int index)
Remove the item at.
- int [reparent](#) ([FI_Tree_Item](#) *item, [FI_Tree_Item](#) *newparent, int pos)
Reparent specified item as a child of ourself.
- void [replace](#) (int pos, [FI_Tree_Item](#) *new_item)
Replace the item at index with newitem.
- void [swap](#) (int ax, int bx)
Swap the two items at index positions ax and bx.
- int [total](#) () const
Return the total items in the array, or 0 if empty.
- [~FI_Tree_Item_Array](#) ()
Destructor. Calls each item's destructor, destroys internal _items array.

11.155.1 Detailed Description

Manages an array of [Fl_Tree_Item](#) pointers.

Because FLTK 1.x.x. has mandated that templates and STL not be used, we use this class to dynamically manage the arrays.

None of the methods do range checking on index values; the caller must be sure that index values are within the range $0 < \text{index} < \text{total}()$ (unless otherwise noted).

11.155.2 Constructor & Destructor Documentation

11.155.2.1 Fl_Tree_Item_Array()

```
Fl_Tree_Item_Array::Fl_Tree_Item_Array (
    int new_chunksize = 10)
```

Constructor; creates an empty array.

The optional 'chunksize' can be specified to optimize memory allocation for potentially large arrays. Default chunksize is 10.

11.155.3 Member Function Documentation

11.155.3.1 add()

```
void Fl_Tree_Item_Array::add (
    Fl_Tree_Item * val)
```

Add an item* to the end of the array.

Assumes the item was created with 'new', and will remain allocated.. Fl_Tree_Item_Array will handle calling the item's destructor when the array is cleared or the item remove()'ed.

11.155.3.2 clear()

```
void Fl_Tree_Item_Array::clear ()
```

Clear the entire array.

Each item will be deleted (destructors will be called), and the array will be cleared. total() will return 0.

11.155.3.3 deparent()

```
int Fl_Tree_Item_Array::deparent (
    int pos)
```

Deparent item at 'pos' from our list of children.

Similar to a [remove\(\)](#) without the destruction of the item. This creates an orphaned item (still allocated, has no parent) which soon after is typically reparented elsewhere.

\returns 0 on success, -1 on error (e.g. if \p 'pos' out of range)

11.155.3.4 insert()

```
void Fl_Tree_Item_Array::insert (
    int pos,
    Fl_Tree_Item * new_item)
```

Insert an item at index position pos.

Handles enlarging array if needed, total increased by 1.

If \p pos $\geq$ total(), the item is appended to the array.

If \p pos < 0 , the item is prepended (works like pos == 0).

11.155.3.5 manage_item_destroy()

```
void Fl_Tree_Item_Array::manage_item_destroy (
    int val) [inline]
```

Option to control if [Fl_Tree_Item_Array](#)'s destructor will also destroy the [Fl_Tree_Item](#)'s.

If set: items and item array is destroyed. If clear: only the item array is destroyed, not items themselves.

11.155.3.6 move()

```
int Fl_Tree_Item_Array::move (
    int to,
    int from)
```

Move item at 'from' to new position 'to' in the array.

Due to how the moving an item shuffles the array around, a positional 'move' implies things that may not be obvious:

- When 'from' moved lower in tree, appears BELOW item that was at 'to'.
- When 'from' moved higher in tree, appears ABOVE item that was at 'to'.

Returns

0 on success, -1 on range error (e.g. if 'to' or 'from' out of range)

11.155.3.7 remove() [1/2]

```
int Fl_Tree_Item_Array::remove (
    Fl_Tree_Item * item)
```

Remove the item from the array.

\returns 0 if removed, or -1 if the item was not in the array.

11.155.3.8 remove() [2/2]

```
void Fl_Tree_Item_Array::remove (
    int index)
```

Remove the item at.

Parameters

in	index	from the array.
		The item will be delete'd (if non-NULL), so its destructor will be called.

11.155.3.9 reparent()

```
int Fl_Tree_Item_Array::reparent (
    Fl_Tree_Item * item,
    Fl_Tree_Item * newparent,
    int pos)
```

Reparent specified item as a child of ourself.

Typically 'newchild' was recently orphaned with [deparent\(\)](#).

\returns 0 on success, -1 on error (e.g. if \p 'pos' out of range)

11.155.3.10 replace()

```
void Fl_Tree_Item_Array::replace (
    int index,
    Fl_Tree_Item * newitem)
```

Replace the item at `index` with `newitem`.

Old item at index position will be destroyed, and the new item will take it's place, and stitched into the linked list.

The documentation for this class was generated from the following files:

- [Fl_Tree_Item_Array.H](#)
- [Fl_Tree_Item_Array.cxx](#)

11.156 Fl_Tree_Prefs Class Reference

Tree widget's preferences.

```
#include <Fl_Tree_Prefs.H>
```

Public Member Functions

- [Fl_Image](#) * **closedeicon** () const
Return the deactivated version of the close icon, if any.
- [Fl_Image](#) * **closeicon** () const
Gets the default 'close' icon Returns the Fl_Image of the icon, or 0 if none.*
- void **closeicon** ([Fl_Image](#) *val)
Sets the icon to be used as the 'close' icon.
- int **closeicon_h** () const
- int **closeicon_w** () const
- [Fl_Color](#) **connectorcolor** () const
Get the connector color used for tree connection lines.
- void **connectorcolor** ([Fl_Color](#) val)
Set the connector color used for tree connection lines.
- [Fl_Tree_Connector](#) **connectorstyle** () const
Get the connector style.
- void **connectorstyle** ([Fl_Tree_Connector](#) val)
Set the connector style.
- void **connectorstyle** (int val)
Set the connector style [integer].
- int **connectorwidth** () const
Get the tree connection line's width.
- void **connectorwidth** (int val)
Set the tree connection line's width.
- void **do_item_draw_callback** ([Fl_Tree_Item](#) *o) const
- [Fl_Tree_Prefs](#) ()
Fl_Tree_Prefs constructor.
- [Fl_Tree_Item_Draw_Callback](#) * **item_draw_callback** () const
- void **item_draw_callback** ([Fl_Tree_Item_Draw_Callback](#) *cb, void *data=0)
- [Fl_Tree_Item_Draw_Mode](#) **item_draw_mode** () const
Get the 'item draw mode' used for the tree.
- void **item_draw_mode** ([Fl_Tree_Item_Draw_Mode](#) val)
Set the 'item draw mode' used for the tree to val.
- void * **item_draw_user_data** () const
- [Fl_Color](#) **item_labelbgcolor** () const
Get the default label background color.
- void **item_labelbgcolor** ([Fl_Color](#) val)

- Set the default label background color.*

 - **FI_Color item_labelbgcolor** () const

Get the default label foreground color.

 - void **item_labelbgcolor** (FI_Color val)

Set the default label foreground color.

 - **FI_Font item_labelfont** () const

Return the label's font.

 - void **item_labelfont** (FI_Font val)

Set the label's font to val.

 - **FI_Fontsize item_labelsize** () const

Return the label's size in pixels.

 - void **item_labelsize** (FI_Fontsize val)

Set the label's size in pixels to val.

 - **FI_Tree_Item_Reselect_Mode item_reselect_mode** () const

Returns the current item re/selection mode.

 - void **item_reselect_mode** (FI_Tree_Item_Reselect_Mode mode)

Sets the item re/selection mode.

 - **FI_Color labelbgcolor** () const

Obsolete: Get the default label background color. Please use [item_labelbgcolor\(\)](#) instead.

 - void **labelbgcolor** (FI_Color val)

Obsolete: Set the default label background color. Please use [item_labelbgcolor\(FI_Color\)](#) instead.

 - **FI_Color labelfgcolor** () const

Obsolete: Get the default label foreground color. Please use [item_labelfgcolor\(\)](#) instead.

 - void **labelfgcolor** (FI_Color val)

Obsolete: Set the default label foreground color. Please use [item_labelfgcolor\(FI_Color\)](#) instead.

 - **FI_Font labelfont** () const

Obsolete: Return the label's font. Please use [item_labelfont\(\)](#) instead.

 - void **labelfont** (FI_Font val)

Obsolete: Set the label's font to val. Please use [item_labelfont\(FI_Font\)](#) instead.

 - int **labelmarginleft** () const

Get the label's left margin value in pixels.

 - void **labelmarginleft** (int val)

Set the label's left margin value in pixels.

 - **FI_Fontsize labelsize** () const

Obsolete: Return the label's size in pixels. Please use [item_labelsize\(\)](#) instead.

 - void **labelsize** (FI_Fontsize val)

Obsolete: Set the label's size in pixels to val. Please use [item_labelsize\(FI_Fontsize\)](#) instead.

 - int **linespacing** () const

Get the line spacing value in pixels.

 - void **linespacing** (int val)

Set the line spacing value in pixels.

 - int **marginbottom** () const

Get the bottom margin's value in pixels.

 - void **marginbottom** (int val)

Set the bottom margin's value in pixels This is the extra distance the vertical scroller lets you travel.

 - int **marginleft** () const

Get the left margin's value in pixels.

 - void **marginleft** (int val)

Set the left margin's value in pixels.

 - int **marginbottom** () const

Get the top margin's value in pixels.

- void **margin***top* (int val)
Set the top margin's value in pixels.
- int **openchild_marginbottom** () const
Get the margin below an open child in pixels.
- void **openchild_marginbottom** (int val)
Set the margin below an open child in pixels.
- `Fl_Image` * **opendeicon** () const
Return the deactivated version of the open icon, if any.
- `Fl_Image` * **openicon** () const
Get the current default 'open' icon.
- void **openicon** (`Fl_Image` *val)
Sets the default icon to be used as the 'open' icon when items are add()ed to the tree.
- int **openicon_h** () const
- int **openicon_w** () const
- `Fl_Boxtype` **selectbox** () const
Get the default selection box's box drawing style as an `Fl_Boxtype`.
- void **selectbox** (`Fl_Boxtype` val)
Set the default selection box's box drawing style to val.
- `Fl_Tree_Select` **selectmode** () const
Get the selection mode used for the tree.
- void **selectmode** (`Fl_Tree_Select` val)
Set the selection mode used for the tree to val.
- char **showcollapse** () const
Returns 1 if the collapse icon is enabled, 0 if not.
- void **showcollapse** (int val)
Set if we should show the collapse icon or not.
- int **showroot** () const
Returns 1 if the root item is to be shown, or 0 if not.
- void **showroot** (int val)
Set if the root item should be shown or not.
- `Fl_Tree_Sort` **sortorder** () const
Get the default sort order value.
- void **sortorder** (`Fl_Tree_Sort` val)
Set the default sort order value.
- `Fl_Image` * **userdeicon** () const
Return the deactivated version of the user icon, if any.
- `Fl_Image` * **usericon** () const
Gets the default 'user icon' (default is 0)
- void **usericon** (`Fl_Image` *val)
Sets the default 'user icon' Returns the `Fl_Image*` of the icon, or 0 if none (default).
- int **usericonmarginleft** () const
Get the user icon's left margin value in pixels.
- void **usericonmarginleft** (int val)
Set the user icon's left margin value in pixels.
- int **widgetmarginleft** () const
Get the widget()'s left margin value in pixels.
- void **widgetmarginleft** (int val)
Set the widget's left margin value in pixels.
- `~Fl_Tree_Prefs` ()
`Fl_Tree_Prefs` destructor.

11.156.1 Detailed Description

Tree widget's preferences.

[Fl_Tree](#)'s Preferences class.

This class manages the [Fl_Tree](#)'s defaults. You should probably be using the methods in [Fl_Tree](#) instead of trying to accessing tree's preferences settings directly.

11.156.2 Member Function Documentation

11.156.2.1 closedeicon()

```
Fl_Image * Fl_Tree_Prefs::closedeicon () const [inline]
```

Return the deactivated version of the close icon, if any.

Returns 0 if none.

11.156.2.2 closeicon()

```
void Fl_Tree_Prefs::closeicon (
    Fl_Image * val)
```

Sets the icon to be used as the 'close' icon.

This overrides the built in default '[-]' icon.

Parameters

in	val	– The new image, or zero to use the default '[-]' icon.

11.156.2.3 item_draw_mode()

```
void Fl_Tree_Prefs::item_draw_mode (
    Fl_Tree_Item_Draw_Mode val) [inline]
```

Set the 'item draw mode' used for the tree to `val`.

This affects how items in the tree are drawn, such as when a `widget()` is defined. See [Fl_Tree_Item_Draw_Mode](#) for possible values.

11.156.2.4 item_labelbgcolor() [1/2]

```
Fl_Color Fl_Tree_Prefs::item_labelbgcolor () const [inline]
```

Get the default label background color.

This returns the [Fl_Tree::color\(\)](#) unless [item_labelbgcolor\(\)](#) has been set explicitly.

11.156.2.5 item_labelbgcolor() [2/2]

```
void Fl_Tree_Prefs::item_labelbgcolor (
    Fl_Color val) [inline]
```

Set the default label background color.

Once set, overrides the default behavior of using [Fl_Tree::color\(\)](#).

11.156.2.6 marginbottom()

```
int Fl_Tree_Prefs::marginbottom () const [inline]
```

Get the bottom margin's value in pixels.

This is the extra distance the vertical scroller lets you travel.

11.156.2.7 opendeicon()

```
Fl_Image * Fl_Tree_Prefs::opendeicon () const [inline]
```

Return the deactivated version of the open icon, if any.

Returns 0 if none.

11.156.2.8 openicon() [1/2]

```
Fl_Image * Fl_Tree_Prefs::openicon () const [inline]
```

Get the current default 'open' icon.

Returns the Fl_Image* of the icon, or 0 if none.

11.156.2.9 openicon() [2/2]

```
void Fl_Tree_Prefs::openicon (
    Fl_Image * val)
```

Sets the default icon to be used as the 'open' icon when items are add()ed to the tree.

This overrides the built in default '[+]' icon.

Parameters

in	val	– The new image, or zero to use the default '[+]' icon.

11.156.2.10 selectmode()

```
void Fl_Tree_Prefs::selectmode (
    Fl_Tree_Select val) [inline]
```

Set the selection mode used for the tree to val.

This affects how items in the tree are selected when clicked on and dragged over by the mouse. See Fl_Tree_Select for possible values.

11.156.2.11 showcollapse()

```
void Fl_Tree_Prefs::showcollapse (
    int val) [inline]
```

Set if we should show the collapse icon or not.

If collapse icons are disabled, the user will not be able to interactively collapse items in the tree, unless the application provides some other means via open() and close().

Parameters

in	val	1: shows collapse icons (default), 0: hides collapse icons.

11.156.2.12 showroot()

```
void Fl_Tree_Prefs::showroot (
    int val) [inline]
```

Set if the root item should be shown or not.

Parameters

in	val	1 – show the root item (default) 0 – hide the root item.

11.156.2.13 sortorder()

```
void Fl_Tree_Prefs::sortorder (
    Fl_Tree_Sort val) [inline]
```

Set the default sort order value.

Defines the order new items appear when add()ed to the tree. See Fl_Tree_Sort for possible values.

11.156.2.14 userdeicon()

```
Fl_Image * Fl_Tree_Prefs::userdeicon () const [inline]
```

Return the deactivated version of the user icon, if any.

Returns 0 if none.

The documentation for this class was generated from the following files:

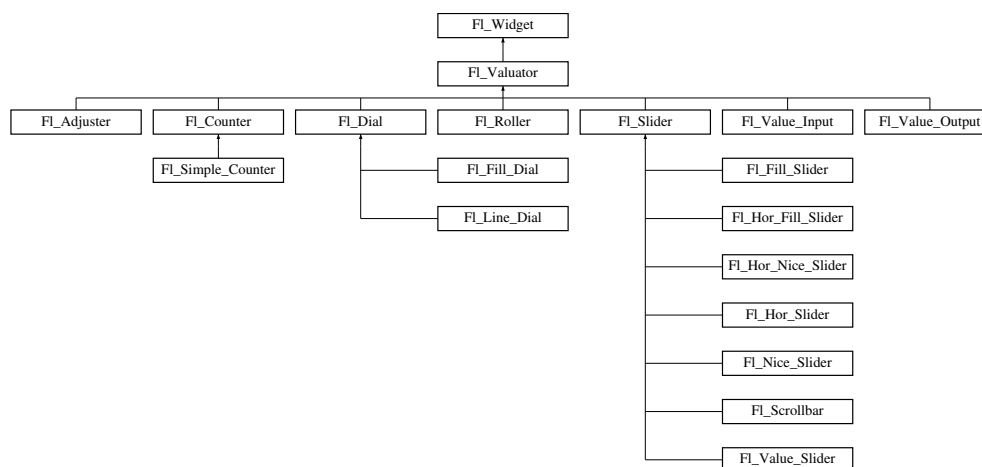
- [Fl_Tree_Prefs.H](#)
- [Fl_Tree_Prefs.cxx](#)

11.157 Fl_Valuator Class Reference

The [Fl_Valuator](#) class controls a single floating-point value and provides a consistent interface to set the value, range, and step, and insures that callbacks are done the same for every object.

```
#include <Fl_Valuator.H>
```

Inheritance diagram for [Fl_Valuator](#):



Public Member Functions

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int **format** (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double **increment** (double, int)
Adds n times the step value to the passed value.
- double **maximum** () const
Gets the maximum value for the valuator.
- void **maximum** (double a)
Sets the maximum value for the valuator.
- double **minimum** () const
Gets the minimum value for the valuator.
- void **minimum** (double a)
Sets the minimum value for the valuator.
- void **precision** (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void **range** (double a, double b)
Sets the minimum and maximum values for the valuator.

- double **round** (double)
Round the passed value to the nearest step increment.
- double **step** () const
Gets or sets the step value.
- void **step** (double a, int b)
See double [FI_Valuator::step\(\)](#) const
- void **step** (double s)
See double [FI_Valuator::step\(\)](#) const.
- void **step** (int a)
See double [FI_Valuator::step\(\)](#) const
- double **value** () const
Gets the floating point(double) value.
- int **value** (double)
Sets the current value.
- **~FI_Valuator** () **FL_OVERRIDE**
Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (**FI_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FI_Gl_Window** * **as_gl_window** ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class **FI_Gl_Window** const * **as_gl_window** () const
- virtual **FI_Group** * **as_group** ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual **FI_Group** const * **as_group** () const
- virtual **FI_Window** * **as_window** ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual **FI_Window** const * **as_window** () const
- void **bind_deimage** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the active state.

- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([Fl_Boxtype](#) new_box)
Sets the box type for the widget.
- [Fl_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([Fl_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()
Disables keyboard focus navigation with this widget.
- [Fl_Color](#) [color](#) () const
Gets the background color of the widget.
- void [color](#) ([Fl_Color](#) bg)
Sets the background color of the widget.
- void [color](#) ([Fl_Color](#) bg, [Fl_Color](#) sel)
Sets the background and selection color of the widget.
- [Fl_Color](#) [color2](#) () const
For back compatibility only.
- void [color2](#) (unsigned a)
For back compatibility only.
- int [contains](#) (const [Fl_Widget](#) *w) const
Checks if w is a child of this widget.
- void [copy_label](#) (const char *new_label)
Sets the current label.
- void [copy_tooltip](#) (const char *text)
Sets the current tooltip text.
- uchar [damage](#) () const
Returns non-zero if [draw\(\)](#) needs to be called.
- void [damage](#) (uchar c)

- Sets the damage bits for the widget.*
- void **damage** (uchar c, int x, int y, int w, int h)
- Sets the damage bits for an area inside the widget.*
- int **damage_resize** (int, int, int, int)
- Internal use only.*
- void **deactivate** ()
- Deactivates the widget.*
- **FL_Image** * **deimage** ()
- Gets the image that is used as part of the widget label when in the inactive state.*
- const **FL_Image** * **deimage** () const
- Gets the image that is used as part of the widget label when in the inactive state.*
- void **deimage** (**FL_Image** &img)
- Sets the image to use as part of the widget label when in the inactive state.*
- void **deimage** (**FL_Image** *img)
- Sets the image to use as part of the widget label when in the inactive state.*
- int **deimage_bound** () const
- Returns whether the inactive image is managed by the widget.*
- void **do_callback** (**FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)
- Calls the widget callback function with default arguments.*
- void **do_callback** (**FL_Widget** *widget, long arg, **FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)
- Calls the widget callback function with arbitrary arguments.*
- void **do_callback** (**FL_Widget** *widget, void *arg=0, **FL_Callback_Reason** reason=**FL_REASON_UNKNOWN**)
- Calls the widget callback function with arbitrary arguments.*
- virtual void **draw** ()=0
- Draws the widget.*
- void **draw_label** (int, int, int, int, **FL_Align**) const
- Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int **h** () const
- Gets the widget height.*
- virtual int **handle** (int event)
- Handles the specified event.*
- virtual void **hide** ()
- Makes a widget invisible.*
- int **horizontal_label_margin** ()
- Get the spacing between the label and the horizontal edge of the widget.*
- void **horizontal_label_margin** (int px)
- Set the spacing between the label and the horizontal edge of the widget.*
- **FL_Image** * **image** ()
- Gets the image that is used as part of the widget label when in the active state.*
- const **FL_Image** * **image** () const
- Gets the image that is used as part of the widget label when in the active state.*
- void **image** (**FL_Image** &img)
- Sets the image to use as part of the widget label when in the active state.*
- void **image** (**FL_Image** *img)
- Sets the image to use as part of the widget label when in the active state.*
- int **image_bound** () const
- Returns whether the image is managed by the widget.*
- int **inside** (const **FL_Widget** *wgt) const
- Checks if this widget is a child of wgt.*
- int **is_label_copied** () const
- Returns whether the current label was assigned with **copy_label()**.*

- `const char * label () const`
Gets the current label text.
- `void label (const char *text)`
Sets the current label pointer.
- `void label (FL_Labeltype a, const char *b)`
Shortcut to set the label text and type in one call.
- `int label_image_spacing ()`
Return the gap size between the label and the image.
- `void label_image_spacing (int gap)`
Set the gap between the label and the image in pixels.
- `FL_Color labelcolor () const`
Gets the label color.
- `void labelcolor (FL_Color c)`
Sets the label color.
- `FL_Font labelfont () const`
Gets the font to use.
- `void labelfont (FL_Font f)`
Sets the font to use.
- `FL_Fonsize labelsize () const`
Gets the font size in pixels.
- `void labelsize (FL_Fonsize pix)`
Sets the font size in pixels.
- `FL_Labeltype labeltype () const`
Gets the label type.
- `void labeltype (FL_Labeltype a)`
Sets the label type.
- `void measure_label (int &ww, int &hh) const`
Sets width ww and height hh accordingly with the label size.
- `bool needs_keyboard () const`
Returns whether this widget needs a keyboard.
- `void needs_keyboard (bool needs)`
Sets whether this widget needs a keyboard.
- `unsigned int output () const`
Returns if a widget is used for output only.
- `FL_Group * parent () const`
Returns a pointer to the parent widget.
- `void parent (FL_Group *p)`
Internal use only - "for hacks only".
- `void position (int X, int Y)`
Repositions the window or widget.
- `void redraw ()`
Schedules the drawing of the widget.
- `void redraw_label ()`
Schedules the drawing of the label.
- `virtual void resize (int x, int y, int w, int h)`
Changes the size or position of the widget.
- `FL_Color selection_color () const`
Gets the selection color.
- `void selection_color (FL_Color a)`
Sets the selection color.
- `void set_active ()`

- Marks the widget as active without sending events or changing focus.*

 - void [set_changed](#) ()
- Marks the value of the widget as changed.*

 - void [set_output](#) ()
- Sets a widget to output only.*

 - void [set_visible](#) ()
- Makes the widget visible.*

 - void [set_visible_focus](#) ()
- Enables keyboard focus navigation with this widget.*

 - int [shortcut_label](#) () const
- Returns whether the widget's label uses '&' to indicate shortcuts.*

 - void [shortcut_label](#) (int value)
- Sets whether the widget's label uses '&' to indicate shortcuts.*

 - virtual void [show](#) ()
- Makes a widget visible.*

 - void [size](#) (int W, int H)
- Changes the size of the widget.*

 - int [take_focus](#) ()
- Gives the widget the keyboard focus.*

 - unsigned int [takeevents](#) () const
- Returns if the widget is able to take events.*

 - int [test_shortcut](#) ()
- Returns true if the widget's label contains the entered '&x' shortcut.*

 - const char * [tooltip](#) () const
- Gets the current tooltip text.*

 - void [tooltip](#) (const char *text)
- Sets the current tooltip text.*

 - [FI_Window](#) * [top_window](#) () const
- Returns a pointer to the top-level window for the widget.*

 - [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
- Finds the x/y offset of the current widget relative to the top-level window.*

 - [uchar](#) [type](#) () const
- Gets the widget type.*

 - void [type](#) ([uchar](#) t)
- Sets the widget type.*

 - int [use_accents_menu](#) ()
- Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*

 - void * [user_data](#) () const
- Gets the user data for this widget.*

 - void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
- Sets the user data for this widget.*

 - void [user_data](#) (void *v)
- Sets the user data for this widget.*

 - int [vertical_label_margin](#) ()
- Get the spacing between the label and the vertical edge of the widget.*

 - void [vertical_label_margin](#) (int px)
- Set the spacing between the label and the vertical edge of the widget.*

 - unsigned int [visible](#) () const
- Returns whether a widget is visible.*

 - unsigned int [visible_focus](#) () const
- Checks whether this widget has a visible focus.*

- void **visible_focus** (int v)
Modifies keyboard focus navigation.
- int **visible_r** () const
Returns whether a widget and all its parents are visible.
- int **w** () const
Gets the widget width.
- **FL_When when** () const
Returns the conditions under which the callback is called.
- void **when** (uchar i)
Sets the flags used to decide when a callback is called.
- **FL_Window * window** () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int **x** () const
Gets the widget position in its window.
- int **y** () const
Gets the widget position in its window.
- virtual **~FL_Widget** ()
Destroys the widget.

Protected Member Functions

- **FL_Valuator** (int X, int Y, int W, int H, const char *L)
*Creates a new **FL_Valuator** widget using the given position, size, and label string.*
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void **value_damage** ()
Asks for partial redraw.

Protected Member Functions inherited from **FL_Widget**

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FL_Boxtype t, FL_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FL_Boxtype t, int x, int y, int w, int h, FL_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.

- void `draw_focus` () const
Draws a focus rectangle around the widget.
- void `draw_focus` (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void `draw_focus` (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void `draw_label` () const
Draws the widget's label at the defined label position.
- void `draw_label` (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int `flags` () const
Gets the widget flags mask.
- void `h` (int v)
Internal use only.
- void `set_flag` (unsigned int c)
Sets a flag in the flags mask.
- void `w` (int v)
Internal use only.
- void `x` (int v)
Internal use only.
- void `y` (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void `default_callback` (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut` (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut` (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
, `OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
, `MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
, `GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
= 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }
flags possible values enumeration.

11.157.1 Detailed Description

The [FL_Valuator](#) class controls a single floating-point value and provides a consistent interface to set the value, range, and step, and insures that callbacks are done the same for every object.

There are probably more of these classes in FLTK than any others:

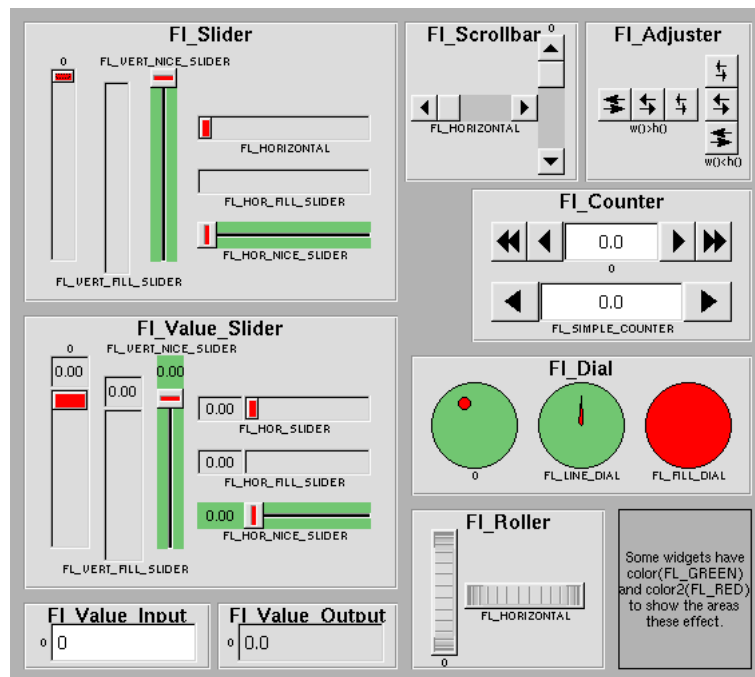


Figure 11.66 Valuator subclasses derived from FL_Valuator

In the above diagram each box surrounds an actual subclass. These are further differentiated by setting the [type\(\)](#) of the widget to the symbolic value labeling the widget. The ones labelled "0" are the default versions with a type(0). For consistency the symbol FL_VERTICAL is defined as zero.

11.157.2 Constructor & Destructor Documentation

11.157.2.1 FL_Valuator()

```
FL_Valuator::FL_Valuator (
    int X,
    int Y,
    int W,
    int H,
    const char * L) [protected]
```

Creates a new [FL_Valuator](#) widget using the given position, size, and label string. The default boxtype is FL_NO_BOX.

11.157.3 Member Function Documentation

11.157.3.1 format()

```
int FL_Valuator::format (
    char * buffer) [virtual]
```

Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter. The actual format used depends on the current step value. If the step value has been set to zero then a %g format is used. If the step value is non-zero, then a %.*f format is used, where the precision is calculated to show sufficient digits for the current step value. An integer step value, such as 1 or 1.0, gives a precision of 0, so the formatted value will appear as an integer.

This method is used by the FL_Valuator_... group of widgets to format the current value into a text string. The return value is the length of the formatted text. The formatted value is written into *buffer*. *buffer* should have space

for at least 128 bytes.

You may override this function to create your own text formatting.

11.157.3.2 increment()

```
double Fl_Valuator::increment (
    double v,
    int n)
```

Adds n times the step value to the passed value.

If step was set to zero it uses `fabs(maximum() - minimum()) / 100`.

11.157.3.3 maximum() [1/2]

```
double Fl_Valuator::maximum () const [inline]
```

Gets the maximum value for the valuator.

11.157.3.4 maximum() [2/2]

```
void Fl_Valuator::maximum (
    double a) [inline]
```

Sets the maximum value for the valuator.

11.157.3.5 minimum() [1/2]

```
double Fl_Valuator::minimum () const [inline]
```

Gets the minimum value for the valuator.

11.157.3.6 minimum() [2/2]

```
void Fl_Valuator::minimum (
    double a) [inline]
```

Sets the minimum value for the valuator.

11.157.3.7 precision()

```
void Fl_Valuator::precision (
    int digits)
```

Sets the step value to $1.0 / 10^{\text{digits}}$.

Precision `digits` is limited to 0...9 to avoid internal overflow errors. Values outside this range are clamped.

Note

For negative values of `digits` the step value is set to $A = 1.0$ and $B = 1$, i.e. $1.0/1 = 1$.

11.157.3.8 range()

```
void Fl_Valuator::range (
    double a,
    double b) [inline]
```

Sets the minimum and maximum values for the valuator.

When the user manipulates the widget, the value is limited to this range. This clamping is done *after* rounding to the step value (this makes a difference if the range is not a multiple of the step).

The minimum may be greater than the maximum. This has the effect of "reversing" the object so the larger values are in the opposite direction. This also switches which end of the filled sliders is filled.

Some widgets consider this a "soft" range. This means they will stop at the range, but if the user releases and grabs the control again and tries to move it further, it is allowed.

The range may affect the display. You must [redraw\(\)](#) the widget after changing the range.

11.157.3.9 round()

```
double Fl_Valuator::round (
    double v)
```

Round the passed value to the nearest step increment.

Does nothing if step is zero.

11.157.3.10 step()

```
double Fl_Valuator::step () const [inline]
```

Gets or sets the step value.

As the user moves the mouse the value is rounded to the nearest multiple of the step value. This is done *before* clamping it to the range. For most widgets the default step is zero.

For precision the step is stored as the ratio of a double *A* and an integer *B* = *A/B*. You can set these values directly.

Currently setting a floating point value sets the nearest *A/1* or *1/B* value possible.

11.157.3.11 value() [1/2]

```
double Fl_Valuator::value (
    void ) const [inline]
```

Gets the floating point(double) value.

See int [value\(double\)](#)

11.157.3.12 value() [2/2]

```
int Fl_Valuator::value (
    double v)
```

Sets the current value.

The new value is *not* clamped or otherwise changed before storing it. Use [clamp\(\)](#) or [round\(\)](#) to modify the value before calling [value\(\)](#). The widget is redrawn if the new value is different than the current one. The initial value is zero.

[changed\(\)](#) will return true if the user has moved the slider, but it will be turned off by [value\(x\)](#) and just before doing a callback (the callback can turn it back on if desired).

11.157.3.13 value_damage()

```
void Fl_Valuator::value_damage () [protected], [virtual]
```

Asks for partial redraw.

Reimplemented in [Fl_Adjuster](#).

The documentation for this class was generated from the following files:

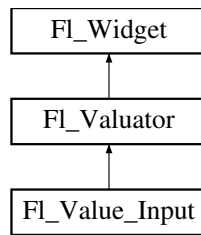
- [Fl_Valuator.H](#)
- [Fl_Valuator.cxx](#)

11.158 Fl_Value_Input Class Reference

The [Fl_Value_Input](#) widget displays a numeric value.

```
#include <Fl_Value_Input.H>
```

Inheritance diagram for [Fl_Value_Input](#):



Public Member Functions

- [FL_Color cursor_color](#) () const
Gets the color of the text cursor.
- void [cursor_color](#) (FL_Color n)
Sets the color of the text cursor.
- [FL_Value_Input](#) (int x, int y, int w, int h, const char *l=0)
Creates a new [FL_Value_Input](#) widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- void [resize](#) (int, int, int, int) [FL_OVERRIDE](#)
Changes the size or position of the widget.
- int [shortcut](#) () const
Returns the current shortcut key for the Input.
- void [shortcut](#) (int s)
Sets the shortcut key to s.
- char [soft](#) () const
If "soft" is turned on, the user is allowed to drag the value outside the range.
- void [soft](#) (char s)
See void [FL_Value_Input::soft\(char s\)](#)
- [FL_Color textcolor](#) () const
Gets the color of the text in the value box.
- void [textcolor](#) (FL_Color n)
Sets the color of the text in the value box.
- [FL_Font textfont](#) () const
Gets the typeface of the text in the value box.
- void [textfont](#) (FL_Font s)
Sets the typeface of the text in the value box.
- [FL_Fontsize textsize](#) () const
Gets the size of the text in the value box.
- void [textsize](#) (FL_Fontsize s)
Sets the size of the text in the value box.

Public Member Functions inherited from [FL_Valuator](#)

- void [bounds](#) (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double [clamp](#) (double)
Clamps the passed value to the valuator range.
- virtual int [format](#) (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double [increment](#) (double, int)
Adds n times the step value to the passed value.
- double [maximum](#) () const

- Gets the maximum value for the valuator.*

 - void **maximum** (double a)

Sets the maximum value for the valuator.

 - double **minimum** () const

Gets the minimum value for the valuator.

 - void **minimum** (double a)

Sets the minimum value for the valuator.

 - void **precision** (int digits)

Sets the step value to $1.0 / 10^{\text{digits}}$.

 - void **range** (double a, double b)

Sets the minimum and maximum values for the valuator.

 - double **round** (double)

Round the passed value to the nearest step increment.

 - double **step** () const

Gets or sets the step value.

 - void **step** (double a, int b)

See double [FI_Valuator::step\(\)](#) const

 - void **step** (double s)

See double [FI_Valuator::step\(\)](#) const.

 - void **step** (int a)

See double [FI_Valuator::step\(\)](#) const

 - double **value** () const

Gets the floating point(double) value.

 - int **value** (double)

Sets the current value.

 - **~FI_Valuator** () **FL_OVERRIDE**

Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void **_clear_fullscreen** ()
 - void **_set_fullscreen** ()
 - void **activate** ()
- Activates the widget.*
- unsigned int **active** () const
- Returns whether the widget is active.*
- int **active_r** () const
- Returns whether the widget and all of its parents are active.*
- [FI_Align](#) **align** () const
- Gets the label alignment.*
- void **align** ([FI_Align](#) alignment)
- Sets the label alignment.*
- long **argument** () const
- Gets the current user data (long) argument that is passed to the callback function.*
- void **argument** (long v)
- Sets the current user data (long) argument that is passed to the callback function.*
- virtual class [FI_Gl_Window](#) * **as_gl_window** ()
- Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).*
- virtual class [FI_Gl_Window](#) const * **as_gl_window** () const
- virtual [FI_Group](#) * **as_group** ()

Returns an *FI_Group* pointer if this widget is an *FI_Group*.

- virtual *FI_Group* const * **as_group** () const
- virtual *FI_Window* * **as_window** ()

Returns an *FI_Window* pointer if this widget is an *FI_Window*.

- virtual *FI_Window* const * **as_window** () const
- void **bind_deimage** (*FI_Image* *img)

Sets the image to use as part of the widget label when in the inactive state.

- void **bind_deimage** (int f)

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.

- void **bind_image** (*FI_Image* *img)

Sets the image to use as part of the widget label when in the active state.

- void **bind_image** (int f)

Bind the image to the widget, so the widget will delete the image when it is no longer needed.

- *FI_Boxtype* **box** () const

Gets the box type of the widget.

- void **box** (*FI_Boxtype* new_box)

Sets the box type for the widget.

- *FI_Callback_p* **callback** () const

Gets the current callback function for the widget.

- void **callback** (*FI_Callback* *cb)

Sets the current callback function for the widget.

- void **callback** (*FI_Callback* *cb, *FI_Callback_User_Data* *p, bool auto_free)

Sets the current callback function and managed user data for the widget.

- void **callback** (*FI_Callback* *cb, void *p)

Sets the current callback function and data for the widget.

- void **callback** (*FI_Callback0* *cb)

Sets the current callback function for the widget.

- void **callback** (*FI_Callback1* *cb, long p=0)

Sets the current callback function for the widget.

- unsigned int **changed** () const

Checks if the widget value changed since the last callback.

- void **clear_active** ()

Marks the widget as inactive without sending events or changing focus.

- void **clear_changed** ()

Marks the value of the widget as unchanged.

- void **clear_damage** (uchar c=0)

Clears or sets the damage flags.

- void **clear_output** ()

Sets a widget to accept input.

- void **clear_visible** ()

Hides the widget.

- void **clear_visible_focus** ()

Disables keyboard focus navigation with this widget.

- *FI_Color* **color** () const

Gets the background color of the widget.

- void **color** (*FI_Color* bg)

Sets the background color of the widget.

- void **color** (*FI_Color* bg, *FI_Color* sel)

Sets the background and selection color of the widget.

- *FI_Color* **color2** () const

For back compatibility only.

- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `Fl_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `Fl_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `Fl_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`Fl_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`Fl_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`Fl_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`Fl_Widget` *widget, long arg, `Fl_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`Fl_Widget` *widget, void *arg=0, `Fl_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `Fl_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `Fl_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `Fl_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`Fl_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`Fl_Image` *img)

- Sets the image to use as part of the widget label when in the active state.*

 - int `image_bound` () const

Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const

Checks if this widget is a child of wgt.
- int `is_label_copied` () const

Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const

Gets the current label text.
- void `label` (const char *text)

Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)

Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()

Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)

Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const

Gets the label color.
- void `labelcolor` (`FL_Color` c)

Sets the label color.
- `FL_Font` `labelfont` () const

Gets the font to use.
- void `labelfont` (`FL_Font` f)

Sets the font to use.
- `FL_Fonsize` `labelsize` () const

Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)

Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const

Gets the label type.
- void `labeltype` (`FL_Labeltype` a)

Sets the label type.
- void `measure_label` (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const

Returns whether this widget needs a keyboard.
- void `needs_keyboard` (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int `output` () const

Returns if a widget is used for output only.
- `FL_Group` * `parent` () const

Returns a pointer to the parent widget.
- void `parent` (`FL_Group` *p)

Internal use only - "for hacks only".
- void `position` (int X, int Y)

Repositions the window or widget.
- void `redraw` ()

Schedules the drawing of the widget.
- void `redraw_label` ()

Schedules the drawing of the label.

- [FI_Color selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([FI_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [FI_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [FI_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const
Gets the user data for this widget.
- void [user_data](#) ([FI_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void [user_data](#) (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)

- Set the spacing between the label and the vertical edge of the widget.*

 - unsigned int **visible** () const
 - Returns whether a widget is visible.*
 - unsigned int **visible_focus** () const
 - Checks whether this widget has a visible focus.*
 - void **visible_focus** (int v)
 - Modifies keyboard focus navigation.*
 - int **visible_r** () const
 - Returns whether a widget and all its parents are visible.*
 - int **w** () const
 - Gets the widget width.*
 - **FL_When** **when** () const
 - Returns the conditions under which the callback is called.*
 - void **when** (uchar i)
 - Sets the flags used to decide when a callback is called.*
 - **FL_Window** * **window** () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
 - int **x** () const
 - Gets the widget position in its window.*
 - int **y** () const
 - Gets the widget position in its window.*
 - virtual **~FL_Widget** ()
 - Destroys the widget.*

Public Attributes

- **FL_Input** **input**

Protected Member Functions

- void **draw** () **FL_OVERRIDE**
- Draws the widget.*

Protected Member Functions inherited from **FL_Valuator**

- **FL_Valuator** (int X, int Y, int W, int H, const char *L)
- Creates a new **FL_Valuator** widget using the given position, size, and label string.*
- void **handle_drag** (double newvalue)
- Called during a drag operation, after an **FL_WHEN_CHANGED** event is received and before the callback.*
- void **handle_push** ()
- Stores the current value in the previous value.*
- void **handle_release** ()
- Called after an **FL_WHEN_RELEASE** event is received and before the callback.*
- int **horizontal** () const
- Tells if the valuator is an **FL_HORIZONTAL** one.*
- double **previous_value** () const
- Gets the previous floating point value before an event changed it.*
- void **set_value** (double v)
- Sets the current floating point value.*
- double **softclamp** (double)
- Clamps the value, but accepts v if the previous value is not already out of range.*

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Widget](#)

- static void **default_callback** ([FI_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPOP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.158.1 Detailed Description

The [FI_Value_Input](#) widget displays a numeric value.

The user can click in the text field and edit it - there is in fact a hidden [FI_Input](#) widget with type([FL_FLOAT_INPUT](#)) or type([FL_INT_INPUT](#)) in there - and when they hit return or tab the value updates to what they typed and the callback is done.

If [step\(\)](#) is non-zero and integral, then the range of numbers is limited to integers instead of floating point numbers. As well as displaying the value as an integer, typed input is also limited to integer values, even if the hidden [FI_Input](#) widget is of type([FL_FLOAT_INPUT](#)).

If [step\(\)](#) is non-zero, the user can also drag the mouse across the object and thus slide the value. The left button moves one [step\(\)](#) per pixel, the middle by 10 [step\(\)](#), and the right button by 100 * [step\(\)](#). It is therefore impossible to select text by dragging across it, although clicking can still move the insertion cursor.

If [step\(\)](#) is non-zero and integral, then the range of numbers are limited to integers instead of floating point values.



Figure 11.67 FI_Value_Input

See also

[FI_Widget::shortcut_label\(int\)](#)

11.158.2 Constructor & Destructor Documentation

11.158.2.1 FI_Value_Input()

```
Fl_Value_Input::Fl_Value_Input (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [FI_Value_Input](#) widget using the given position, size, and label string. The default boxtype is [FL_DOWN_BOX](#).

11.158.3 Member Function Documentation

11.158.3.1 cursor_color() [1/2]

```
Fl_Color Fl_Value_Input::cursor_color () const [inline]
```

Gets the color of the text cursor.
The text cursor is black by default.

11.158.3.2 `cursor_color()` [2/2]

```
void Fl_Value_Input::cursor_color (
    Fl_Color n) [inline]
```

Sets the color of the text cursor.
The text cursor is black by default.

11.158.3.3 `draw()`

```
void Fl_Value_Input::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.158.3.4 `handle()`

```
int Fl_Value_Input::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

11.158.3.5 `resize()`

```
void Fl_Value_Input::resize (
    int x,
    int y,
    int w,
    int h) [virtual]
```

Changes the size or position of the widget.

This is a virtual function so that the widget may implement its own handling of resizing. The default version does *not* call the `redraw()` method, but instead relies on the parent widget to do so because the parent may know a faster way to update the display, such as scrolling from the old position.

Some window managers under X11 call `resize()` a lot more often than needed. Please verify that the position or size of a widget did actually change before doing any extensive calculations.

`position(X, Y)` is a shortcut for `resize(X, Y, w(), h())`, and `size(W, H)` is a shortcut for `resize(x(), y(), W, H)`.

Parameters

in	<i>x,y</i>	new position relative to the parent window
in	<i>w,h</i>	new size

See also

[position\(int,int\)](#), [size\(int,int\)](#)

Reimplemented from [Fl_Widget](#).

11.158.3.6 `shortcut()` [1/2]

```
int Fl_Value_Input::shortcut () const [inline]
```

Returns the current shortcut key for the Input.

See also

[Fl_Value_Input::shortcut\(int\)](#)

11.158.3.7 `shortcut()` [2/2]

```
void Fl_Value_Input::shortcut (
    int s) [inline]
```

Sets the shortcut key to *s*.

Setting this overrides the use of '&' in the [label\(\)](#). The value is a bitwise OR of a key and a set of shift flags, for example `FL_ALT | 'a'`, `FL_ALT | (FL_F + 10)`, or just `'a'`. A value of 0 disables the shortcut.

The key can be any value returned by [Fl::event_key\(\)](#), but will usually be an ASCII letter. Use a lower-case letter unless you require the shift key to be held down.

The shift flags can be any set of values accepted by [Fl::event_state\(\)](#). If the bit is on that shift key must be pushed. Meta, Alt, Ctrl, and Shift must be off if they are not in the shift flags (zero for the other bits indicates a "don't care" setting).

11.158.3.8 `soft()`

```
char Fl_Value_Input::soft () const [inline]
```

If "soft" is turned on, the user is allowed to drag the value outside the range.

If they drag the value to one of the ends, let go, then grab again and continue to drag, they can get to any value.

The default is true.

11.158.3.9 `textcolor()`

```
Fl_Color Fl_Value_Input::textcolor () const [inline]
```

Gets the color of the text in the value box.

11.158.3.10 textfont() [1/2]

```
Fl_Font Fl_Value_Input::textfont () const [inline]
```

Gets the typeface of the text in the value box.

11.158.3.11 textfont() [2/2]

```
void Fl_Value_Input::textfont (
    Fl_Font s) [inline]
```

Sets the typeface of the text in the value box.

11.158.3.12 textsize() [1/2]

```
Fl_Fontsize Fl_Value_Input::textsize () const [inline]
```

Gets the size of the text in the value box.

11.158.3.13 textsize() [2/2]

```
void Fl_Value_Input::textsize (
    Fl_Fontsize s) [inline]
```

Sets the size of the text in the value box.

The documentation for this class was generated from the following files:

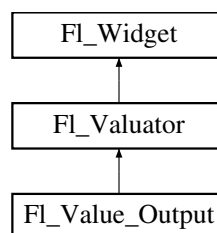
- Fl_Value_Input.H
- Fl_Value_Input.cxx

11.159 Fl_Value_Output Class Reference

The [Fl_Value_Output](#) widget displays a floating point value.

```
#include <Fl_Value_Output.H>
```

Inheritance diagram for [Fl_Value_Output](#):

**Public Member Functions**

- [Fl_Value_Output](#) (int *x*, int *y*, int *w*, int *h*, const char **l*=0)
Creates a new Fl_Value_Output widget using the given position, size, and label string.
- int [handle](#) (int) [FL_OVERRIDE](#)
Handles the specified event.
- uchar [soft](#) () const
If "soft" is turned on, the user is allowed to drag the value outside the range.
- void [soft](#) (uchar *s*)

If "soft" is turned on, the user is allowed to drag the value outside the range.

- [FI_Color](#) `textcolor` () const
Sets the color of the text in the value box.
- void `textcolor` ([FI_Color](#) s)
Gets the color of the text in the value box.
- [FI_Font](#) `textfont` () const
Gets the typeface of the text in the value box.
- void `textfont` ([FI_Font](#) s)
Sets the typeface of the text in the value box.
- [FI_Fontsize](#) `textsize` () const
Gets the size of the text in the value box.
- void `textsize` ([FI_Fontsize](#) s)

Public Member Functions inherited from [FI_Valuator](#)

- void `bounds` (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double `clamp` (double)
Clamps the passed value to the valuator range.
- virtual int `format` (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double `increment` (double, int)
Adds n times the step value to the passed value.
- double `maximum` () const
Gets the maximum value for the valuator.
- void `maximum` (double a)
Sets the maximum value for the valuator.
- double `minimum` () const
Gets the minimum value for the valuator.
- void `minimum` (double a)
Sets the minimum value for the valuator.
- void `precision` (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void `range` (double a, double b)
Sets the minimum and maximum values for the valuator.
- double `round` (double)
Round the passed value to the nearest step increment.
- double `step` () const
Gets or sets the step value.
- void `step` (double a, int b)
See double [FI_Valuator::step\(\)](#) const
- void `step` (double s)
See double [FI_Valuator::step\(\)](#) const.
- void `step` (int a)
See double [FI_Valuator::step\(\)](#) const
- double `value` () const
Gets the floating point(double) value.
- int `value` (double)
Sets the current value.
- `~FI_Valuator` () [FL_OVERRIDE](#)
Destructor is accessible despite protected constructor.

Public Member Functions inherited from [FI_Widget](#)

- void [_clear_fullscreen](#) ()
- void [_set_fullscreen](#) ()
- void [activate](#) ()
Activates the widget.
- unsigned int [active](#) () const
Returns whether the widget is active.
- int [active_r](#) () const
Returns whether the widget and all of its parents are active.
- [FI_Align](#) [align](#) () const
Gets the label alignment.
- void [align](#) ([FI_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [FI_Gl_Window](#) * [as_gl_window](#) ()
Returns an [FI_Gl_Window](#) pointer if this widget is an [FI_Gl_Window](#).
- virtual class [FI_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [FI_Group](#) * [as_group](#) ()
Returns an [FI_Group](#) pointer if this widget is an [FI_Group](#).
- virtual [FI_Group](#) const * [as_group](#) () const
- virtual [FI_Window](#) * [as_window](#) ()
Returns an [FI_Window](#) pointer if this widget is an [FI_Window](#).
- virtual [FI_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([FI_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [FI_Boxtype](#) [box](#) () const
Gets the box type of the widget.
- void [box](#) ([FI_Boxtype](#) new_box)
Sets the box type for the widget.
- [FI_Callback_p](#) [callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback](#) *cb, [FI_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([FI_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([FI_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([FI_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color color` () const

Gets the background color of the widget.
- void `color` (FL_Color bg)

Sets the background color of the widget.
- void `color` (FL_Color bg, FL_Color sel)

Sets the background and selection color of the widget.
- `FL_Color color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const FL_Widget *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image * deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image * deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (FL_Image &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (FL_Image *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (FL_Callback_Reason reason=FL_REASON_UNKNOWN)

Calls the widget callback function with default arguments.

- void `do_callback` (`FI_Widget` *widget, long arg, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FI_Widget` *widget, void *arg=0, `FI_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FI_Align`) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int `h` () const
Gets the widget height.
- virtual void `hide` ()
Makes a widget invisible.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FI_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FI_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FI_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FI_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FI_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FI_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FI_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FI_Color` c)
Sets the label color.
- `FI_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FI_Font` f)
Sets the font to use.
- `FI_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FI_Fonsize` pix)
Sets the font size in pixels.
- `FI_Labeltype` `labeltype` () const

- Gets the label type.*

 - void [labeltype](#) ([FL_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([FL_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [FL_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([FL_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from FI_Valuator

- [FI_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FI_Valuator](#) widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void [value_damage](#) ()
Asks for partial redraw.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)

Internal use only.

- void `x` (int `v`)

Internal use only.

- void `y` (int `v`)

Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback` (`Fl_Widget` *`widget`, void *`data`)

The default callback for all widgets that don't set a callback.

- static unsigned int `label_shortcut` (const char *`t`)

Returns the Unicode value of the '&x' shortcut in a given text.

- static int `test_shortcut` (const char *`t`, const bool `require_alt`=false)

Returns true if the given text `t` contains the entered '&x' shortcut.

Protected Types inherited from `Fl_Widget`

- enum {
`INACTIVE` = 1<<0 , `INVISIBLE` = 1<<1 , `OUTPUT` = 1<<2 , `NOBORDER` = 1<<3 ,
`FORCE_POSITION` = 1<<4 , `NON_MODAL` = 1<<5 , `SHORTCUT_LABEL` = 1<<6 , `CHANGED` = 1<<7
 ,
`OVERRIDE` = 1<<8 , `VISIBLE_FOCUS` = 1<<9 , `COPIED_LABEL` = 1<<10 , `CLIP_CHILDREN` = 1<<11
 ,
`MENU_WINDOW` = 1<<12 , `TOOLTIP_WINDOW` = 1<<13 , `MODAL` = 1<<14 , `NO_OVERLAY` = 1<<15
 ,
`GROUP_RELATIVE` = 1<<16 , `COPIED_TOOLTIP` = 1<<17 , `FULLSCREEN` = 1<<18 , `MAC_USE_ACCENTS_MENU`
 = 1<<19 ,
`NEEDS_KEYBOARD` = 1<<20 , `IMAGE_BOUND` = 1<<21 , `DEIMAGE_BOUND` = 1<<22 ,
`AUTO_DELETE_USER_DATA` = 1<<23 ,
`MAXIMIZED` = 1<<24 , `POPUP` = 1<<25 , `USERFLAG3` = 1<<29 , `USERFLAG2` = 1<<30 ,
`USERFLAG1` = 1<<31 }

flags possible values enumeration.

11.159.1 Detailed Description

The `Fl_Value_Output` widget displays a floating point value.

If `step()` is not zero, the user can adjust the value by dragging the mouse left and right. The left button moves one `step()` per pixel, the middle by 10 * `step()`, and the right button by 100 * `step()`.

This is much lighter-weight than `Fl_Value_Input` because it contains no text editing code or character buffer.

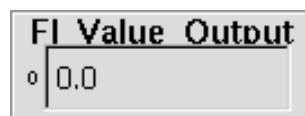


Figure 11.68 `Fl_Value_Output`

11.159.2 Constructor & Destructor Documentation

11.159.2.1 `Fl_Value_Output()`

```
Fl_Value_Output::Fl_Value_Output (
    int X,
    int Y,
    int W,
```

```
int H,
const char * l = 0)
```

Creates a new [Fl_Value_Output](#) widget using the given position, size, and label string.

The default boxtype is `FL_NO_BOX`.

Inherited destructor destroys the Valuator.

11.159.3 Member Function Documentation

11.159.3.1 draw()

```
void Fl_Value_Output::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implements [Fl_Widget](#).

11.159.3.2 handle()

```
int Fl_Value_Output::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Widget](#).

11.159.3.3 soft() [1/2]

```
uchar Fl_Value_Output::soft () const [inline]
```

If "soft" is turned on, the user is allowed to drag the value outside the range.

If they drag the value to one of the ends, let go, then grab again and continue to drag, they can get to any value. Default is one.

11.159.3.4 `soft()` [2/2]

```
void Fl_Value_Output::soft (
    uchar s) [inline]
```

If "soft" is turned on, the user is allowed to drag the value outside the range.

If they drag the value to one of the ends, let go, then grab again and continue to drag, they can get to any value. Default is one.

11.159.3.5 `textcolor()` [1/2]

```
Fl_Color Fl_Value_Output::textcolor () const [inline]
```

Sets the color of the text in the value box.

11.159.3.6 `textcolor()` [2/2]

```
void Fl_Value_Output::textcolor (
    Fl_Color s) [inline]
```

Gets the color of the text in the value box.

11.159.3.7 `textfont()` [1/2]

```
Fl_Font Fl_Value_Output::textfont () const [inline]
```

Gets the typeface of the text in the value box.

11.159.3.8 `textfont()` [2/2]

```
void Fl_Value_Output::textfont (
    Fl_Font s) [inline]
```

Sets the typeface of the text in the value box.

11.159.3.9 `textsize()`

```
Fl_Fontsize Fl_Value_Output::textsize () const [inline]
```

Gets the size of the text in the value box.

The documentation for this class was generated from the following files:

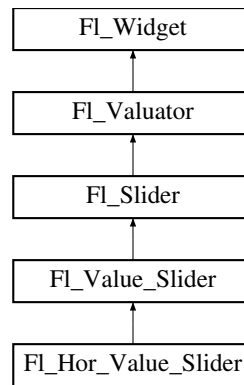
- `Fl_Value_Output.H`
- `Fl_Value_Output.cxx`

11.160 `Fl_Value_Slider` Class Reference

The `Fl_Value_Slider` widget is a `Fl_Slider` widget with a box displaying the current value.

```
#include <Fl_Value_Slider.H>
```

Inheritance diagram for `Fl_Value_Slider`:



Public Member Functions

- `Fl_Value_Slider` (int x, int y, int w, int h, const char *l=0)
Creates a new `Fl_Value_Slider` widget using the given position, size, and label string.
- int `handle` (int) `FL_OVERRIDE`
Handles the specified event.
- `Fl_Color` `textcolor` () const
Gets the color of the text in the value box.
- void `textcolor` (`Fl_Color` s)
Sets the color of the text in the value box.
- `Fl_Font` `textfont` () const
Gets the typeface of the text in the value box.
- void `textfont` (`Fl_Font` s)
Sets the typeface of the text in the value box.
- `Fl_Fonsize` `textsize` () const
Gets the size of the text in the value box.
- void `textsize` (`Fl_Fonsize` s)
Sets the size of the text in the value box.
- int `value_height` () const
Gets the height of the value box in pixels (vertical mode only).
- void `value_height` (int s)
Sets the height of the value box in pixels (vertical mode only).
- int `value_width` () const
Gets the width of the value box in pixels (horizontal mode only).
- void `value_width` (int s)
Sets the width of the value box in pixels (horizontal mode only).

Public Member Functions inherited from `Fl_Slider`

- void `bounds` (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- `Fl_Slider` (int X, int Y, int W, int H, const char *L=0)
Creates a new `Fl_Slider` widget using the given position, size, and label string.
- `Fl_Slider` (`uchar` t, int X, int Y, int W, int H, const char *L)
Creates a new `Fl_Slider` widget using the given type, position, size, and label string.
- int `scrollvalue` (int pos, int size, int first, int total)
Sets the size and position of the sliding knob in the box.
- `Fl_Boxtype` `slider` () const
Gets the slider box type.

- void **slider** ([FI_Boxtype](#) c)
Sets the slider box type.
- float **slider_size** () const
Get the dimensions of the moving piece of slider.
- void [slider_size](#) (double v)
Set the dimensions of the moving piece of slider.

Public Member Functions inherited from [FI_Valuator](#)

- void **bounds** (double a, double b)
Sets the minimum (a) and maximum (b) values for the valuator widget.
- double **clamp** (double)
Clamps the passed value to the valuator range.
- virtual int [format](#) (char *)
Uses internal rules to format the fields numerical value into the character array pointed to by the passed parameter.
- double [increment](#) (double, int)
Adds n times the step value to the passed value.
- double [maximum](#) () const
Gets the maximum value for the valuator.
- void [maximum](#) (double a)
Sets the maximum value for the valuator.
- double [minimum](#) () const
Gets the minimum value for the valuator.
- void [minimum](#) (double a)
Sets the minimum value for the valuator.
- void [precision](#) (int digits)
Sets the step value to $1.0 / 10^{\text{digits}}$.
- void [range](#) (double a, double b)
Sets the minimum and maximum values for the valuator.
- double [round](#) (double)
Round the passed value to the nearest step increment.
- double [step](#) () const
Gets or sets the step value.
- void **step** (double a, int b)
See double [FI_Valuator::step\(\)](#) const
- void **step** (double s)
See double [FI_Valuator::step\(\)](#) const.
- void **step** (int a)
See double [FI_Valuator::step\(\)](#) const
- double [value](#) () const
Gets the floating point(double) value.
- int [value](#) (double)
Sets the current value.
- [~FI_Valuator](#) () [FL_OVERRIDE](#)
Destructor is accessible despite protected constructor.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (FI_Align alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class FI_GL_Window * **as_gl_window** ()
Returns an FI_GL_Window pointer if this widget is an FI_GL_Window.
- virtual class FI_GL_Window const * **as_gl_window** () const
- virtual FI_Group * **as_group** ()
Returns an FI_Group pointer if this widget is an FI_Group.
- virtual FI_Group const * **as_group** () const
- virtual FI_Window * **as_window** ()
Returns an FI_Window pointer if this widget is an FI_Window.
- virtual FI_Window const * **as_window** () const
- void **bind_deimage** (FI_Image *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (FI_Image *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- **FI_Boxtype box** () const
Gets the box type of the widget.
- void **box** (FI_Boxtype new_box)
Sets the box type for the widget.
- **FI_Callback_p callback** () const
Gets the current callback function for the widget.
- void **callback** (FI_Callback *cb)
Sets the current callback function for the widget.
- void **callback** (FI_Callback *cb, FI_Callback_User_Data *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void **callback** (FI_Callback *cb, void *p)
Sets the current callback function and data for the widget.
- void **callback** (FI_Callback0 *cb)
Sets the current callback function for the widget.
- void **callback** (FI_Callback1 *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int **changed** () const

- Checks if the widget value changed since the last callback.*

 - void `clear_active` ()

Marks the widget as inactive without sending events or changing focus.
- void `clear_changed` ()

Marks the value of the widget as unchanged.
- void `clear_damage` (uchar c=0)

Clears or sets the damage flags.
- void `clear_output` ()

Sets a widget to accept input.
- void `clear_visible` ()

Hides the widget.
- void `clear_visible_focus` ()

Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const

Gets the background color of the widget.
- void `color` (`FL_Color` bg)

Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)

Sets the background and selection color of the widget.
- `FL_Color` `color2` () const

For back compatibility only.
- void `color2` (unsigned a)

For back compatibility only.
- int `contains` (const `FL_Widget` *w) const

Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)

Sets the current label.
- void `copy_tooltip` (const char *text)

Sets the current tooltip text.
- `uchar` `damage` () const

Returns non-zero if `draw()` needs to be called.
- void `damage` (uchar c)

Sets the damage bits for the widget.
- void `damage` (uchar c, int x, int y, int w, int h)

Sets the damage bits for an area inside the widget.
- int `damage_resize` (int, int, int, int)

Internal use only.
- void `deactivate` ()

Deactivates the widget.
- `FL_Image` * `deimage` ()

Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const

Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)

Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)

Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const

Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)

Calls the widget callback function with default arguments.

- void [do_callback](#) ([FL_Widget](#) *widget, long arg, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [do_callback](#) ([FL_Widget](#) *widget, void *arg=0, [FL_Callback_Reason](#) reason=[FL_REASON_UNKNOWN](#))
Calls the widget callback function with arbitrary arguments.
- void [draw_label](#) (int, int, int, int, [FL_Align](#)) const
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- int [h](#) () const
Gets the widget height.
- virtual void [hide](#) ()
Makes a widget invisible.
- int [horizontal_label_margin](#) ()
Get the spacing between the label and the horizontal edge of the widget.
- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const

- Gets the label type.*

 - void [labeltype](#) ([Fl_Labeltype](#) a)
- Sets the label type.*

 - void [measure_label](#) (int &ww, int &hh) const

Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const

Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)

Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const

Returns if a widget is used for output only.
- [Fl_Group](#) * [parent](#) () const

Returns a pointer to the parent widget.
- void [parent](#) ([Fl_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)

Changes the size or position of the widget.
- [Fl_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.

- `const char * tooltip () const`
Gets the current tooltip text.
- `void tooltip (const char *text)`
Sets the current tooltip text.
- `Fl_Window * top_window () const`
Returns a pointer to the top-level window for the widget.
- `Fl_Window * top_window_offset (int &xoff, int &yoff) const`
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type () const`
Gets the widget type.
- `void type (uchar t)`
Sets the widget type.
- `int use_accents_menu ()`
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- `void * user_data () const`
Gets the user data for this widget.
- `void user_data (Fl_Callback_User_Data *v, bool auto_free)`
Sets the user data for this widget.
- `void user_data (void *v)`
Sets the user data for this widget.
- `int vertical_label_margin ()`
Get the spacing between the label and the vertical edge of the widget.
- `void vertical_label_margin (int px)`
Set the spacing between the label and the vertical edge of the widget.
- `unsigned int visible () const`
Returns whether a widget is visible.
- `unsigned int visible_focus () const`
Checks whether this widget has a visible focus.
- `void visible_focus (int v)`
Modifies keyboard focus navigation.
- `int visible_r () const`
Returns whether a widget and all its parents are visible.
- `int w () const`
Gets the widget width.
- `Fl_When when () const`
Returns the conditions under which the callback is called.
- `void when (uchar i)`
Sets the flags used to decide when a callback is called.
- `Fl_Window * window () const`
Returns a pointer to the nearest parent window up the widget hierarchy.
- `int x () const`
Gets the widget position in its window.
- `int y () const`
Gets the widget position in its window.
- `virtual ~Fl_Widget ()`
Destroys the widget.

Protected Member Functions

- `void draw () FL_OVERRIDE`
Draws the widget.

Protected Member Functions inherited from [FI_Slider](#)

- void **draw** (int, int, int, int)
- int **handle** (int, int, int, int, int)

Protected Member Functions inherited from [FI_Valuator](#)

- [FI_Valuator](#) (int X, int Y, int W, int H, const char *L)
Creates a new [FI_Valuator](#) widget using the given position, size, and label string.
- void **handle_drag** (double newvalue)
Called during a drag operation, after an FL_WHEN_CHANGED event is received and before the callback.
- void **handle_push** ()
Stores the current value in the previous value.
- void **handle_release** ()
Called after an FL_WHEN_RELEASE event is received and before the callback.
- int **horizontal** () const
Tells if the valuator is an FL_HORIZONTAL one.
- double **previous_value** () const
Gets the previous floating point value before an event changed it.
- void **set_value** (double v)
Sets the current floating point value.
- double **softclamp** (double)
Clamps the value, but accepts v if the previous value is not already out of range.
- virtual void **value_damage** ()
Asks for partial redraw.

Protected Member Functions inherited from [FI_Widget](#)

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** ([FI_Boxtype](#) t, [FI_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** ([FI_Boxtype](#) t, int x, int y, int w, int h, [FI_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FI_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.

- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Widget

- static void **default_callback** (FI_Widget *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int **label_shortcut** (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int **test_shortcut** (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from FI_Widget

- enum {
INACTIVE = 1<<0 , **INVISIBLE** = 1<<1 , **OUTPUT** = 1<<2 , **NOBORDER** = 1<<3 ,
FORCE_POSITION = 1<<4 , **NON_MODAL** = 1<<5 , **SHORTCUT_LABEL** = 1<<6 , **CHANGED** = 1<<7
, **OVERRIDE** = 1<<8 , **VISIBLE_FOCUS** = 1<<9 , **COPIED_LABEL** = 1<<10 , **CLIP_CHILDREN** = 1<<11
, **MENU_WINDOW** = 1<<12 , **TOOLTIP_WINDOW** = 1<<13 , **MODAL** = 1<<14 , **NO_OVERLAY** = 1<<15
, **GROUP_RELATIVE** = 1<<16 , **COPIED_TOOLTIP** = 1<<17 , **FULLSCREEN** = 1<<18 , **MAC_USE_ACCENTS_MENU**
= 1<<19 ,
NEEDS_KEYBOARD = 1<<20 , **IMAGE_BOUND** = 1<<21 , **DEIMAGE_BOUND** = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , **POPUP** = 1<<25 , **USERFLAG3** = 1<<29 , **USERFLAG2** = 1<<30 ,
USERFLAG1 = 1<<31 }
flags possible values enumeration.

11.160.1 Detailed Description

The **FI_Value_Slider** widget is a **FI_Slider** widget with a box displaying the current value.

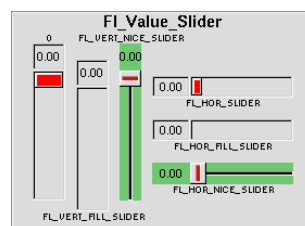


Figure 11.69 FI_Value_Slider

11.160.2 Constructor & Destructor Documentation

11.160.2.1 Fl_Value_Slider()

```
Fl_Value_Slider::Fl_Value_Slider (
    int X,
    int Y,
    int W,
    int H,
    const char * l = 0)
```

Creates a new [Fl_Value_Slider](#) widget using the given position, size, and label string. The default boxtype is FL_DOWN_BOX.

11.160.3 Member Function Documentation

11.160.3.1 draw()

```
void Fl_Value_Slider::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw(); // calls Fl_Scrollbar::draw()
```

Reimplemented from [Fl_Slider](#).

11.160.3.2 handle()

```
int Fl_Value_Slider::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Slider](#).

11.160.3.3 value_height() [1/2]

```
int Fl_Value_Slider::value_height () const [inline]
```

Gets the height of the value box in pixels (vertical mode only).

Since

1.4.0

11.160.3.4 value_height() [2/2]

```
void Fl_Value_Slider::value_height (  
    int s) [inline]
```

Sets the height of the value box in pixels (vertical mode only).

Limited range checking is applied but drawing errors may occur if the size *s* is set too high or too low, particularly if the widget is resized (later).

The programmer is responsible for setting sensible values and widget sizes.

The default value set by the constructor is 25.

Parameters

in	s	new height of the value box

Since

1.4.0

11.160.3.5 value_width() [1/2]

```
int Fl_Value_Slider::value_width () const [inline]
```

Gets the width of the value box in pixels (horizontal mode only).

Since

1.4.0

11.160.3.6 value_width() [2/2]

```
void Fl_Value_Slider::value_width (  
    int s) [inline]
```

Sets the width of the value box in pixels (horizontal mode only).

Limited range checking is applied but drawing errors may occur if the size *s* is set too high or too low, particularly if the widget is resized (later).

The programmer is responsible for setting sensible values and widget sizes.

The default value set by the constructor is 35.

Parameters

in	s	new width of the value box

Since

1.4.0

The documentation for this class was generated from the following files:

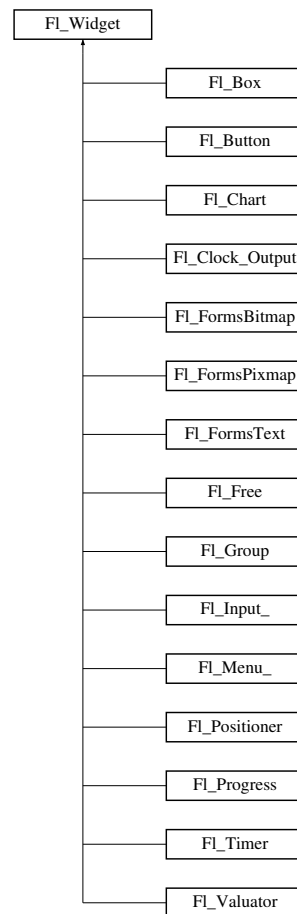
- Fl_Value_Slider.H
- Fl_Value_Slider.cxx

11.161 FL_Widget Class Reference

[FL_Widget](#) is the base class for all widgets in FLTK.

```
#include <Fl_Widget.H>
```

Inheritance diagram for FL_Widget:



Public Member Functions

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FL_Align align** () const
Gets the label alignment.
- void **align** (FL_Align alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FL_Gl_Window** * **as_gl_window** ()
Returns an FL_Gl_Window pointer if this widget is an FL_Gl_Window.

- virtual class [FL_GI_Window](#) const * **as_gl_window** () const
- virtual [FL_Group](#) * **as_group** ()
 - Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).*
- virtual [FL_Group](#) const * **as_group** () const
- virtual [FL_Window](#) * **as_window** ()
 - Returns an [FL_Window](#) pointer if this widget is an [FL_Window](#).*
- virtual [FL_Window](#) const * **as_window** () const
- void **bind_deimage** ([FL_Image](#) *img)
 - Sets the image to use as part of the widget label when in the inactive state.*
- void **bind_deimage** (int f)
 - Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.*
- void **bind_image** ([FL_Image](#) *img)
 - Sets the image to use as part of the widget label when in the active state.*
- void **bind_image** (int f)
 - Bind the image to the widget, so the widget will delete the image when it is no longer needed.*
- [FL_Boxtype](#) **box** () const
 - Gets the box type of the widget.*
- void **box** ([FL_Boxtype](#) new_box)
 - Sets the box type for the widget.*
- [FL_Callback_p](#) **callback** () const
 - Gets the current callback function for the widget.*
- void **callback** ([FL_Callback](#) *cb)
 - Sets the current callback function for the widget.*
- void **callback** ([FL_Callback](#) *cb, [FL_Callback_User_Data](#) *p, bool auto_free)
 - Sets the current callback function and managed user data for the widget.*
- void **callback** ([FL_Callback](#) *cb, void *p)
 - Sets the current callback function and data for the widget.*
- void **callback** ([FL_Callback0](#) *cb)
 - Sets the current callback function for the widget.*
- void **callback** ([FL_Callback1](#) *cb, long p=0)
 - Sets the current callback function for the widget.*
- unsigned int **changed** () const
 - Checks if the widget value changed since the last callback.*
- void **clear_active** ()
 - Marks the widget as inactive without sending events or changing focus.*
- void **clear_changed** ()
 - Marks the value of the widget as unchanged.*
- void **clear_damage** (uchar c=0)
 - Clears or sets the damage flags.*
- void **clear_output** ()
 - Sets a widget to accept input.*
- void **clear_visible** ()
 - Hides the widget.*
- void **clear_visible_focus** ()
 - Disables keyboard focus navigation with this widget.*
- [FL_Color](#) **color** () const
 - Gets the background color of the widget.*
- void **color** ([FL_Color](#) bg)
 - Sets the background color of the widget.*
- void **color** ([FL_Color](#) bg, [FL_Color](#) sel)
 - Sets the background and selection color of the widget.*

- `FL_Color color2 () const`
For back compatibility only.
- `void color2 (unsigned a)`
For back compatibility only.
- `int contains (const FL_Widget *w) const`
Checks if w is a child of this widget.
- `void copy_label (const char *new_label)`
Sets the current label.
- `void copy_tooltip (const char *text)`
Sets the current tooltip text.
- `uchar damage () const`
Returns non-zero if draw() needs to be called.
- `void damage (uchar c)`
Sets the damage bits for the widget.
- `void damage (uchar c, int x, int y, int w, int h)`
Sets the damage bits for an area inside the widget.
- `int damage_resize (int, int, int, int)`
Internal use only.
- `void deactivate ()`
Deactivates the widget.
- `FL_Image * deimage ()`
Gets the image that is used as part of the widget label when in the inactive state.
- `const FL_Image * deimage () const`
Gets the image that is used as part of the widget label when in the inactive state.
- `void deimage (FL_Image &img)`
Sets the image to use as part of the widget label when in the inactive state.
- `void deimage (FL_Image *img)`
Sets the image to use as part of the widget label when in the inactive state.
- `int deimage_bound () const`
Returns whether the inactive image is managed by the widget.
- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with default arguments.
- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`
Calls the widget callback function with arbitrary arguments.
- `virtual void draw ()=0`
Draws the widget.
- `void draw_label (int, int, int, int, FL_Align) const`
Draws the label in an arbitrary bounding box with an arbitrary alignment.
- `int h () const`
Gets the widget height.
- `virtual int handle (int event)`
Handles the specified event.
- `virtual void hide ()`
Makes a widget invisible.
- `int horizontal_label_margin ()`
Get the spacing between the label and the horizontal edge of the widget.
- `void horizontal_label_margin (int px)`
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image * image ()`

- Gets the image that is used as part of the widget label when in the active state.*

 - `const FL_Image * image () const`
- Gets the image that is used as part of the widget label when in the active state.*

 - `void image (FL_Image &img)`
- Sets the image to use as part of the widget label when in the active state.*

 - `void image (FL_Image *img)`
- Sets the image to use as part of the widget label when in the active state.*

 - `int image_bound () const`
- Returns whether the image is managed by the widget.*

 - `int inside (const FL_Widget *wgt) const`
- Checks if this widget is a child of wgt.*

 - `int is_label_copied () const`
- Returns whether the current label was assigned with copy_label().*

 - `const char * label () const`
- Gets the current label text.*

 - `void label (const char *text)`
- Sets the current label pointer.*

 - `void label (FL_Labeltype a, const char *b)`
- Shortcut to set the label text and type in one call.*

 - `int label_image_spacing ()`
- Return the gap size between the label and the image.*

 - `void label_image_spacing (int gap)`
- Set the gap between the label and the image in pixels.*

 - `FL_Color labelcolor () const`
- Gets the label color.*

 - `void labelcolor (FL_Color c)`
- Sets the label color.*

 - `FL_Font labelfont () const`
- Gets the font to use.*

 - `void labelfont (FL_Font f)`
- Sets the font to use.*

 - `FL_Fonsize labelsz () const`
- Gets the font size in pixels.*

 - `void labelsz (FL_Fonsize pix)`
- Sets the font size in pixels.*

 - `FL_Labeltype labeltype () const`
- Gets the label type.*

 - `void labeltype (FL_Labeltype a)`
- Sets the label type.*

 - `void measure_label (int &ww, int &hh) const`
- Sets width ww and height hh accordingly with the label size.*

 - `bool needs_keyboard () const`
- Returns whether this widget needs a keyboard.*

 - `void needs_keyboard (bool needs)`
- Sets whether this widget needs a keyboard.*

 - `unsigned int output () const`
- Returns if a widget is used for output only.*

 - `FL_Group * parent () const`
- Returns a pointer to the parent widget.*

 - `void parent (FL_Group *p)`
- Internal use only - "for hacks only".*

- void [position](#) (int X, int Y)
Repositions the window or widget.
- void [redraw](#) ()
Schedules the drawing of the widget.
- void [redraw_label](#) ()
Schedules the drawing of the label.
- virtual void [resize](#) (int x, int y, int w, int h)
Changes the size or position of the widget.
- [Fl_Color](#) [selection_color](#) () const
Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)
Sets the selection color.
- void [set_active](#) ()
Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()
Marks the value of the widget as changed.
- void [set_output](#) ()
Sets a widget to output only.
- void [set_visible](#) ()
Makes the widget visible.
- void [set_visible_focus](#) ()
Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()
Makes a widget visible.
- void [size](#) (int W, int H)
Changes the size of the widget.
- int [take_focus](#) ()
Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const
Returns if the widget is able to take events.
- int [test_shortcut](#) ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const
Gets the current tooltip text.
- void [tooltip](#) (const char *text)
Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const
Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const
Gets the widget type.
- void [type](#) ([uchar](#) t)
Sets the widget type.
- int [use_accents_menu](#) ()
Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.
- void * [user_data](#) () const

- Gets the user data for this widget.*
- void **user_data** ([Fl_Callback_User_Data](#) *v, bool auto_free)
- Sets the user data for this widget.*
- void **user_data** (void *v)
- Sets the user data for this widget.*
- int **vertical_label_margin** ()
- Get the spacing between the label and the vertical edge of the widget.*
- void **vertical_label_margin** (int px)
- Set the spacing between the label and the vertical edge of the widget.*
- unsigned int **visible** () const
- Returns whether a widget is visible.*
- unsigned int **visible_focus** () const
- Checks whether this widget has a visible focus.*
- void **visible_focus** (int v)
- Modifies keyboard focus navigation.*
- int **visible_r** () const
- Returns whether a widget and all its parents are visible.*
- int **w** () const
- Gets the widget width.*
- [Fl_When](#) **when** () const
- Returns the conditions under which the callback is called.*
- void **when** (uchar i)
- Sets the flags used to decide when a callback is called.*
- [Fl_Window](#) * **window** () const
- Returns a pointer to the nearest parent window up the widget hierarchy.*
- int **x** () const
- Gets the widget position in its window.*
- int **y** () const
- Gets the widget position in its window.*
- virtual [~Fl_Widget](#) ()
- Destroys the widget.*

Static Public Member Functions

- static void **default_callback** ([Fl_Widget](#) *widget, void *data)
- The default callback for all widgets that don't set a callback.*
- static unsigned int **label_shortcut** (const char *t)
- Returns the Unicode value of the '&x' shortcut in a given text.*
- static int **test_shortcut** (const char *, const bool require_alt=false)
- Returns true if the given text t contains the entered '&x' shortcut.*

Protected Types

- enum {
- [INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
- [FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
- ,
- [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
- ,
- [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
- ,
- [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
- = 1<<19 ,

```
NEEDS_KEYBOARD = 1<<20 , IMAGE_BOUND = 1<<21 , DEIMAGE_BOUND = 1<<22 ,
AUTO_DELETE_USER_DATA = 1<<23 ,
MAXIMIZED = 1<<24 , POPUP = 1<<25 , USERFLAG3 = 1<<29 , USERFLAG2 = 1<<30 ,
USERFLAG1 = 1<<31 }
```

flags possible values enumeration.

Protected Member Functions

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Friends

- class **FI_Group**

11.161.1 Detailed Description

FI_Widget is the base class for all widgets in FLTK.

You can't create one of these because the constructor is not public. However you can subclass it.

All "property" accessing methods, such as **color()**, **parent()**, or **argument()** are implemented as trivial inline functions and thus are as fast and small as accessing fields in a structure. Unless otherwise noted, the property setting methods such as **color(n)** or **label(s)** are also trivial inline functions, even if they change the widget's appearance. It is up to the user code to call **redraw()** after these.

11.161.2 Member Enumeration Documentation

11.161.2.1 anonymous enum

anonymous enum [protected]

flags possible values enumeration.

See [activate\(\)](#), [output\(\)](#), [visible\(\)](#), [changed\(\)](#), [set_visible_focus\(\)](#)

Enumerator

INACTIVE	the widget can't receive focus, and is disabled but potentially visible
INVISIBLE	the widget is not drawn, but can receive a few special events
OUTPUT	for output only
NOBORDER	don't draw a decoration (FI_Window)
FORCE_POSITION	don't let the window manager position the window (FI_Window)
NON_MODAL	this is a hovering toolbar window (FI_Window)
SHORTCUT_LABEL	the label contains a shortcut we need to draw
CHANGED	the widget value changed
OVERRIDE	position window on top (FI_Window)
VISIBLE_FOCUS	accepts keyboard focus navigation if the widget can have the focus
COPIED_LABEL	the widget label is internally copied, its destruction is handled by the widget
CLIP_CHILDREN	all drawing within this widget will be clipped (FI_Group)
MENU_WINDOW	a temporary popup window, dismissed by clicking outside (FI_Window)
TOOLTIP_WINDOW	a temporary popup, transparent to events, and dismissed easily (FI_Window)
MODAL	a window blocking input to all other windows (FI_Window)
NO_OVERLAY	window not using a hardware overlay plane (FI_Menu_Window)
GROUP_RELATIVE	Reserved, not implemented. DO NOT USE.
COPIED_TOOLTIP	the widget tooltip is internally copied, its destruction is handled by the widget
FULLSCREEN	a fullscreen window (FI_Window)
MAC_USE_ACCENTS_MENU	On the macOS platform, pressing and holding a key on the keyboard opens an accented-character menu window (FI_Input , FI_Text_Editor)
NEEDS_KEYBOARD	set on touch screen devices if a widget needs a keyboard when it gets the focus. Reserved, not yet used in 1.4.0. See also FI_Widget::needs_keyboard()
IMAGE_BOUND	binding the image to the widget will transfer ownership, so that the widget will delete the image when it is no longer needed
DEIMAGE_BOUND	bind the inactive image to the widget, so the widget deletes the image when it is no longer needed
AUTO_DELETE_USER_DATA	automatically call <code>delete</code> on the <code>user_data</code> pointer when destroying this widget; if set, <code>user_data</code> must point to a class derived from the class FI_Callback_User_Data
MAXIMIZED	a maximized FI_Window
POPUP	popup window (i.e., positioned relatively to another mapped window)
USERFLAG3	reserved for 3rd party extensions
USERFLAG2	reserved for 3rd party extensions
USERFLAG1	reserved for 3rd party extensions

11.161.3 Constructor & Destructor Documentation

11.161.3.1 Fl_Widget()

```
Fl_Widget::Fl_Widget (
    int x,
    int y,
    int w,
    int h,
    const char * label = 0L) [protected]
```

Creates a widget at the given position and size.

The [Fl_Widget](#) is a protected constructor, but all derived widgets have a matching public constructor. It takes a value for [x\(\)](#), [y\(\)](#), [w\(\)](#), [h\(\)](#), and an optional value for [label\(\)](#).

Parameters

in	<i>x,y</i>	the position of the widget relative to the enclosing window
in	<i>w,h</i>	size of the widget in pixels
in	<i>label</i>	optional text for the widget label

11.161.3.2 ~Fl_Widget()

```
Fl_Widget::~~Fl_Widget () [virtual]
```

Destroys the widget.

Destroys the widget, taking care of throwing focus before if any.

Destroying single widgets is not very common. You almost always want to destroy the parent group instead, which will destroy all of the child widgets and groups in that group.

Since

FLTK 1.3, the widget's destructor removes the widget from its parent group, if it is member of a group.

Destruction removes the widget from any parent group! And groups when destroyed destroy all their children. This is convenient and fast.

11.161.4 Member Function Documentation

11.161.4.1 activate()

```
void Fl_Widget::activate ()
```

Activates the widget.

Changing this value will send FL_ACTIVATE to the widget if [active_r\(\)](#) is true.

See also

[active\(\)](#), [active_r\(\)](#), [deactivate\(\)](#)

11.161.4.2 active()

```
unsigned int Fl_Widget::active () const [inline]
```

Returns whether the widget is active.

Return values

0	if the widget is inactive

See also

[active_r\(\)](#), [activate\(\)](#), [deactivate\(\)](#)

11.161.4.3 active_r()

```
int Fl_Widget::active_r () const
```

Returns whether the widget and all of its parents are active.

Return values

0	if this or any of the parent widgets are inactive

See also

[active\(\)](#), [activate\(\)](#), [deactivate\(\)](#)

11.161.4.4 align() [1/2]

```
Fl_Align Fl_Widget::align () const [inline]
```

Gets the label alignment.

Returns

label alignment

See also

[label\(\)](#), [align\(Fl_Align\)](#), [Fl_Align](#)

11.161.4.5 align() [2/2]

```
void Fl_Widget::align (
    Fl_Align alignment) [inline]
```

Sets the label alignment.

This controls how the label is displayed next to or inside the widget. The default value is FL_ALIGN_CENTER, which centers the label inside the widget.

Parameters

in	<i>alignment</i>	new label alignment

See also

[align\(\)](#), [Fl_Align](#)

11.161.4.6 argument() [1/2]

```
long Fl_Widget::argument () const [inline]
```

Gets the current user data (long) argument that is passed to the callback function.

Note

On platforms with `sizeof(long) < sizeof(void*)`, particularly on Windows 64-bit platforms, this method can truncate stored addresses (void*) to the size of a long value. Use with care and only if you are sure that the stored user_data value fits in a long value because it was stored with [argument\(long\)](#) or another method using only long values. You may want to use [user_data\(\)](#) instead.

See also

[user_data\(\)](#)

11.161.4.7 argument() [2/2]

```
void Fl_Widget::argument (
    long v) [inline]
```

Sets the current user data (long) argument that is passed to the callback function.

See also

[argument\(\)](#)

11.161.4.8 as_gl_window()

```
virtual class Fl_Gl_Window * Fl_Widget::as_gl_window () [inline], [virtual]
```

Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).

Use this method if you have a widget (pointer) and need to know whether this widget is derived from [Fl_Gl_Window](#). If it returns non-NULL, then the widget in question is derived from [Fl_Gl_Window](#).

Return values

<i>NULL</i>	if this widget is not derived from Fl_Gl_Window .

Note

This method is provided to avoid `dynamic_cast`.

See also

[Fl_Widget::as_group\(\)](#), [Fl_Widget::as_window\(\)](#)

Reimplemented in [Fl_Gl_Window](#).

11.161.4.9 as_group()

```
virtual Fl_Group * Fl_Widget::as_group () [inline], [virtual]
```

Returns an [Fl_Group](#) pointer if this widget is an [Fl_Group](#).

Use this method if you have a widget (pointer) and need to know whether this widget is derived from [Fl_Group](#). If it returns non-NULL, then the widget in question is derived from [Fl_Group](#), and you can use the returned pointer to access its children or other [Fl_Group](#)-specific methods.

Example:

```
void my_callback (Fl_Widget *w, void *) {
    Fl_Group *g = w->as_group();
    if (g)
        printf ("This group has %d children\n", g->children());
    else
        printf ("This widget is not a group!\n");
}
```

Return values

<i>NULL</i>	if this widget is not derived from Fl_Group .

Note

This method is provided to avoid `dynamic_cast`.

See also

[Fl_Widget::as_window\(\)](#), [Fl_Widget::as_gl_window\(\)](#)

Reimplemented in [Fl_Group](#).

11.161.4.10 as_window()

```
virtual Fl\_Window * Fl_Widget::as_window () [inline], [virtual]
```

Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).

Use this method if you have a widget (pointer) and need to know whether this widget is derived from [Fl_Window](#). If it returns non-NULL, then the widget in question is derived from [Fl_Window](#), and you can use the returned pointer to access its children or other [Fl_Window](#)-specific methods.

Return values

<i>NULL</i>	if this widget is not derived from Fl_Window .

Note

This method is provided to avoid `dynamic_cast`.

See also

[Fl_Widget::as_group\(\)](#), [Fl_Widget::as_gl_window\(\)](#)

Reimplemented in [Fl_Window](#).

11.161.4.11 bind_deimage() [1/2]

```
void Fl_Widget::bind_deimage (
    Fl\_Image * img)
```

Sets the image to use as part of the widget label when in the inactive state.

Parameters

<i>in</i>	<i>img</i>	the new image for the deactivated widget

Note

The image will be bound to the widget. When the widget is deleted, the image will be deleted as well.

See also

void [deimage\(\[Fl_Image*\]\(#\) img\)](#)

11.161.4.12 bind_deimage() [2/2]

```
void Fl_Widget::bind_deimage (
    int f) [inline]
```

Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.

Parameters

<i>f</i>	1: mark the image as bound, 0: mark the image as managed by the user

See also

[deimage_bound\(\)](#), [bind_image\(\)](#)

11.161.4.13 bind_image() [1/2]

```
void Fl_Widget::bind_image (  
    Fl\_Image * img)
```

Sets the image to use as part of the widget label when in the active state.

The image will be bound to the widget. When the widget is deleted, the image will be deleted as well.

Calling [bind_image\(\)](#) with a new image will delete the old image if it was bound, and then set the new image, and bind that. If old and new image are the same, nothing happens.

Calling [bind_image\(\)](#) with NULL will delete the old image if it was bound and not set a new image.

Parameters

in	<i>img</i>	the new image for the label

See also

void [image\(Fl_Image* img\)](#)

11.161.4.14 bind_image() [2/2]

```
void Fl_Widget::bind_image (
    int f) [inline]
```

Bind the image to the widget, so the widget will delete the image when it is no longer needed.

Parameters

<i>f</i>	1: mark the image as bound, 0: mark the image as managed by the user

See also

[image_bound\(\)](#), [bind_deimage\(\)](#)

11.161.4.15 box() [1/2]

```
Fl_Boxtype Fl_Widget::box (
    void ) const [inline]
```

Gets the box type of the widget.

Returns

the current box type

See also

[box\(Fl_Boxtype\)](#), [Fl_Boxtype](#)

11.161.4.16 box() [2/2]

```
void Fl_Widget::box (
    Fl_Boxtype new_box) [inline]
```

Sets the box type for the widget.

This identifies a routine that draws the background of the widget. See [Fl_Boxtype](#) for the available types. The default depends on the widget, but is usually [FL_NO_BOX](#) or [FL_UP_BOX](#).

Parameters

in	<i>new_box</i>	the new box type

See also

[box\(\)](#), [Fl_Boxtype](#)

11.161.4.17 callback() [1/6]

```
Fl_Callback_p Fl_Widget::callback () const [inline]
```

Gets the current callback function for the widget.

Each widget has a single callback.

Returns

current callback

11.161.4.18 callback() [2/6]

```
void Fl_Widget::callback (
    Fl_Callback * cb) [inline]
```

Sets the current callback function for the widget.
Each widget has a single callback.

Parameters

in	<i>cb</i>	new callback

11.161.4.19 callback() [3/6]

```
void Fl_Widget::callback (
    Fl_Callback * cb,
    Fl_Callback_User_Data * p,
    bool auto_free) [inline]
```

Sets the current callback function and managed user data for the widget.
Setting `auto_free` will transfer ownership of the callback user data to the widget. Deleting the widget will then also delete the user data.

Parameters

in	<i>cb</i>	new callback
in	<i>p</i>	user data
in	<i>auto_free</i>	if set, the widget will free user data when destroyed

11.161.4.20 callback() [4/6]

```
void Fl_Widget::callback (
    Fl_Callback * cb,
    void * p) [inline]
```

Sets the current callback function and data for the widget.
Each widget has a single callback.

Parameters

in	<i>cb</i>	new callback
in	<i>p</i>	user data

11.161.4.21 callback() [5/6]

```
void Fl_Widget::callback (
    Fl_Callback0 * cb) [inline]
```

Sets the current callback function for the widget.
Each widget has a single callback.

Parameters

in	<i>cb</i>	new callback

11.161.4.22 callback() [6/6]

```
void Fl_Widget::callback (
    Fl_Callback1 * cb,
    long p = 0) [inline]
```

Sets the current callback function for the widget.
Each widget has a single callback.

Parameters

in	<i>cb</i>	new callback
in	<i>p</i>	user data

11.161.4.23 changed()

```
unsigned int Fl_Widget::changed () const [inline]
```

Checks if the widget value changed since the last callback.

"Changed" is a flag that is turned on when the user changes the value stored in the widget. This is only used by subclasses of [Fl_Widget](#) that store values, but is in the base class so it is easier to scan all the widgets in a panel and [do_callback\(\)](#) on the changed ones in response to an "OK" button.

Most widgets turn this flag off when they do the callback, and when the program sets the stored value.

Note

[do_callback\(\)](#) turns this flag off after the callback.

Return values

0	if the value did not change

See also

[set_changed\(\)](#), [clear_changed\(\)](#)

[do_callback\(Fl_Widget *widget, void *data\)](#)

11.161.4.24 clear_active()

```
void Fl_Widget::clear_active () [inline]
```

Marks the widget as inactive without sending events or changing focus.

This is mainly for specialized use, for normal cases you want [deactivate\(\)](#).

See also

[deactivate\(\)](#)

11.161.4.25 clear_changed()

```
void Fl_Widget::clear_changed () [inline]
```

Marks the value of the widget as unchanged.

See also

[changed\(\)](#), [set_changed\(\)](#)

11.161.4.26 clear_damage()

```
void Fl_Widget::clear_damage (
    uchar c = 0) [inline]
```

Clears or sets the damage flags.

Damage flags are cleared when parts of the widget drawing is repaired.

The optional argument `c` specifies the bits that **are set** after the call (default: 0) and **not** the bits that are cleared!

Note

Therefore it is possible to set damage bits with this method, but this should be avoided. Use [damage\(uchar\)](#) instead.

Parameters

in	c	new bitmask of damage flags (default: 0)

See also

[damage\(uchar\)](#), [damage\(\)](#)

11.161.4.27 clear_output()

```
void Fl_Widget::clear_output () [inline]
```

Sets a widget to accept input.

See also

[set_output\(\)](#), [output\(\)](#)

11.161.4.28 clear_visible()

```
void Fl_Widget::clear_visible () [inline]
```

Hides the widget.

You must still redraw the parent to see a change in the window. Normally you want to use the [hide\(\)](#) method instead.

11.161.4.29 clear_visible_focus()

```
void Fl_Widget::clear_visible_focus () [inline]
```

Disables keyboard focus navigation with this widget.

Normally, all widgets participate in keyboard focus navigation.

See also

[set_visible_focus\(\)](#), [visible_focus\(\)](#), [visible_focus\(int\)](#)

11.161.4.30 color() [1/3]

```
Fl_Color Fl_Widget::color (
    void ) const [inline]
```

Gets the background color of the widget.

Returns

current background color

See also

[color\(Fl_Color\)](#), [color\(Fl_Color, Fl_Color\)](#)

11.161.4.31 color() [2/3]

```
void Fl_Widget::color (
    Fl_Color bg) [inline]
```

Sets the background color of the widget.

The color is passed to the box routine. The color is either an index into an internal table of RGB colors or an RGB color value generated using [fl_rgb_color\(\)](#).

The default for most widgets is FL_BACKGROUND_COLOR. Use [Fl::set_color\(\)](#) to redefine colors in the color map.

Parameters

in	<i>bg</i>	background color

See also

[color\(\)](#), [color\(Fl_Color, Fl_Color\)](#), [selection_color\(Fl_Color\)](#)

11.161.4.32 color() [3/3]

```
void Fl_Widget::color (
    Fl_Color bg,
    Fl_Color sel) [inline]
```

Sets the background and selection color of the widget.

The two color form sets both the background and selection colors.

Parameters

in	<i>bg</i>	background color
in	<i>sel</i>	selection color

See also

[color\(unsigned\)](#), [selection_color\(unsigned\)](#)

11.161.4.33 color2() [1/2]

```
Fl_Color Fl_Widget::color2 () const [inline]
```

For back compatibility only.

Deprecated Use [selection_color\(\)](#) instead.

11.161.4.34 color2() [2/2]

```
void Fl_Widget::color2 (
    unsigned a) [inline]
```

For back compatibility only.

Deprecated Use [selection_color\(unsigned\)](#) instead.

11.161.4.35 contains()

```
int Fl_Widget::contains (
    const Fl_Widget * w) const
```

Checks if w is a child of this widget.

Parameters

<code>in</code>	<code>w</code>	potential child widget

Returns

Returns 1 if `w` is a child of this widget, or is equal to this widget. Returns 0 if `w` is NULL.

11.161.4.36 `copy_label()`

```
void Fl_Widget::copy_label (
    const char * new_label)
```

Sets the current label.

Unlike [label\(\)](#), this method allocates a copy of the label string instead of using the original string pointer.

The internal copy will automatically be freed whenever you assign a new label or when the widget is destroyed.

Parameters

<code>in</code>	<code>new_label</code>	the new label text

See also

[label\(\)](#)

11.161.4.37 `copy_tooltip()`

```
void Fl_Widget::copy_tooltip (
    const char * text)
```

Sets the current tooltip text.

Unlike [tooltip\(\)](#), this method allocates a copy of the tooltip string instead of using the original string pointer.

The internal copy will automatically be freed whenever you assign a new tooltip or when the widget is destroyed.

If no tooltip is set, the tooltip of the parent is inherited. Setting a tooltip for a group and setting no tooltip for a child will show the group's tooltip instead. To avoid this behavior, you can set the child's tooltip to an empty string ("").

Parameters

<code>in</code>	<code>text</code>	New tooltip text (an internal copy is made and managed)

See also

[tooltip\(const char*\)](#), [tooltip\(\)](#)

11.161.4.38 `damage()` [1/3]

```
uchar Fl_Widget::damage () const [inline]
```

Returns non-zero if [draw\(\)](#) needs to be called.

The damage value is actually a bit field that the widget subclass can use to figure out what parts to draw.

Returns

a bitmap of flags describing the kind of damage to the widget

See also

[damage\(uchar\)](#), [clear_damage\(uchar\)](#)

11.161.4.39 damage() [2/3]

```
void Fl_Widget::damage (
    uchar c)
```

Sets the damage bits for the widget.

Setting damage bits will schedule the widget for the next redraw.

Parameters

in	<i>c</i>	bitmask of flags to set

See also

[damage\(\)](#), [clear_damage\(uchar\)](#)

11.161.4.40 damage() [3/3]

```
void Fl_Widget::damage (
    uchar c,
    int x,
    int y,
    int w,
    int h)
```

Sets the damage bits for an area inside the widget.
Setting damage bits will schedule the widget for the next redraw.

Parameters

in	<i>c</i>	bitmask of flags to set
in	<i>x,y,w,h</i>	size of damaged area

See also

[damage\(\)](#), [clear_damage\(uchar\)](#)

11.161.4.41 deactivate()

```
void Fl_Widget::deactivate ()
```

Deactivates the widget.

Inactive widgets will be drawn "grayed out", e.g. with less contrast than the active widget. Inactive widgets will not receive any keyboard or mouse button events. Other events (including `FL_ENTER`, `FL_MOVE`, `FL_LEAVE`, `FL_SHORTCUT`, and others) will still be sent. A widget is only active if [active\(\)](#) is true on it *and all of its parents*. Changing this value will send `FL_DEACTIVATE` to the widget if [active_r\(\)](#) is true. Currently you cannot deactivate `Fl_Window` widgets.

See also

[activate\(\)](#), [active\(\)](#), [active_r\(\)](#)

11.161.4.42 default_callback()

```
void Fl_Widget::default_callback (
    Fl_Widget * widget,
    void * data) [static]
```

The default callback for all widgets that don't set a callback.

This callback function puts a pointer to the widget on the queue returned by [Fl::readqueue\(\)](#). This is the default for all widgets if you don't set a callback.

You can avoid the overhead of this default handling if you set the callback to `NULL` explicitly.

Relying on the default callback and reading the callback queue with [Fl::readqueue\(\)](#) is not recommended. If you need a callback, you should set one with [Fl_Widget::callback\(Fl_Callback *cb, void *data\)](#) or one of its variants.

Parameters

in	<i>widget</i>	the Fl_Widget given to the callback
in	<i>data</i>	user data associated with that callback

See also

[callback\(\)](#), [Fl::readqueue\(\)](#)
[do_callback\(Fl_Widget *widget, void *data\)](#)

11.161.4.43 deimage() [1/4]

```
Fl_Image * Fl_Widget::deimage () [inline]
```

Gets the image that is used as part of the widget label when in the inactive state.

Returns

the current image for the deactivated widget

11.161.4.44 deimage() [2/4]

```
const Fl_Image * Fl_Widget::deimage () const [inline]
```

Gets the image that is used as part of the widget label when in the inactive state.

Returns

the current image for the deactivated widget

11.161.4.45 deimage() [3/4]

```
void Fl_Widget::deimage (
    Fl_Image & img)
```

Sets the image to use as part of the widget label when in the inactive state.

Parameters

in	<i>img</i>	the new image for the deactivated widget

See also

void [deimage\(Fl_Image* img\)](#)

11.161.4.46 deimage() [4/4]

```
void Fl_Widget::deimage (
    Fl_Image * img)
```

Sets the image to use as part of the widget label when in the inactive state.

Parameters

in	<i>img</i>	the new image for the deactivated widget

Note

The caller is responsible for making sure *img* is not deleted while it's used by the widget, and, if appropriate, for deleting it after the widget's deletion.

See also

void [bind_deimage\(Fl_Image* img\)](#)

11.161.4.47 deimage_bound()

```
int Fl_Widget::deimage_bound () const [inline]
```

Returns whether the inactive image is managed by the widget.

Return values

0	if the image is not bound to the widget
1	if the image will be deleted when the widget is deleted

See also

[image_bound\(\)](#), [bind_deimage\(\)](#)

11.161.4.48 do_callback() [1/3]

```
void Fl_Widget::do_callback (
    Fl\_Callback\_Reason reason = FL\_REASON\_UNKNOWN) [inline]
```

Calls the widget callback function with default arguments.

This is the same as calling

```
do\_callback(this, user\_data(), reason);
```

Parameters

in	<i>reason</i>	give a reason to why this callback was called, defaults to FL_REASON_UNKNOWN

See also

[callback\(\)](#)

[do_callback](#)([Fl_Widget](#) *widget, void *data, [Fl_Callback_Reason](#) reason), [Fl_Callback_Reason](#)

11.161.4.49 do_callback() [2/3]

```
void Fl_Widget::do_callback (
    Fl\_Widget * widget,
    long arg,
    Fl\_Callback\_Reason reason = FL\_REASON\_UNKNOWN) [inline]
```

Calls the widget callback function with arbitrary arguments.

Parameters

in	<i>widget</i>	call the callback with <i>widget</i> as the first argument
in	<i>arg</i>	call the callback with <i>arg</i> as the user data (second) argument
in	<i>reason</i>	give a reason to why this callback was called, defaults to FL_REASON_UNKNOWN

See also

[callback\(\)](#)

[do_callback](#)([Fl_Widget](#) *widget, void *data), [Fl_Callback_Reason](#)

11.161.4.50 do_callback() [3/3]

```
void Fl_Widget::do_callback (
    Fl\_Widget * widget,
    void * arg = 0,
    Fl\_Callback\_Reason reason = FL\_REASON\_UNKNOWN)
```

Calls the widget callback function with arbitrary arguments.

All overloads of [do_callback\(\)](#) call this method. It does nothing if the widget's [callback\(\)](#) is NULL. It clears the widget's *changed* flag **after** the callback was called unless the callback is the default callback. Hence it is not necessary to call [clear_changed\(\)](#) after calling [do_callback\(\)](#) in your own widget's [handle\(\)](#) method.

A *reason* must be set for widgets if different actions can trigger the same callback.

Note

It is legal to delete the widget in the callback (i.e. in user code), but you must not access the widget in the `handle()` method after calling `do_callback()` if the widget was deleted in the callback. We recommend to use `Fl_Widget_Tracker` to check whether the widget was deleted in the callback.

Parameters

in	<i>widget</i>	call the callback with <i>widget</i> as the first argument
in	<i>arg</i>	use <i>arg</i> as the user data (second) argument
in	<i>reason</i>	give a reason to why this callback was called, defaults to <code>FL_REASON_UNKNOWN</code>

See also

`default_callback()`
`callback()`
class `Fl_Widget_Tracker`
`Fl::callback_reason()`

11.161.4.51 draw()

```
virtual void Fl_Widget::draw () [pure virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call `redraw()` instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own `draw()` method*, e.g. for an embedded scrollbar, you can do it (because `draw()` is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                 // calls Fl_Scrollbar::draw()
```

Implemented in `Fl_Adjuster`, `Fl_Box`, `Fl_Browser`, `Fl_Button`, `Fl_Cairo_Window`, `Fl_Chart`, `Fl_Choice`, `Fl_Clock_Output`, `Fl_Counter`, `Fl_Dial`, `Fl_File_Input`, `Fl_Flex`, `Fl_FormsBitmap`, `Fl_FormsPixmap`, `Fl_FormsText`, `Fl_Free`, `Fl_Gl_Window`, `Fl_Glut_Window`, `Fl_Grid`, `Fl_Group`, `Fl_Help_View`, `Fl_Input`, `Fl_Input_Choice`, `Fl_Light_Button`, `Fl_Menu_Bar`, `Fl_Menu_Button`, `Fl_Pack`, `Fl_Positioner`, `Fl_Progress`, `Fl_Return_Button`, `Fl_Roller`, `Fl_Scroll`, `Fl_Scrollbar`, `Fl_Shortcut_Button`, `Fl_Slider`, `Fl_Spinner`, `Fl_Sys_Menu_Bar`, `Fl_Table`, `Fl_Tabs`, `Fl_Terminal`, `Fl_Text_Display`, `Fl_Timer`, `Fl_Tree`, `Fl_Value_Input`, `Fl_Value_Output`, `Fl_Value_Slider`, `Fl_Window`, and `Fl_Wizard`.

11.161.4.52 draw_focus() [1/3]

```
void Fl_Widget::draw_focus () const [inline], [protected]
```

Draws a focus rectangle around the widget.

This method uses the widget's boxtype and coordinates and its background color `color()`.

See also

`Fl_Widget::draw_focus(Fl_Boxtype, int, int, int, int, Fl_Color) const`

11.161.4.53 draw_focus() [2/3]

```
void Fl_Widget::draw_focus (
    Fl_Boxtype t,
    int X,
    int Y,
    int W,
    int H) const [inline], [protected]
```

Draws a focus rectangle around the widget.

This method uses the given boxtype and coordinates and the widget's background color `color()`.

See also

[Fl_Widget::draw_focus\(Fl_Boxtype, int, int, int, int, Fl_Color\) const](#)

11.161.4.54 draw_focus() [3/3]

```
void Fl_Widget::draw_focus (
    Fl_Boxtype bt,
    int X,
    int Y,
    int W,
    int H,
    Fl_Color bg) const [protected]
```

Draws a focus box for the widget at the given position and size.

This method does nothing if

- the global option [Fl::visible_focus\(\)](#) or
- the per-widget option [visible_focus\(\)](#) is false (off).

This means that [Fl_Widget::draw_focus\(\)](#) or one of the more specialized methods can be called without checking these visible focus options.

Note

This method must only be called if the widget has the focus. This is not tested internally.

The boxtype `bt` is used to calculate the inset so the focus box is drawn inside the box borders.

The default focus box drawing color is black. The background color `bg` is used to determine a better visible color if necessary by using [fl_contrast\(\)](#) with the given background color.

Parameters

in	<i>bt</i>	Boxtype that needs to be considered (frame width)
in	<i>X,Y,W,H</i>	Bounding box
in	<i>bg</i>	Background color

See also

[Fl_Widget::draw_focus\(\)](#)

[Fl_Widget::draw_focus\(Fl_Boxtype, int, int, int, int\) const](#)

11.161.4.55 draw_label() [1/3]

```
void Fl_Widget::draw_label () const [protected]
```

Draws the widget's label at the defined label position.

This is the normal call for a widget's [draw\(\)](#) method.

11.161.4.56 draw_label() [2/3]

```
void Fl_Widget::draw_label (
    int X,
    int Y,
    int W,
    int H) const [protected]
```

Draws the label in an arbitrary bounding box.

[draw\(\)](#) can use this instead of [draw_label\(void\)](#) to change the bounding box

11.161.4.57 draw_label() [3/3]

```
void Fl_Widget::draw_label (
    int X,
    int Y,
    int W,
    int H,
    Fl_Align a) const
```

Draws the label in an arbitrary bounding box with an arbitrary alignment. Anybody can call this to force the label to draw anywhere.

11.161.4.58 h() [1/2]

```
int Fl_Widget::h () const [inline]
```

Gets the widget height.

Returns

the height of the widget in pixels.

11.161.4.59 h() [2/2]

```
void Fl_Widget::h (
    int v) [inline], [protected]
```

Internal use only.

Use [position\(int,int\)](#), [size\(int,int\)](#) or [resize\(int,int,int,int\)](#) instead.

11.161.4.60 handle()

```
int Fl_Widget::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented in [Fl_Adjuster](#), [Fl_Box](#), [Fl_Browser](#), [Fl_Button](#), [Fl_Check_Browser](#), [Fl_Choice](#), [Fl_Clock](#), [Fl_Color_Chooser](#), [Fl_Counter](#), [Fl_Dial](#), [Fl_File_Input](#), [Fl_Free](#), [Fl_Gl_Window](#), [Fl_Glut_Window](#), [Fl_Group](#), [Fl_Help_View](#), [Fl_Input](#), [Fl_Light_Button](#), [Fl_Menu_Bar](#), [Fl_Menu_Button](#), [Fl_Positioner](#), [Fl_Repeat_Button](#), [Fl_Return_Button](#), [Fl_Roller](#), [Fl_Scheme_Choice](#), [Fl_Scroll](#), [Fl_Scrollbar](#), [Fl_Secret_Input](#), [Fl_Shortcut_Button](#), [Fl_Slider](#), [Fl_Spinner::Fl_Spinner_Input](#), [Fl_Spinner](#), [Fl_Table](#), [Fl_Table_Row](#), [Fl_Tabs](#), [Fl_Terminal](#), [Fl_Text_Display](#), [Fl_Text_Editor](#), [Fl_Tile](#), [Fl_Timer](#), [Fl_Tree](#), [Fl_Value_Input](#), [Fl_Value_Output](#), [Fl_Value_Slider](#), and [Fl_Window](#).

11.161.4.61 `hide()`

```
void Fl_Widget::hide () [virtual]
```

Makes a widget invisible.

See also

[show\(\)](#), [visible\(\)](#), [visible_r\(\)](#)

Reimplemented in [Fl_Browser](#), [Fl_Double_Window](#), [Fl_Gl_Window](#), [Fl_Overlay_Window](#), and [Fl_Window](#).

11.161.4.62 `horizontal_label_margin()` [1/2]

```
int Fl_Widget::horizontal_label_margin () [inline]
```

Get the spacing between the label and the horizontal edge of the widget.

Returns

px gap in pixels

11.161.4.63 `horizontal_label_margin()` [2/2]

```
void Fl_Widget::horizontal_label_margin (
    int px) [inline]
```

Set the spacing between the label and the horizontal edge of the widget.

Parameters

in	<i>px</i>	gap in pixels

11.161.4.64 `image()` [1/4]

```
Fl_Image * Fl_Widget::image () [inline]
```

Gets the image that is used as part of the widget label when in the active state.

Returns

the current image

11.161.4.65 `image()` [2/4]

```
const Fl_Image * Fl_Widget::image () const [inline]
```

Gets the image that is used as part of the widget label when in the active state.

Returns

the current image

11.161.4.66 `image()` [3/4]

```
void Fl_Widget::image (
    Fl_Image & img)
```

Sets the image to use as part of the widget label when in the active state.

Parameters

in	<i>img</i>	the new image for the label

See also

[void image\(Fl_Image* img\)](#)

11.161.4.67 image() [4/4]

```
void Fl_Widget::image (
    Fl_Image * img)
```

Sets the image to use as part of the widget label when in the active state.

The caller is responsible for making sure *img* is not deleted while it's used by the widget, and, if appropriate, for deleting it after the widget's deletion.

Calling [image\(\)](#) with a new image will delete the old image if it was bound, and set the new image without binding it.

If old and new are the same, the image will not be deleted, but it will be unbound.

Calling [image\(\)](#) with NULL will delete the old image if it was bound and not set a new image.

Parameters

in	<i>img</i>	the new image for the label

See also

[bind_image\(Fl_Image* img\)](#)

11.161.4.68 image_bound()

```
int Fl_Widget::image_bound () const [inline]
```

Returns whether the image is managed by the widget.

Return values

0	if the image is not bound to the widget
1	if the image will be deleted when the widget is deleted

See also

[deimage_bound\(\)](#), [bind_image\(\)](#)

11.161.4.69 inside()

```
int Fl_Widget::inside (
    const Fl_Widget * wgt) const [inline]
```

Checks if this widget is a child of *wgt*.

Returns 1 if this widget is a child of *wgt*, or is equal to *wgt*. Returns 0 if *wgt* is NULL.

Parameters

in	<i>wgt</i>	the possible parent widget.

See also

[contains\(\)](#)

11.161.4.70 is_label_copied()

```
int Fl_Widget::is_label_copied () const [inline]
```

Returns whether the current label was assigned with [copy_label\(\)](#).

This can be useful for temporarily overwriting the widget's label and restoring it later.

Return values

0	current label was assigned with label() .
1	current label was assigned with copy_label() .

11.161.4.71 label() [1/3]

```
const char * Fl_Widget::label () const [inline]
```

Gets the current label text.

Returns

a pointer to the current label text

See also

[label\(const char *\)](#), [copy_label\(const char *\)](#)

11.161.4.72 label() [2/3]

```
void Fl_Widget::label (
    const char * text)
```

Sets the current label pointer.

The label is shown somewhere on or next to the widget. See [Labels and Label Types](#) for details about what can be put in a label. The passed pointer is stored unchanged in the widget (the string is *not* copied), so if you need to set the label to a formatted value, make sure the buffer is static, global, or allocated. The [copy_label\(\)](#) method can be used to make a copy of the label string automatically.

Parameters

in	<i>text</i>	pointer to new label text

See also

[copy_label\(\)](#)

11.161.4.73 label() [3/3]

```
void Fl_Widget::label (
    Fl_Labeltype a,
    const char * b) [inline]
```

Shortcut to set the label text and type in one call.

See also

[label\(const char *\)](#), [labeltype\(Fl_Labeltype\)](#)

11.161.4.74 label_image_spacing() [1/2]

```
int Fl_Widget::label_image_spacing () [inline]
```

Return the gap size between the label and the image.

Returns

spacing in pixels

11.161.4.75 label_image_spacing() [2/2]

```
void Fl_Widget::label_image_spacing (
    int gap) [inline]
```

Set the gap between the label and the image in pixels.
This value is limited to 0..255.

Parameters

in	<i>gap</i>	spacing in pixels

11.161.4.76 label_shortcut()

```
unsigned int Fl_Widget::label_shortcut (
    const char * t) [static]
```

Returns the Unicode value of the '&x' shortcut in a given text.

The given text *t* (usually a widget's label or a menu text) is searched for a '&x' shortcut label, and if found, the Unicode value (code point) of the '&x' shortcut is returned.

Parameters

<i>t</i>	text or label to search for '&x' shortcut.

Returns

Unicode (UCS-4) value of shortcut in *t* or 0.

Note

Internal use only.

11.161.4.77 labelcolor() [1/2]

```
Fl_Color Fl_Widget::labelcolor () const [inline]
```

Gets the label color.

The default color is FL_FOREGROUND_COLOR.

Returns

the current label color

11.161.4.78 labelcolor() [2/2]

```
void Fl_Widget::labelcolor (
    Fl_Color c) [inline]
```

Sets the label color.

The default color is FL_FOREGROUND_COLOR.

Parameters

in	<i>c</i>	the new label color

11.161.4.79 labelfont() [1/2]

```
Fl_Font Fl_Widget::labelfont () const [inline]
```

Gets the font to use.

Fonts are identified by indexes into a table. The default value uses a Helvetica typeface (Arial for Microsoft® Windows®). The function [Fl::set_font\(\)](#) can define new typefaces.

Returns

current font used by the label

See also

[Fl_Font](#)

11.161.4.80 labelfont() [2/2]

```
void Fl_Widget::labelfont (
    Fl_Font f) [inline]
```

Sets the font to use.

Fonts are identified by indexes into a table. The default value uses a Helvetica typeface (Arial for Microsoft® Windows®). The function [Fl::set_font\(\)](#) can define new typefaces.

Parameters

<i>in</i>	<i>f</i>	the new font for the label

See also

[Fl_Font](#)

11.161.4.81 labelsize() [1/2]

```
Fl_Fontsize Fl_Widget::labelsize () const [inline]
```

Gets the font size in pixels.

The default size is 14 pixels.

Returns

the current font size

11.161.4.82 labelsize() [2/2]

```
void Fl_Widget::labelsize (
    Fl_Fontsize pix) [inline]
```

Sets the font size in pixels.

Parameters

<i>in</i>	<i>pix</i>	the new font size

See also

[Fl_Fontsize labelsize\(\)](#)

11.161.4.83 labeltype() [1/2]

```
Fl_Labeltype Fl_Widget::labeltype () const [inline]
```

Gets the label type.

Returns

the current label type.

See also

[Fl_Labeltype](#)

11.161.4.84 labeltype() [2/2]

```
void Fl_Widget::labeltype (
    Fl_Labeltype a) [inline]
```

Sets the label type.

The label type identifies the function that draws the label of the widget. This is generally used for special effects such as embossing or for using the [label\(\)](#) pointer as another form of data such as an icon. The value `FL_NORMAL_LABEL` prints the label as plain text.

Parameters

<code>in</code>	<code>a</code>	new label type

See also

[Fl_Labeltype](#)

11.161.4.85 measure_label()

```
void Fl_Widget::measure_label (
    int & ww,
    int & hh) const [inline]
```

Sets width `ww` and height `hh` accordingly with the label size.

Labels with images will return `w()` and `h()` of the image.

This calls [fl_measure\(\)](#) internally. For more information about the arguments `ww` and `hh` and word wrapping

See also

[fl_measure\(const char*, int&, int&, int\)](#)

11.161.4.86 needs_keyboard() [1/2]

```
bool Fl_Widget::needs_keyboard () const [inline]
```

Returns whether this widget needs a keyboard.

Returns

true or false

See also

[needs_keyboard\(bool\)](#)

11.161.4.87 needs_keyboard() [2/2]

```
void Fl_Widget::needs_keyboard (
    bool needs) [inline]
```

Sets whether this widget needs a keyboard.

Set this on touch screen devices if a widget needs a keyboard when it gets the focus.

Note

This flag can be set but is not yet **used** in FLTK 1.4.0. It is intended to be used in the future on real touch devices.

Parameters

in	<i>needs</i>	Set this to true or false

11.161.4.88 output()

```
unsigned int Fl_Widget::output () const [inline]
```

Returns if a widget is used for output only.

[output\(\)](#) means the same as [lactive\(\)](#) except it does not change how the widget is drawn. The widget will not receive any events. This is useful for making scrollbars or buttons that work as displays rather than input devices.

Return values

0	if the widget is used for input and output

See also

[set_output\(\)](#), [clear_output\(\)](#)

11.161.4.89 parent() [1/2]

```
Fl_Group * Fl_Widget::parent () const [inline]
```

Returns a pointer to the parent widget.

Usually this is a [Fl_Group](#) or [Fl_Window](#).

Return values

<i>NULL</i>	if the widget has no parent

See also

[Fl_Group::add\(Fl_Widget*\)](#)

11.161.4.90 parent() [2/2]

```
void Fl_Widget::parent (
    Fl_Group * p) [inline]
```

Internal use only - "for hacks only".

It is **STRONGLY recommended** not to use this method, because it short-circuits [Fl_Group](#)'s normal widget adding and removing methods, if the widget is already a child widget of another [Fl_Group](#).

Use [Fl_Group::add\(Fl_Widget*\)](#) and/or [Fl_Group::remove\(Fl_Widget*\)](#) instead.

11.161.4.91 position()

```
void Fl_Widget::position (
    int X,
    int Y) [inline]
```

Repositions the window or widget.

position(X, Y) is a shortcut for `resize(X, Y, w(), h())`.

Parameters

in	<i>X,Y</i>	new position relative to the parent window

See also

[resize\(int,int,int,int\)](#), [size\(int,int\)](#)

11.161.4.92 redraw()

```
void Fl_Widget::redraw ()
```

Schedules the drawing of the widget.

Marks the widget as needing its [draw\(\)](#) routine called.

11.161.4.93 redraw_label()

```
void Fl_Widget::redraw_label ()
```

Schedules the drawing of the label.

Marks the widget or the parent as needing a redraw for the label area of a widget.

11.161.4.94 resize()

```
void Fl_Widget::resize (
    int x,
    int y,
    int w,
    int h) [virtual]
```

Changes the size or position of the widget.

This is a virtual function so that the widget may implement its own handling of resizing. The default version does *not* call the [redraw\(\)](#) method, but instead relies on the parent widget to do so because the parent may know a faster way to update the display, such as scrolling from the old position.

Some window managers under X11 call [resize\(\)](#) a lot more often than needed. Please verify that the position or size of a widget did actually change before doing any extensive calculations.

[position\(X, Y\)](#) is a shortcut for [resize\(X, Y, w\(\), h\(\)\)](#), and [size\(W, H\)](#) is a shortcut for [resize\(x\(\), y\(\), W, H\)](#).

Parameters

in	<i>x,y</i>	new position relative to the parent window
in	<i>w,h</i>	new size

See also

[position\(int,int\)](#), [size\(int,int\)](#)

Reimplemented in [Fl_Browser_](#), [Fl_Double_Window](#), [Fl_Flex](#), [Fl_Gl_Window](#), [Fl_Grid](#), [Fl_Group](#), [Fl_Help_View](#), [Fl_Input_](#), [Fl_Input_Choice](#), [Fl_Overlay_Window](#), [Fl_Pack](#), [Fl_Scroll](#), [Fl_Spinner](#), [Fl_Table](#), [Fl_Tabs](#), [Fl_Terminal](#), [Fl_Text_Display](#), [Fl_Tile](#), [Fl_Tree](#), [Fl_Value_Input](#), and [Fl_Window](#).

11.161.4.95 selection_color() [1/2]

```
Fl_Color Fl_Widget::selection_color () const [inline]
```

Gets the selection color.

Returns

the current selection color

See also

[selection_color\(Fl_Color\)](#), [color\(Fl_Color, Fl_Color\)](#)

11.161.4.96 selection_color() [2/2]

```
void Fl_Widget::selection_color (
    Fl_Color a) [inline]
```

Sets the selection color.

The selection color is defined for Forms compatibility and is usually used to color the widget when it is selected, although some widgets use this color for other purposes. You can set both colors at once with [color\(Fl_Color bg, Fl_Color sel\)](#).

Parameters

<code>in</code>	<code>a</code>	the new selection color

See also

[selection_color\(\)](#), [color\(Fl_Color, Fl_Color\)](#)

11.161.4.97 set_active()

```
void Fl_Widget::set_active () [inline]
```

Marks the widget as active without sending events or changing focus.

This is mainly for specialized use, for normal cases you want [activate\(\)](#).

See also

[activate\(\)](#)

11.161.4.98 set_changed()

```
void Fl_Widget::set_changed () [inline]
```

Marks the value of the widget as changed.

See also

[changed\(\)](#), [clear_changed\(\)](#)

11.161.4.99 set_output()

```
void Fl_Widget::set_output () [inline]
```

Sets a widget to output only.

See also

[output\(\)](#), [clear_output\(\)](#)

11.161.4.100 set_visible()

```
void Fl_Widget::set_visible () [inline]
```

Makes the widget visible.

You must still redraw the parent widget to see a change in the window. Normally you want to use the [show\(\)](#) method instead.

11.161.4.101 set_visible_focus()

```
void Fl_Widget::set_visible_focus () [inline]
```

Enables keyboard focus navigation with this widget.

Note, however, that this will not necessarily mean that the widget will accept focus, but for widgets that can accept focus, this method enables it if it has been disabled.

See also

[visible_focus\(\)](#), [clear_visible_focus\(\)](#), [visible_focus\(int\)](#)

11.161.4.102 shortcut_label() [1/2]

```
int Fl_Widget::shortcut_label () const [inline]
```

Returns whether the widget's label uses '&' to indicate shortcuts.

See also

[void shortcut_label\(int value\)](#)

11.161.4.103 shortcut_label() [2/2]

```
void Fl_Widget::shortcut_label (
    int value) [inline]
```

Sets whether the widget's label uses '&' to indicate shortcuts.

By default, all objects of classes [Fl_Menu_](#) (and derivatives), [Fl_Button](#) (and derivatives), [Fl_Text_Display](#), [Fl_Value_Input](#), and [Fl_Input_](#) (and derivatives) use character '&' in their label, unless '&' is repeated, to indicate shortcuts: '&' does not appear in the drawn label, the next character after '&' in the label is drawn underlined, and typing this character triggers the corresponding menu window, button, or other widget. If the label contains 2 consecutive '&', only one is drawn and the next character is not underlined and not used as a shortcut. If `value` is set to 0, all these labels don't process character '&' as indicating a shortcut: '&' is drawn in the label, the next character is not underlined and does not define a shortcut.

11.161.4.104 show()

```
void Fl_Widget::show () [virtual]
```

Makes a widget visible.

An invisible widget never gets redrawn and does not get keyboard or mouse events, but can receive a few other events like `FL_SHOW`.

The [visible\(\)](#) method returns true if the widget is set to be visible. The [visible_r\(\)](#) method returns true if the widget and all of its parents are visible. A widget is only visible if [visible\(\)](#) is true on it *and all of its parents*.

Changing it will send `FL_SHOW` or `FL_HIDE` events to the widget. *Do not change it if the parent is not visible, as this will send false `FL_SHOW` or `FL_HIDE` events to the widget.* [redraw\(\)](#) is called if necessary on this or the parent.

See also

[hide\(\)](#), [visible\(\)](#), [visible_r\(\)](#)

Reimplemented in [Fl_Browser](#), [Fl_Double_Window](#), [Fl_Gl_Window](#), [Fl_Overlay_Window](#), [Fl_Single_Window](#), [Fl_Tabs](#), and [Fl_Window](#).

11.161.4.105 size()

```
void Fl_Widget::size (
    int W,
    int H) [inline]
```

Changes the size of the widget.

`size(W, H)` is a shortcut for `resize(x(), y(), W, H)`.

Parameters

in	<i>W,H</i>	new size

See also

[position\(int,int\)](#), [resize\(int,int,int,int\)](#)

11.161.4.106 take_focus()

```
int Fl_Widget::take_focus ()
```

Gives the widget the keyboard focus.

Tries to make this widget be the [Fl::focus\(\)](#) widget, by first sending it an FL_FOCUS event, and if it returns non-zero, setting [Fl::focus\(\)](#) to this widget. You should use this method to assign the focus to a widget.

Returns

true if the widget accepted the focus.

11.161.4.107 takeevents()

```
unsigned int Fl_Widget::takeevents () const [inline]
```

Returns if the widget is able to take events.

This is the same as ([active\(\)](#) && ![output\(\)](#) && [visible\(\)](#)) but is faster.

Return values

0	if the widget takes no events

11.161.4.108 test_shortcut() [1/2]

```
int Fl_Widget::test_shortcut ()
```

Returns true if the widget's label contains the entered '&x' shortcut.

This method must only be called in [handle\(\)](#) methods or callbacks after a keypress event (usually FL_KEYDOWN or FL_SHORTCUT). The widget's label is searched for a '&x' shortcut, and if found, this is compared with the entered key value.

[Fl::event_text\(\)](#) is used to get the entered key value.

Returns

true, if the entered text matches the widget's '&x' shortcut, false (0) otherwise.

Note

Useful when a widget's [handle\(int\)](#) method needs dedicated processing of FL_SHORTCUT.

11.161.4.109 test_shortcut() [2/2]

```
int Fl_Widget::test_shortcut (
    const char * t,
    const bool require_alt = false) [static]
```

Returns true if the given text *t* contains the entered '&x' shortcut.

This method must only be called in [handle\(\)](#) methods or callbacks after a keypress event (usually FL_KEYDOWN or FL_SHORTCUT). The given text *t* (usually a widget's label or menu text) is searched for a '&x' shortcut, and if found, this is compared with the entered key value.

[Fl::event_text\(\)](#) is used to get the entered key value. [Fl::event_state\(\)](#) is used to get the Alt modifier, if *require_alt* is true.

Parameters

<i>t</i>	text or label to search for '&x' shortcut.
<i>require_alt</i>	if true: match only if Alt key is pressed.

Returns

true, if the entered text matches the '&x' shortcut in *t* false (0) otherwise.

Note

Useful when a widget's [handle\(int\)](#) method needs dedicated processing of FL_SHORTCUT.

11.161.4.110 tooltip() [1/2]

```
const char * Fl_Widget::tooltip () const [inline]
```

Gets the current tooltip text.

Returns

a pointer to the tooltip text or NULL

See also

[tooltip\(const char*\)](#), [copy_tooltip\(const char*\)](#)

11.161.4.111 tooltip() [2/2]

```
void Fl_Widget::tooltip (
    const char * text)
```

Sets the current tooltip text.

Sets a string of text to display in a popup tooltip window when the user hovers the mouse over the widget. The string is *not* copied, so make sure any formatted string is stored in a static, global, or allocated buffer. If you want a copy made and managed for you, use the [copy_tooltip\(\)](#) method, which will manage the tooltip string automatically. If no tooltip is set, the tooltip of the parent is inherited. Setting a tooltip for a group and setting no tooltip for a child will show the group's tooltip instead. To avoid this behavior, you can set the child's tooltip to an empty string ("").

Parameters

<i>in</i>	<i>text</i>	New tooltip text (no copy is made)

See also

[copy_tooltip\(const char*\)](#), [tooltip\(\)](#)

11.161.4.112 top_window()

```
Fl_Window * Fl_Widget::top_window () const
```

Returns a pointer to the top-level window for the widget.

In other words, the 'window manager window' that contains this widget. This method differs from [window\(\)](#) in that it won't return sub-windows (if there are any).

Returns

the top-level window, or NULL if no top-level window is associated with this widget.

See also

[window\(\)](#)

11.161.4.113 top_window_offset()

```
Fl_Window * Fl_Widget::top_window_offset (
    int & xoff,
    int & yoff) const
```

Finds the x/y offset of the current widget relative to the top-level window.

Parameters

out	<i>xoff,yoff</i>	Returns the x/y offset

Returns

the top-level window (or NULL for a widget that's not in any window)

11.161.4.114 type() [1/2]

```
uchar Fl_Widget::type () const [inline]
```

Gets the widget type.

Returns the widget type value, which gives some information about the derived widget class to which the object belongs. Noticeably, the condition `type() >= FL_WINDOW` indicates a widget is an [Fl_Window](#) or derived object.

11.161.4.115 type() [2/2]

```
void Fl_Widget::type (
    uchar t) [inline]
```

Sets the widget type.

See also

[type\(\)](#)

11.161.4.116 user_data()

```
void * Fl_Widget::user_data () const [inline]
```

Gets the user data for this widget.

Gets the current user data (void *) argument that is passed to the callback function.

Returns

user data as a pointer

11.161.4.117 vertical_label_margin() [1/2]

```
int Fl_Widget::vertical_label_margin () [inline]
```

Get the spacing between the label and the vertical edge of the widget.

Returns

px gap in pixels

11.161.4.118 vertical_label_margin() [2/2]

```
void Fl_Widget::vertical_label_margin (
    int px) [inline]
```

Set the spacing between the label and the vertical edge of the widget.

Parameters

in	<i>px</i>	gap in pixels

11.161.4.119 visible()

```
unsigned int Fl_Widget::visible () const [inline]
```

Returns whether a widget is visible.

Return values

0	if the widget is not drawn and hence invisible.

See also

[show\(\)](#), [hide\(\)](#), [visible_r\(\)](#)

11.161.4.120 visible_focus() [1/2]

```
unsigned int Fl_Widget::visible_focus () const [inline]
```

Checks whether this widget has a visible focus.

Return values

0	if this widget has no visible focus.

See also

[visible_focus\(int\)](#), [set_visible_focus\(\)](#), [clear_visible_focus\(\)](#)

11.161.4.121 visible_focus() [2/2]

```
void Fl_Widget::visible_focus (
    int v) [inline]
```

Modifies keyboard focus navigation.

Parameters

in	v	set or clear visible focus

See also

[set_visible_focus\(\)](#), [clear_visible_focus\(\)](#), [visible_focus\(\)](#)

11.161.4.122 visible_r()

```
int Fl_Widget::visible_r () const
```

Returns whether a widget and all its parents are visible.

Return values

0	if the widget or any of its parents are invisible.

See also

[show\(\)](#), [hide\(\)](#), [visible\(\)](#)

11.161.4.123 w() [1/2]

```
int Fl_Widget::w () const [inline]
```

Gets the widget width.

Returns

the width of the widget in pixels.

11.161.4.124 w() [2/2]

```
void Fl_Widget::w (
    int v) [inline], [protected]
```

Internal use only.

Use [position\(int,int\)](#), [size\(int,int\)](#) or [resize\(int,int,int,int\)](#) instead.

11.161.4.125 when() [1/2]

```
Fl_When Fl_Widget::when () const [inline]
```

Returns the conditions under which the callback is called.

You can set the flags with [when\(uchar\)](#), the default value is FL_WHEN_RELEASE.

Returns

set of flags

See also

[when\(uchar\)](#), [Fl_When](#), [do_callback\(\)](#), [Fl::callback_reason\(\)](#)

11.161.4.126 when() [2/2]

```
void Fl_Widget::when (
    uchar i) [inline]
```

Sets the flags used to decide when a callback is called.

This controls when callbacks are done. The following values are useful, the default value is FL_WHEN_RELEASE:

- 0: The callback is not done, but [changed\(\)](#) is turned on.
- FL_WHEN_CHANGED: The callback is done each time the text is changed by the user.
- FL_WHEN_RELEASE: The callback will be done when this widget loses the focus, including when the window is unmapped. This is a useful value for text fields in a panel where doing the callback on every change is wasteful. However the callback will also happen if the mouse is moved out of the window, which means it should not do anything visible (like pop up an error message). You might do better setting this to zero, and scanning all the items for [changed\(\)](#) when the OK button on a panel is pressed.
- FL_WHEN_ENTER_KEY: If the user types the Enter key, the entire text is selected, and the callback is done if the text has changed. Normally the Enter key will navigate to the next field (or insert a newline for a [Fl_Multiline_Input](#)) - this changes the behavior.
- FL_WHEN_ENTER_KEY|FL_WHEN_NOT_CHANGED: The Enter key will do the callback even if the text has not changed. Useful for command fields.
- [FL_WHEN_CLOSED](#) : If the user requests that the widget is closed, the callback is called with [FL_REASON_CLOSED](#). The [Fl_Tabs](#) widget checks this flag on its children to determine whether to display a close button on the tab of that widget.

[Fl_Widget::when\(\)](#) is a set of bitflags used by subclasses of [Fl_Widget](#) to decide when to do the callback.

If the value is zero then the callback is never done. Other values are described in the individual widgets. This field is in the base class so that you can scan a panel and [do_callback\(\)](#) on all the ones that don't do their own callbacks in response to an "OK" button.

Parameters

<i>in</i>	<i>i</i>	set of flags

See also

[Fl_When](#), [do_callback\(\)](#), [Fl::callback_reason\(\)](#)

11.161.4.127 window()

```
Fl_Window * Fl_Widget::window () const
```

Returns a pointer to the nearest parent window up the widget hierarchy.

This will return sub-windows if there are any, or the parent window if there's no sub-windows. If this widget IS the top-level window, NULL is returned.

Return values

<i>NULL</i>	if no window is associated with this widget.

Note

for an [Fl_Window](#) widget, this returns its *parent* window (if any), not *this* window.

See also

[top_window\(\)](#)

11.161.4.128 x() [1/2]

```
int Fl_Widget::x () const [inline]
```

Gets the widget position in its window.

Returns

the x position relative to the window

11.161.4.129 x() [2/2]

```
void Fl_Widget::x (
    int v) [inline], [protected]
```

Internal use only.

Use [position\(int,int\)](#), [size\(int,int\)](#) or [resize\(int,int,int,int\)](#) instead.

11.161.4.130 y() [1/2]

```
int Fl_Widget::y () const [inline]
```

Gets the widget position in its window.

Returns

the y position relative to the window

11.161.4.131 y() [2/2]

```
void Fl_Widget::y (
    int v) [inline], [protected]
```

Internal use only.

Use [position\(int,int\)](#), [size\(int,int\)](#) or [resize\(int,int,int,int\)](#) instead.

The documentation for this class was generated from the following files:

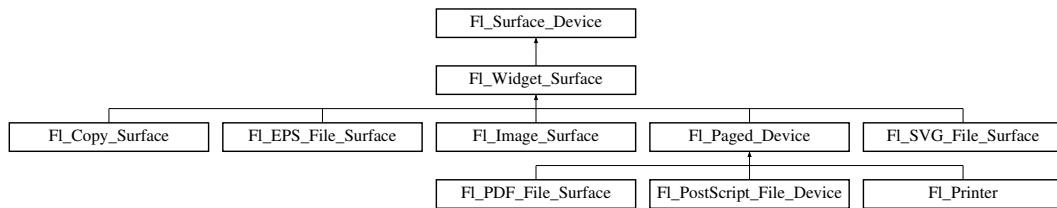
- [Fl_Widget.H](#)
- [Fl.cxx](#)
- [fl_boxtype.cxx](#)
- [fl_labeltype.cxx](#)
- [fl_shortcut.cxx](#)
- [Fl_Tooltip.cxx](#)
- [Fl_Widget.cxx](#)
- [Fl_Window.cxx](#)

11.162 Fl_Widget_Surface Class Reference

A surface on which any FLTK widget can be drawn.

```
#include <Fl_Widget_Surface.H>
```

Inheritance diagram for Fl_Widget_Surface:



Public Member Functions

- void [draw](#) ([Fl_Widget](#) *widget, int delta_x=0, int delta_y=0)
Draws the widget on the drawing surface.
- void [draw_decorated_window](#) ([Fl_Window](#) *win, int x_offset=0, int y_offset=0)
Draws a window with its title bar and frame if any.
- virtual void [origin](#) (int *x, int *y)
Computes the coordinates of the current origin of graphics functions.
- virtual void [origin](#) (int x, int y)
Sets the position of the origin of graphics in the drawable part of the drawing surface.
- void [print_window_part](#) ([Fl_Window](#) *win, int x, int y, int w, int h, int delta_x=0, int delta_y=0)
Draws a rectangular part of an on-screen window.
- virtual int [printable_rect](#) (int *w, int *h)
Computes the width and height of the drawable area of the drawing surface.
- virtual void [translate](#) (int x, int y)
Translates the current graphics origin accounting for the current rotation.
- virtual void [untranslate](#) ()
Undoes the effect of a previous [translate\(\)](#) call.

Public Member Functions inherited from [Fl_Surface_Device](#)

- [Fl_Graphics_Driver](#) * [driver](#) ()
Returns the graphics driver of this drawing surface.
- virtual bool [is_current](#) ()
Is this surface the current drawing surface?
- virtual void [set_current](#) (void)
Make this surface the current drawing surface.
- virtual ~[Fl_Surface_Device](#) ()
The destructor.

Protected Member Functions

- [Fl_Widget_Surface](#) ([Fl_Graphics_Driver](#) *d)
The constructor.

Protected Member Functions inherited from [Fl_Surface_Device](#)

- void **driver** (Fl_Graphics_Driver *graphics_driver)
Sets the graphics driver of this drawing surface.
- virtual void **end_current** ()
FLTK calls this each time a surface ceases to be the current drawing surface.
- **Fl_Surface_Device** (Fl_Graphics_Driver *graphics_driver)
Constructor that sets the graphics driver to use for the created surface.

Protected Attributes

- int **x_offset**
horizontal offset to the origin of graphics coordinates
- int **y_offset**
vertical offset to the origin of graphics coordinates

Additional Inherited Members

Static Public Member Functions inherited from [Fl_Surface_Device](#)

- static [Fl_Surface_Device](#) * **pop_current** ()
Removes the top element from the current drawing surface stack, and makes the new top element current.
- static void **push_current** ([Fl_Surface_Device](#) *new_current)
Pushes new_current on top of the stack of current drawing surfaces, and makes it current.
- static [Fl_Surface_Device](#) * **surface** ()
The current drawing surface.

11.162.1 Detailed Description

A surface on which any FLTK widget can be drawn.

11.162.2 Constructor & Destructor Documentation

11.162.2.1 Fl_Widget_Surface()

```
Fl_Widget_Surface::Fl_Widget_Surface (
    Fl_Graphics_Driver * d) [protected]
```

The constructor.

Parameters

<i>d</i>	can be nul.

11.162.3 Member Function Documentation

11.162.3.1 draw()

```
void Fl_Widget_Surface::draw (
    Fl_Widget * widget,
    int delta_x = 0,
    int delta_y = 0)
```

Draws the widget on the drawing surface.

The widget's position on the surface is determined by the last call to [origin\(\)](#) and by the optional delta_x and delta_y arguments. Its dimensions are in points unless there was a previous call to [scale\(\)](#).

Parameters

in	<i>widget</i>	Any FLTK widget (e.g., standard, custom, window).
in	<i>delta_x, delta_y</i>	Optional horizontal and vertical offsets for positioning the widget top left relatively to the current origin of graphics.

11.162.3.2 draw_decorated_window()

```
void Fl_Widget_Surface::draw_decorated_window (
    Fl_Window * win,
    int win_offset_x = 0,
    int win_offset_y = 0)
```

Draws a window with its title bar and frame if any.

win_offset_x and *win_offset_y* are optional coordinates of where to position the window top left. Equivalent to [draw\(\)](#) if *win* is a subwindow or has no border. Use [Fl_Window::decorated_w\(\)](#) and [Fl_Window::decorated_h\(\)](#) to get the size of the framed window.

11.162.3.3 origin() [1/2]

```
void Fl_Widget_Surface::origin (
    int * x,
    int * y) [virtual]
```

Computes the coordinates of the current origin of graphics functions.

Parameters

out	<i>x,y</i>	If non-null, *x and *y are set to the horizontal and vertical coordinates of the graphics origin.

Reimplemented in [Fl_Copy_Surface](#), [Fl_EPS_File_Surface](#), [Fl_Image_Surface](#), [Fl_PDF_File_Surface](#), [Fl_PostScript_File_Device](#), [Fl_Printer](#), and [Fl_SVG_File_Surface](#).

11.162.3.4 origin() [2/2]

```
void Fl_Widget_Surface::origin (
    int x,
    int y) [virtual]
```

Sets the position of the origin of graphics in the drawable part of the drawing surface.

Arguments should be expressed relatively to the result of a previous [printable_rect\(\)](#) call. That is, `printable_rect(&w, &h); origin(w/2, 0);` sets the graphics origin at the top center of the drawable area. Successive [origin\(\)](#) calls don't combine their effects. [Origin\(\)](#) calls are not affected by [rotate\(\)](#) calls (for classes derived from [Fl_Paged_Device](#)).

Parameters

in	<i>x,y</i>	Horizontal and vertical positions in the drawing surface of the desired origin of graphics.

Reimplemented in [Fl_Copy_Surface](#), [Fl_EPS_File_Surface](#), [Fl_Image_Surface](#), [Fl_PDF_File_Surface](#), [Fl_PostScript_File_Device](#), [Fl_Printer](#), and [Fl_SVG_File_Surface](#).

11.162.3.5 print_window_part()

```
void Fl_Widget_Surface::print_window_part (
    Fl_Window * win,
    int x,
    int y,
    int w,
    int h,
```

```
int delta_x = 0,
int delta_y = 0)
```

Draws a rectangular part of an on-screen window.

Parameters

<i>win</i>	The window from where to capture. Can be an Fl_Gl_Window . Sub-windows that intersect the rectangle are also captured.
<i>x</i>	The rectangle left
<i>y</i>	The rectangle top
<i>w</i>	The rectangle width
<i>h</i>	The rectangle height
<i>delta_x, delta_y</i>	Optional horizontal and vertical offsets from current graphics origin where to draw the top left of the captured rectangle.

11.162.3.6 printable_rect()

```
int Fl_Widget_Surface::printable_rect (
    int * w,
    int * h) [virtual]
```

Computes the width and height of the drawable area of the drawing surface.

Values are in the same unit as that used by FLTK drawing functions and are unchanged by calls to [origin\(\)](#). If the object is derived from class [Fl_Paged_Device](#), values account for the user-selected paper type and print orientation and are changed by [scale\(\)](#) calls.

Returns

0 if OK, non-zero if any error

Reimplemented in [Fl_Copy_Surface](#), [Fl_EPS_File_Surface](#), [Fl_Image_Surface](#), [Fl_PDF_File_Surface](#), [Fl_PostScript_File_Device](#), [Fl_Printer](#), and [Fl_SVG_File_Surface](#).

11.162.3.7 translate()

```
void Fl_Widget_Surface::translate (
    int x,
    int y) [virtual]
```

Translates the current graphics origin accounting for the current rotation.

Each [translate\(\)](#) call must be matched by an [untranslate\(\)](#) call. Successive [translate\(\)](#) calls add up their effects.

Reimplemented in [Fl_Copy_Surface](#), [Fl_EPS_File_Surface](#), [Fl_Image_Surface](#), [Fl_PDF_File_Surface](#), [Fl_PostScript_File_Device](#), [Fl_Printer](#), and [Fl_SVG_File_Surface](#).

11.162.3.8 untranslate()

```
void Fl_Widget_Surface::untranslate (
    void ) [virtual]
```

Undoes the effect of a previous [translate\(\)](#) call.

Reimplemented in [Fl_Copy_Surface](#), [Fl_EPS_File_Surface](#), [Fl_Image_Surface](#), [Fl_PDF_File_Surface](#), [Fl_PostScript_File_Device](#), [Fl_Printer](#), and [Fl_SVG_File_Surface](#).

The documentation for this class was generated from the following files:

- [Fl_Widget_Surface.H](#)
- [Fl_Widget_Surface.cxx](#)

11.163 Fl_Widget_Tracker Class Reference

This class should be used to control safe widget deletion.

```
#include <Fl.H>
```

Public Member Functions

- `int deleted ()`
Returns 1, if the watched widget has been deleted.
- `int exists ()`
Returns 1, if the watched widget exists (has not been deleted).
- `FL_Widget_Tracker (FL_Widget *wi)`
The constructor adds a widget to the watch list.
- `FL_Widget * widget ()`
Returns a pointer to the watched widget.
- `~FL_Widget_Tracker ()`
The destructor removes a widget from the watch list.

11.163.1 Detailed Description

This class should be used to control safe widget deletion.

You can use an `FL_Widget_Tracker` object to watch another widget, if you need to know whether this widget has been deleted during a callback.

This simplifies the use of the "safe widget deletion" methods `FL::watch_widget_pointer()` and `FL::release_widget_pointer()` and makes their use more reliable, because the destructor automatically releases the widget pointer from the widget watch list.

`FL_Widget_Tracker` is intended to be used as an automatic (local/stack) variable, such that its destructor is called when the object's scope is left. This ensures that no stale widget pointers are left in the widget watch list (see example below).

You can also create `FL_Widget_Tracker` objects with `new`, but then it is your responsibility to delete the object (and thus remove the widget pointer from the watch list) when it is no longer needed.

Example:

```
int MyClass::handle (int event) {

    if (...) {
        FL_Widget_Tracker wp(this);           // watch myself
        do_callback();                         // call the callback

        if (wp.deleted()) return 1;           // exit, if deleted

        // Now we are sure that the widget has not been deleted,
        // and it is safe to access the widget:

        box(FL_FLAT_BOX);
        color(FL_WHITE);
        redraw();
    }
}
```

11.163.2 Member Function Documentation

11.163.2.1 deleted()

```
int FL_Widget_Tracker::deleted () [inline]
```

Returns 1, if the watched widget has been deleted.

This is a convenience method. You can also use something like

```
if (wp.widget() == 0) // ...
```

where `wp` is an `FL_Widget_Tracker` object.

11.163.2.2 exists()

```
int FL_Widget_Tracker::exists () [inline]
```

Returns 1, if the watched widget exists (has not been deleted).

This is a convenience method. You can also use something like

```
if (wp.widget() != 0) // ...
```

where `wp` is an `FL_Widget_Tracker` object.

11.163.2.3 widget()

```
Fl_Widget * Fl_Widget_Tracker::widget () [inline]
```

Returns a pointer to the watched widget.

This pointer is NULL, if the widget has been deleted.

The documentation for this class was generated from the following files:

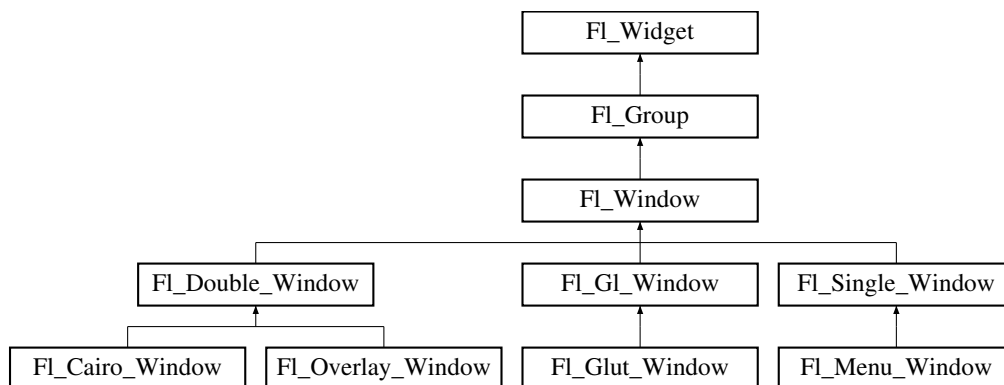
- [Fl.H](#)
- [Fl.cxx](#)

11.164 Fl_Window Class Reference

This widget produces an actual window.

```
#include <Fl_Window.H>
```

Inheritance diagram for Fl_Window:



Public Types

- typedef struct HICON__ * **HICON**

Public Member Functions

- void [allow_expand_outside_parent](#) ()
Allow this subwindow to expand outside the area of its parent window.
- virtual class [Fl_Double_Window](#) * [as_double_window](#) ()
Return non-null if this is an [Fl_Double_Window](#) object.
- virtual class [Fl_Overlay_Window](#) * [as_overlay_window](#) ()
Return non-null if this is an [Fl_Overlay_Window](#) object.
- [Fl_Window](#) const * [as_window](#) () const **FL_OVERRIDE**
- [Fl_Window](#) * [as_window](#) () **FL_OVERRIDE**
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- unsigned int **border** () const
Returns whether the window possesses a border.
- void [border](#) (int b)
Sets whether or not the window manager border is around the window.
- void [clear_border](#) ()
Fast inline function to turn the window manager border off.
- void [clear_modal_states](#) ()
Clears the "modal" flags and converts a "modal" or "non-modal" window back into a "normal" window.
- void **copy_label** (const char *a)
Sets the window titlebar label to a copy of a character string.
- void [cursor](#) (const [Fl_RGB_Image](#) *, int, int)

- Changes the cursor for this window using the provided image as cursor's shape.*

 - void **cursor** (FL_Cursor c, FL_Color, FL_Color=FL_WHITE)

For back compatibility only.
- void **cursor** (FL_Cursor)

Changes the cursor for this window.
- int **decorated_h** () const

Returns the window height including any window title bar and any frame added by the window manager.
- int **decorated_w** () const

Returns the window width including any frame added by the window manager.
- void **default_cursor** (FL_Cursor c, FL_Color, FL_Color=FL_WHITE)

For back compatibility only.
- void **default_cursor** (FL_Cursor)

Sets the default window cursor.
- void **draw_backdrop** ()

Draw the background image if one is set and is aligned inside.
- **FL_Window** (int w, int h, const char *title=0)

Creates a window from the given width w, height h, and title.
- **FL_Window** (int x, int y, int w, int h, const char *title=0)

Creates a window from the given position (x, y), size (w, h) and title.
- virtual void **flush** ()

*Forces the window to be drawn, this window is also made current and calls **draw()**.*
- void **free_position** ()

*Undoes the effect of a previous **resize()** or **show()** so that the next time **show()** is called the window manager is free to position the window.*
- void **fullscreen** ()

Makes the window completely fill one or more screens, without any window manager border visible.
- unsigned int **fullscreen_active** () const

Returns non zero if FULLSCREEN flag is set, 0 otherwise.
- void **fullscreen_off** ()

*Turns off any side effects of **fullscreen()***
- void **fullscreen_off** (int X, int Y, int W, int H)

*Turns off any side effects of **fullscreen()** and does **resize(x,y,w,h)**.*
- void **fullscreen_screens** (int top, int bottom, int left, int right)

Sets which screens should be used when this window is in fullscreen mode.
- **uchar get_size_range** (int *minw, int *minh, int *maxw=NULL, int *maxh=NULL, int *dw=NULL, int *dh=NULL, int *aspect=NULL)

Gets the allowable range to which the user can resize this window.
- int **handle** (int) **FL_OVERRIDE**

Handles the specified event.
- void **hide** () **FL_OVERRIDE**

Removes the window from the screen.
- void **hotspot** (const FL_Widget &p, int offscreen=0)

*See void **FL_Window::hotspot**(int x, int y, int offscreen = 0)*
- void **hotspot** (const FL_Widget *, int offscreen=0)

*See void **FL_Window::hotspot**(int x, int y, int offscreen = 0)*
- void **hotspot** (int x, int y, int offscreen=0)

Positions the window so that the mouse is pointing at the given position, or at the center of the given widget, which may be the window itself.
- const void * **icon** () const

Gets the current icon window target dependent data.
- void **icon** (const FL_RGB_Image *)

- Sets or resets a single window icon.*

 - void **icon** (const void *ic)

Platform-specific method to set the window icon usable on Windows and X11 only.
- void **iconize** ()

Iconifies the window.
- const char * **iconlabel** () const

See void [FL_Window::iconlabel\(const char\)](#)*
- void **iconlabel** (const char *)

Sets the icon label.
- void **icons** (const [FL_RGB_Image](#) *[], int)

Sets the window icons.
- void **icons** (HICON big_icon, HICON small_icon)

Sets the window icons using HICON handles (Windows platform only).
- const char * **label** () const

See void [FL_Window::label\(const char\)](#)*
- void **label** (const char *)

Sets the window title bar label.
- void **label** (const char *label, const char *[iconlabel](#))

Sets the icon label.
- void **make_current** ()

Sets things up so that the drawing functions in [<FL/fl_draw.H>](#) will go into this window.
- void **maximize** ()

Maximizes a top-level window to its current screen.
- unsigned int **maximize_active** () const

Returns whether the window is currently maximized.
- unsigned int **menu_window** () const

Returns true if this window is a menu window.
- unsigned int **modal** () const

Returns true if this window is modal.
- unsigned int **non_modal** () const

Returns true if this window is modal or non-modal.
- [fl_uintptr_t](#) **os_id** ()

Returns a platform-specific identification of a shown window, or 0 if not shown.
- unsigned int **override** () const

Returns non zero if OVERRIDE flag is set, 0 otherwise.
- void **resize** (int X, int Y, int W, int H) [FL_OVERRIDE](#)

Changes the size and position of the window.
- int **screen_num** ()

The number of the screen containing the mapped window.
- void **screen_num** (int screen_num)

Set the number of the screen where to map the window.
- void **set_menu_window** ()

Marks the window as a menu window.
- void **set_modal** ()

A "modal" window, when [shown\(\)](#), will prevent any events from being delivered to other windows in the same program, and will also remain on top of the other windows (if the X window manager supports the "transient for" property).
- void **set_non_modal** ()

A "non-modal" window (terminology borrowed from Microsoft Windows) acts like a [modal\(\)](#) one in that it remains on top, but it has no effect on event delivery.

- void **set_override** ()
Activates the flags NOBORDER|OVERRIDE.
- void **set_tooltip_window** ()
Marks the window as a tooltip window.
- const **FL_Image** * **shape** ()
Returns the image controlling the window shape or NULL.
- void **shape** (const **FL_Image** &b)
Set the window's shape with an FL_Image.
- void **shape** (const **FL_Image** *img)
Assigns a non-rectangular shape to the window.
- void **show** () **FL_OVERRIDE**
Puts the window on the screen.
- void **show** (int argc, char **argv)
Puts the window on the screen with show() and parses command-line arguments.
- int **shown** ()
Returns non-zero if show() has been called (but not hide()).
- void **size_range** (int minw, int minh, int maxw=0, int maxh=0, int dw=0, int dh=0, int aspect=0)
Sets the allowable range to which the user can resize this window.
- unsigned int **tooltip_window** () const
Returns true if this window is a tooltip window.
- void **un_maximize** ()
Returns a previously maximized top-level window to its previous size.
- void **wait_for_expose** ()
Waits for the window to be displayed after calling show().
- int **x_root** () const
Gets the x position of the window on the screen.
- const char * **xclass** () const
Returns the xclass for this window, or a default.
- void **xclass** (const char *c)
Sets the xclass for this window.
- int **y_root** () const
Gets the y position of the window on the screen.
- virtual **~FL_Window** ()
The destructor also deletes all the children.

Public Member Functions inherited from FL_Group

- **FL_Widget** *& **_ddfdesign_kludge** ()
This is for forms compatibility only.
- void **add** (**FL_Widget** &)
The widget is removed from its current group (if any) and then added to the end of this group.
- void **add** (**FL_Widget** *o)
See void FL_Group::add(FL_Widget &w)
- void **add_resizable** (**FL_Widget** &o)
Adds a widget to the group and makes it the resizable widget.
- **FL_Widget** *const * **array** () const
Returns a pointer to the internal array of children.
- **FL_Group** const * **as_group** () const **FL_OVERRIDE**
- **FL_Group** * **as_group** () **FL_OVERRIDE**
Returns an FL_Group pointer if this widget is an FL_Group.
- void **begin** ()

- Sets the current group so you can build the widget tree by just constructing the widgets.*

 - `FL_Widget * child (int n) const`
Returns the n'th child.
 - `int children () const`
Returns how many child widgets the group has.
 - `void clear ()`
Deletes all child widgets from memory recursively.
 - `unsigned int clip_children ()`
Returns the current clipping mode.
 - `void clip_children (int c)`
Controls whether the group widget clips the drawing of child widgets to its bounding box.
 - `virtual int delete_child (int n)`
Removes the widget at `index` from the group and deletes it.
 - `void end ()`
Exactly the same as `current(this->parent())`.
 - `int find (const FL_Widget &o) const`
See `int FL_Group::find(const FL_Widget *w) const`.
 - `int find (const FL_Widget *) const`
Searches the children for the widget and returns the index.
 - `FL_Group (int, int, int, int, const char * = 0)`
Creates a new `FL_Group` widget using the given position, size, and label string.
 - `void focus (FL_Widget *W)`
 - `void forms_end ()`
This is for forms compatibility only.
 - `void init_sizes ()`
Resets the internal array of widget sizes and positions.
 - `void insert (FL_Widget &, int i)`
The widget is removed from its current group (if any) and then inserted into this group.
 - `void insert (FL_Widget &o, FL_Widget *before)`
This does `insert(w, find(before))`.
 - `void remove (FL_Widget &)`
Removes a widget from the group but does not delete it.
 - `void remove (FL_Widget *o)`
Removes the widget `o` from the group.
 - `void remove (int index)`
Removes the widget at `index` from the group but does not delete it.
 - `FL_Widget * resizable () const`
Returns the group's resizable widget.
 - `void resizable (FL_Widget &o)`
Sets the group's resizable widget.
 - `void resizable (FL_Widget *o)`
The resizable widget defines both the resizing box and the resizing behavior of the group and its children.
 - `virtual ~FL_Group ()`
The destructor also deletes all the children.

Public Member Functions inherited from FI_Widget

- void **_clear_fullscreen** ()
- void **_set_fullscreen** ()
- void **activate** ()
Activates the widget.
- unsigned int **active** () const
Returns whether the widget is active.
- int **active_r** () const
Returns whether the widget and all of its parents are active.
- **FI_Align align** () const
Gets the label alignment.
- void **align** (**FI_Align** alignment)
Sets the label alignment.
- long **argument** () const
Gets the current user data (long) argument that is passed to the callback function.
- void **argument** (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class **FI_Gl_Window** * **as_gl_window** ()
Returns an FI_Gl_Window pointer if this widget is an FI_Gl_Window.
- virtual class **FI_Gl_Window** const * **as_gl_window** () const
- void **bind_deimage** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the inactive state.
- void **bind_deimage** (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void **bind_image** (**FI_Image** *img)
Sets the image to use as part of the widget label when in the active state.
- void **bind_image** (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- **FI_Boxtype box** () const
Gets the box type of the widget.
- void **box** (**FI_Boxtype** new_box)
Sets the box type for the widget.
- **FI_Callback_p callback** () const
Gets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback** *cb, **FI_Callback_User_Data** *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void **callback** (**FI_Callback** *cb, void *p)
Sets the current callback function and data for the widget.
- void **callback** (**FI_Callback0** *cb)
Sets the current callback function for the widget.
- void **callback** (**FI_Callback1** *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int **changed** () const
Checks if the widget value changed since the last callback.
- void **clear_active** ()
Marks the widget as inactive without sending events or changing focus.
- void **clear_changed** ()
Marks the value of the widget as unchanged.

- void `clear_damage` (`uchar` c=0)
Clears or sets the damage flags.
- void `clear_output` ()
Sets a widget to accept input.
- void `clear_visible` ()
Hides the widget.
- void `clear_visible_focus` ()
Disables keyboard focus navigation with this widget.
- `FL_Color` `color` () const
Gets the background color of the widget.
- void `color` (`FL_Color` bg)
Sets the background color of the widget.
- void `color` (`FL_Color` bg, `FL_Color` sel)
Sets the background and selection color of the widget.
- `FL_Color` `color2` () const
For back compatibility only.
- void `color2` (unsigned a)
For back compatibility only.
- int `contains` (const `FL_Widget` *w) const
Checks if w is a child of this widget.
- void `copy_label` (const char *new_label)
Sets the current label.
- void `copy_tooltip` (const char *text)
Sets the current tooltip text.
- `uchar` `damage` () const
Returns non-zero if `draw()` needs to be called.
- void `damage` (`uchar` c)
Sets the damage bits for the widget.
- void `damage` (`uchar` c, int x, int y, int w, int h)
Sets the damage bits for an area inside the widget.
- int **`damage_resize`** (int, int, int, int)
Internal use only.
- void `deactivate` ()
Deactivates the widget.
- `FL_Image` * `deimage` ()
Gets the image that is used as part of the widget label when in the inactive state.
- const `FL_Image` * `deimage` () const
Gets the image that is used as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the inactive state.
- void `deimage` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the inactive state.
- int `deimage_bound` () const
Returns whether the inactive image is managed by the widget.
- void `do_callback` (`FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with default arguments.
- void `do_callback` (`FL_Widget` *widget, long arg, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `do_callback` (`FL_Widget` *widget, void *arg=0, `FL_Callback_Reason` reason=`FL_REASON_UNKNOWN`)
Calls the widget callback function with arbitrary arguments.
- void `draw_label` (int, int, int, int, `FL_Align`) const

- Draws the label in an arbitrary bounding box with an arbitrary alignment.*
- int `h` () const
Gets the widget height.
- int `horizontal_label_margin` ()
Get the spacing between the label and the horizontal edge of the widget.
- void `horizontal_label_margin` (int px)
Set the spacing between the label and the horizontal edge of the widget.
- `FL_Image` * `image` ()
Gets the image that is used as part of the widget label when in the active state.
- const `FL_Image` * `image` () const
Gets the image that is used as part of the widget label when in the active state.
- void `image` (`FL_Image` &img)
Sets the image to use as part of the widget label when in the active state.
- void `image` (`FL_Image` *img)
Sets the image to use as part of the widget label when in the active state.
- int `image_bound` () const
Returns whether the image is managed by the widget.
- int `inside` (const `FL_Widget` *wgt) const
Checks if this widget is a child of wgt.
- int `is_label_copied` () const
Returns whether the current label was assigned with `copy_label()`.
- const char * `label` () const
Gets the current label text.
- void `label` (const char *text)
Sets the current label pointer.
- void `label` (`FL_Labeltype` a, const char *b)
Shortcut to set the label text and type in one call.
- int `label_image_spacing` ()
Return the gap size between the label and the image.
- void `label_image_spacing` (int gap)
Set the gap between the label and the image in pixels.
- `FL_Color` `labelcolor` () const
Gets the label color.
- void `labelcolor` (`FL_Color` c)
Sets the label color.
- `FL_Font` `labelfont` () const
Gets the font to use.
- void `labelfont` (`FL_Font` f)
Sets the font to use.
- `FL_Fonsize` `labelsize` () const
Gets the font size in pixels.
- void `labelsize` (`FL_Fonsize` pix)
Sets the font size in pixels.
- `FL_Labeltype` `labeltype` () const
Gets the label type.
- void `labeltype` (`FL_Labeltype` a)
Sets the label type.
- void `measure_label` (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool `needs_keyboard` () const
Returns whether this widget needs a keyboard.

- void `needs_keyboard` (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int `output` () const
Returns if a widget is used for output only.
- `FL_Group * parent` () const
Returns a pointer to the parent widget.
- void `parent` (`FL_Group *p`)
Internal use only - "for hacks only".
- void `position` (int X, int Y)
Repositions the window or widget.
- void `redraw` ()
Schedules the drawing of the widget.
- void `redraw_label` ()
Schedules the drawing of the label.
- `FL_Color selection_color` () const
Gets the selection color.
- void `selection_color` (`FL_Color a`)
Sets the selection color.
- void `set_active` ()
Marks the widget as active without sending events or changing focus.
- void `set_changed` ()
Marks the value of the widget as changed.
- void `set_output` ()
Sets a widget to output only.
- void `set_visible` ()
Makes the widget visible.
- void `set_visible_focus` ()
Enables keyboard focus navigation with this widget.
- int `shortcut_label` () const
Returns whether the widget's label uses '&' to indicate shortcuts.
- void `shortcut_label` (int value)
Sets whether the widget's label uses '&' to indicate shortcuts.
- void `size` (int W, int H)
Changes the size of the widget.
- int `take_focus` ()
Gives the widget the keyboard focus.
- unsigned int `takeevents` () const
Returns if the widget is able to take events.
- int `test_shortcut` ()
Returns true if the widget's label contains the entered '&x' shortcut.
- const char * `tooltip` () const
Gets the current tooltip text.
- void `tooltip` (const char *text)
Sets the current tooltip text.
- `FL_Window * top_window` () const
Returns a pointer to the top-level window for the widget.
- `FL_Window * top_window_offset` (int &xoff, int &yoff) const
Finds the x/y offset of the current widget relative to the top-level window.
- `uchar type` () const
Gets the widget type.
- void `type` (`uchar t`)

- Sets the widget type.*
- int **use_accents_menu** ()
 - Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.*
- void * **user_data** () const
 - Gets the user data for this widget.*
- void **user_data** (Fl_Callback_User_Data *v, bool auto_free)
 - Sets the user data for this widget.*
- void **user_data** (void *v)
 - Sets the user data for this widget.*
- int **vertical_label_margin** ()
 - Get the spacing between the label and the vertical edge of the widget.*
- void **vertical_label_margin** (int px)
 - Set the spacing between the label and the vertical edge of the widget.*
- unsigned int **visible** () const
 - Returns whether a widget is visible.*
- unsigned int **visible_focus** () const
 - Checks whether this widget has a visible focus.*
- void **visible_focus** (int v)
 - Modifies keyboard focus navigation.*
- int **visible_r** () const
 - Returns whether a widget and all its parents are visible.*
- int **w** () const
 - Gets the widget width.*
- Fl_When **when** () const
 - Returns the conditions under which the callback is called.*
- void **when** (uchar i)
 - Sets the flags used to decide when a callback is called.*
- Fl_Window * **window** () const
 - Returns a pointer to the nearest parent window up the widget hierarchy.*
- int **x** () const
 - Gets the widget position in its window.*
- int **y** () const
 - Gets the widget position in its window.*
- virtual ~Fl_Widget ()
 - Destroys the widget.*

Static Public Member Functions

- static Fl_Window * **current** ()
 - Returns the last window that was made current.*
- static void **default_callback** (Fl_Window *, void *v)
 - Back compatibility: Sets the default callback v for win to call on close event.*
- static void **default_icon** (const Fl_RGB_Image *)
 - Sets a single default window icon.*
- static void **default_icons** (const Fl_RGB_Image *[], int)
 - Sets the default window icons.*
- static void **default_icons** (HICON big_icon, HICON small_icon)
 - Sets the default window icons (Windows platform only).*
- static const char * **default_xclass** ()
 - Returns the default xclass.*
- static void **default_xclass** (const char *)

- *Sets the default window xclass.*
- static bool `is_a_rescale ()`
Returns true when a window is being rescaled.
- static char `show_next_window_iconic ()`
Returns the static flag whether the next window should be opened iconified.
- static void `show_next_window_iconic (char stat)`
Sets a static flag whether the next window should be opened iconified.

Static Public Member Functions inherited from `Fl_Group`

- static `Fl_Group * current ()`
Returns the currently active group.
- static void `current (Fl_Group *g)`
Sets the current group.

Static Public Member Functions inherited from `Fl_Widget`

- static void `default_callback (Fl_Widget *widget, void *data)`
The default callback for all widgets that don't set a callback.
- static unsigned int `label_shortcut (const char *t)`
Returns the Unicode value of the '&x' shortcut in a given text.
- static int `test_shortcut (const char *, const bool require_alt=false)`
Returns true if the given text t contains the entered '&x' shortcut.

Protected Member Functions

- void `default_size_range ()`
Protected method to calculate the default size range of a window.
- void `draw () FL_OVERRIDE`
Draws the widget.
- int `force_position () const`
Returns the internal state of the window's FORCE_POSITION flag.
- void `force_position (int force)`
Sets an internal flag that tells FLTK and the window manager to honor position requests.
- void `free_icons ()`
Deletes all icons previously attached to the window.
- int `is_resizable ()`
Protected method to determine whether a window is resizable.

Protected Member Functions inherited from `Fl_Group`

- `Fl_Rect * bounds ()`
Returns the internal array of widget sizes and positions.
- void `draw_child (Fl_Widget &widget) const`
Forces a child to redraw.
- void `draw_children ()`
Draws all children of the group.
- void `draw_outside_label (const Fl_Widget &widget) const`
Parents normally call this to draw outside labels of child widgets.
- virtual int `on_insert (Fl_Widget *, int)`
Allow derived groups to act when a widget is added as a child.
- virtual int `on_move (int, int)`
Allow derived groups to act when a widget is moved within the group.

- virtual void [on_remove](#) (int)
Allow derived groups to act when a child widget is removed from the group.
- int * [sizes](#) ()
Returns the internal array of widget sizes and positions.
- void [update_child](#) ([FL_Widget](#) &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from [FL_Widget](#)

- void [clear_flag](#) (unsigned int c)
Clears a flag in the flags mask.
- void [draw_backdrop](#) () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void [draw_box](#) () const
Draws the widget box according its box style.
- void [draw_box](#) ([FL_Boxtype](#) t, [FL_Color](#) c) const
Draws a box of type t, of color c at the widget's position and size.
- void [draw_box](#) ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void [draw_focus](#) () const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FL_Boxtype](#) t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void [draw_focus](#) ([FL_Boxtype](#) t, int x, int y, int w, int h, [FL_Color](#) bg) const
Draws a focus box for the widget at the given position and size.
- void [draw_label](#) () const
Draws the widget's label at the defined label position.
- void [draw_label](#) (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- [FL_Widget](#) (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int [flags](#) () const
Gets the widget flags mask.
- void [h](#) (int v)
Internal use only.
- void [set_flag](#) (unsigned int c)
Sets a flag in the flags mask.
- void [w](#) (int v)
Internal use only.
- void [x](#) (int v)
Internal use only.
- void [y](#) (int v)
Internal use only.

Static Protected Attributes

- static [FL_Window](#) * [current_](#)
Stores the last window that was made current.

Friends

- class [FL_Window_Driver](#)
- class [FL_X](#)

Additional Inherited Members

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }

flags possible values enumeration.

11.164.1 Detailed Description

This widget produces an actual window.

This can either be a main window, with a border and title and all the window management controls, or a "subwindow" inside a window. This is controlled by whether or not the window has a [parent\(\)](#).

Once you create a window, you usually add children [Fl_Widget](#)'s to it by using `window->add(child)` for each new widget. See [Fl_Group](#) for more information on how to add and remove children.

There are several subclasses of [Fl_Window](#) that provide double-buffering, overlay, menu, and OpenGL support.

The window's callback is done if the user tries to close a window using the window manager and [Fl::modal\(\)](#) is zero or equal to the window. [Fl_Window](#) has a default callback that calls [Fl_Window::hide\(\)](#). Callback reasons can be [FL_REASON_CANCELLED](#) if the Escape key was pressed, or [FL_REASON_CLOSED](#) when the close button is clicked. [FL_WHEN_...](#) flags are ignored.

11.164.2 Constructor & Destructor Documentation

11.164.2.1 [Fl_Window\(\)](#) [1/2]

```
Fl_Window::Fl_Window (
    int w,
    int h,
    const char * title = 0)
```

Creates a window from the given width `w`, height `h`, and `title`.

If [Fl_Group::current\(\)](#) is not NULL, the window is created as a subwindow of the parent window.

The (w, h) form of the constructor creates a top-level window and asks the window manager to position the window.

The (x, y, w, h) form of the constructor either creates a subwindow or a top-level window at the specified location (x, y), subject to window manager configuration. If you do not specify the position of the window, the window manager will pick a place to show the window or allow the user to pick a location. Use [position\(x, y\)](#) or [hotspot\(\)](#) before calling [show\(\)](#) to request a position on the screen. See [Fl_Window::resize\(\)](#) for some more details on positioning windows. Top-level windows initially have [visible\(\)](#) set to 0 and [parent\(\)](#) set to NULL. Subwindows initially have [visible\(\)](#) set to 1 and [parent\(\)](#) set to the parent window pointer.

[Fl_Widget::box\(\)](#) defaults to [FL_FLAT_BOX](#). If you plan to completely fill the window with children widgets you should change this to [FL_NO_BOX](#). If you turn the window border off you may want to change this to [FL_UP_BOX](#).

See also

[Fl_Window\(int x, int y, int w, int h, const char *title\)](#)

11.164.2.2 [Fl_Window\(\)](#) [2/2]

```
Fl_Window::Fl_Window (
```

```

    int x,
    int y,
    int w,
    int h,
    const char * title = 0)

```

Creates a window from the given position (x, y), size (w, h) and title.

On a multi-screen system, the values computed by [Fl::screen_xywh\(int &X, int &Y, int &W, int &H, int n\)](#) can be used to discover the coordinates of the area of screen #n. When these screens have various scale factor values, an (x, y) pair may not be enough to specify the targeted screen for the window, because the same (x,y) pair can belong to several screens. In that situation, a call to [Fl_Window::screen_num\(int\)](#) is to be used to identify unambiguously the targeted screen.

See also

```

Fl\_Window\(int w, int h, const char \*title\)
Fl::screen\_xywh\(int &X, int &Y, int &W, int &H, int n\)

```

Note

Under Wayland, it's generally not possible for the client app to control the position of a window in the system. It's possible to specify on what screen should the compositor place a fullscreen window. It's also possible to make an [Fl_Window](#) the child of another window or group and control with `x` and `y` its screen position relatively to the enclosing window. Apply member function [Fl_Window::allow_expand_outside_parent\(\)](#) to the child window to allow it to expand partially or totally outside its parent.

11.164.2.3 ~Fl_Window()

```
Fl_Window::~Fl_Window () [virtual]
```

The destructor *also deletes all the children*.

This allows a whole tree to be deleted at once, without having to keep a pointer to all the children in the user code. A kludge has been done so the [Fl_Window](#) and all of its children can be automatic (local) variables, but you must declare the [Fl_Window](#) *first* so that it is destroyed last.

11.164.3 Member Function Documentation

11.164.3.1 allow_expand_outside_parent()

```
void Fl_Window::allow_expand_outside_parent ()
```

Allow this subwindow to expand outside the area of its parent window.

This is presently implemented only for the Wayland platform to help support window docking.

Since

1.4.0

11.164.3.2 as_double_window()

```
virtual class Fl_Double_Window * Fl_Window::as_double_window () [inline], [virtual]
```

Return non-null if this is an [Fl_Double_Window](#) object.

Reimplemented in [Fl_Double_Window](#).

11.164.3.3 as_overlay_window()

```
virtual class Fl_Overlay_Window * Fl_Window::as_overlay_window () [inline], [virtual]
```

Return non-null if this is an [Fl_Overlay_Window](#) object.

Reimplemented in [Fl_Overlay_Window](#).

11.164.3.4 as_window() [1/2]

```
Fl_Window const * Fl_Window::as_window () const [inline], [virtual]
```

Reimplemented from [Fl_Widget](#).

11.164.3.5 as_window() [2/2]

```
Fl_Window * Fl_Window::as_window () [inline], [virtual]
```

Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).

Use this method if you have a widget (pointer) and need to know whether this widget is derived from [Fl_Window](#). If it returns non-NULL, then the widget in question is derived from [Fl_Window](#), and you can use the returned pointer to access its children or other [Fl_Window](#)-specific methods.

Return values

NULL	if this widget is not derived from Fl_Window .

Note

This method is provided to avoid `dynamic_cast`.

See also

[Fl_Widget::as_group\(\)](#), [Fl_Widget::as_gl_window\(\)](#)

Reimplemented from [Fl_Widget](#).

11.164.3.6 border()

```
void Fl_Window::border (
    int b)
```

Sets whether or not the window manager border is around the window.

The default value is true. The macOS platform ignores requests to change the border state of a fullscreen or maximized window. *With some X window managers, this does not work after [show\(\)](#) has been called.*

11.164.3.7 clear_border()

```
void Fl_Window::clear_border () [inline]
```

Fast inline function to turn the window manager border off.

It only works before [show\(\)](#) is called.

11.164.3.8 clear_modal_states()

```
void Fl_Window::clear_modal_states () [inline]
```

Clears the "modal" flags and converts a "modal" or "non-modal" window back into a "normal" window.

Note that there are *three* states for a window: modal, non-modal, and normal.

You can not change the "modality" of a window whilst it is shown, so it is necessary to first [hide\(\)](#) the window, change its "modality" as required, then re-show the window for the new state to take effect.

This method can also be used to change a "modal" window into a "non-modal" one. On several supported platforms, the "modal" state over-rides the "non-modal" state, so the "modal" state must be cleared before the window can be set into the "non-modal" state. In general, the following sequence should work:

```
win->hide();
win->clear_modal_states();
// Set win to new state as desired, or leave "normal", e.g...
win->set_non_modal();
win->show();
```

Note

Under some window managers, the sequence of hiding the window and changing its modality will often cause it to be re-displayed at a different position when it is subsequently shown. This is an irritating feature but appears to be unavoidable at present. As a result we would advise to use this method only when absolutely necessary.

See also

void [set_modal\(\)](#), void [set_non_modal\(\)](#)

11.164.3.9 current()

```
Fl_Window * Fl_Window::current () [static]
```

Returns the last window that was made current.

See also

[Fl_Window::make_current\(\)](#)

11.164.3.10 cursor() [1/3]

```
void Fl_Window::cursor (
    const Fl_RGB_Image * image,
    int hotx,
    int hoty)
```

Changes the cursor for this window using the provided image as cursor's shape.

The window must be [show\(\)](#) for this function to have any effect. This always calls the system. If you are changing the cursor a lot you may want to keep track of how you set it in a static variable and call this only if the new cursor is different.

The default cursor will be used if the provided image cannot be used as a cursor.

Parameters

<i>image</i>	Sets the cursor size and shape
<i>hotx,hoty</i>	Sets the cursor's active location relatively to top-left of <i>image</i> when clicking

See also

[cursor\(Fl_Cursor\)](#), [default_cursor\(\)](#)

11.164.3.11 cursor() [2/3]

```
void Fl_Window::cursor (
    Fl_Cursor c,
    Fl_Color ,
    Fl_Color = FL_WHITE)
```

For back compatibility only.

Same as [Fl_Window::cursor\(Fl_Cursor\)](#)

11.164.3.12 cursor() [3/3]

```
void Fl_Window::cursor (
    Fl_Cursor c)
```

Changes the cursor for this window.

The window must be [show\(\)](#) for this function to have any effect. This always calls the system. If you are changing the cursor a lot you may want to keep track of how you set it in a static variable and call this only if the new cursor is different.

The type `Fl_Cursor` is an enumeration defined in [<FL/Enumerations.H>](#).

See also

[cursor\(const Fl_RGB_Image*, int, int\)](#), [default_cursor\(\)](#)

11.164.3.13 decorated_h()

```
int Fl_Window::decorated_h () const
```

Returns the window height including any window title bar and any frame added by the window manager.

Same as [h\(\)](#) if applied to a subwindow, or if window is not yet mapped.

Note

Under X11, FLTK is able to compute the size of window titlebars and borders only if these decoration elements are strictly X11-based. When that's not the case, [decorated_h\(\)](#) returns the same value as [h\(\)](#) and [decorated_w\(\)](#) as [w\(\)](#), and FLTK cannot access window decorations.

Under X11 again, the values returned by [decorated_h\(\)](#) and [decorated_w\(\)](#) may not be reliable **during a resize operation**. The size of decoration elements of a window is best computed when the window is first mapped.

11.164.3.14 decorated_w()

```
int Fl_Window::decorated_w () const
```

Returns the window width including any frame added by the window manager.
Same as [w\(\)](#) if applied to a subwindow, or if window is not yet mapped.

See also

[decorated_h\(\)](#).

11.164.3.15 default_cursor() [1/2]

```
void Fl_Window::default_cursor (
    Fl_Cursor c,
    Fl_Color ,
    Fl_Color = FL_WHITE)
```

For back compatibility only.
same as [Fl_Window::default_cursor\(Fl_Cursor\)](#)

11.164.3.16 default_cursor() [2/2]

```
void Fl_Window::default_cursor (
    Fl_Cursor c)
```

Sets the default window cursor.
This is the cursor that will be used after the mouse pointer leaves a widget with a custom cursor set.

See also

[cursor\(const Fl_RGB_Image*, int, int\), default_cursor\(\)](#)

11.164.3.17 default_icon()

```
void Fl_Window::default_icon (
    const Fl_RGB_Image * icon) [static]
```

Sets a single default window icon.
If *icon* is NULL the current default icons are removed.

Parameters

in	<i>icon</i>	default icon for all windows subsequently created or NULL

See also

[Fl_Window::default_icons\(const Fl_RGB_Image *\[\], int\)](#)
[Fl_Window::icon\(const Fl_RGB_Image *\)](#)
[Fl_Window::icons\(const Fl_RGB_Image *\[\], int\)](#)

Note

See [Window icons](#) for the Wayland platform.

11.164.3.18 default_icons() [1/2]

```
void Fl_Window::default_icons (
    const Fl_RGB_Image * icons[],
    int count) [static]
```

Sets the default window icons.

The default icons are used for all windows that don't have their own icons set before [show\(\)](#) is called. You can change the default icons whenever you want, but this only affects windows that are created (and shown) after this call.

The given images in `icons` are copied. You can use a local variable or free the images immediately after this call.

Parameters

in	<i>icons</i>	default icons for all windows subsequently created
in	<i>count</i>	number of images in <i>icons</i> . Set to 0 to remove the current default icons

See also

[Fl_Window::default_icon\(const Fl_RGB_Image *\)](#)

[Fl_Window::icon\(const Fl_RGB_Image *\)](#)

[Fl_Window::icons\(const Fl_RGB_Image *\[\], int\)](#)

Note

See [Window icons](#) for the Wayland platform.

11.164.3.19 default_icons() [2/2]

```
static void Fl_Window::default_icons (
    HICON big_icon,
    HICON small_icon) [static]
```

Sets the default window icons (Windows platform only).

Convenience function to set the default icons using Windows' native HICON icon handles.

The given icons are copied. You can free the icons immediately after this call.

Parameters

in	<i>big_icon</i>	default large icon for all windows subsequently created
in	<i>small_icon</i>	default small icon for all windows subsequently created

See also

[Fl_Window::default_icon\(const Fl_RGB_Image *\)](#)

[Fl_Window::default_icons\(const Fl_RGB_Image *\[\], int\)](#)

[Fl_Window::icon\(const Fl_RGB_Image *\)](#)

[Fl_Window::icons\(const Fl_RGB_Image *\[\], int\)](#)

[Fl_Window::icons\(HICON, HICON\)](#)

11.164.3.20 default_size_range()

```
void Fl_Window::default_size_range () [protected]
```

Protected method to calculate the default size range of a window.

This method is called internally prior to showing a window to ensure that the window's size range values are calculated if a [resizable\(\)](#) widget has been set but [size_range\(\)](#) has not been called explicitly.

This method does nothing if [size_range\(\)](#) has been called before.

Otherwise FLTK tries to figure out the window's size range from the setting of the window's [resizable\(\)](#) widget as follows and roughly in the given order.

1. If `resizable()` is NULL (this is the default) then the window cannot be resized and the resize border and max-size control will not be displayed for the window.
2. If either dimension of `resizable()` is zero, then the window cannot resize in that direction.
3. The `resizable()` widget is clipped to the window area.
4. The non-resizable portion of the window is calculated as the difference of the window's size and the clipped `resizable()` widget's size.
5. If either dimension of the clipped `resizable()` widget is greater than 100, then 100 is considered its minimum width/height. This allows the resizable widget to shrink below its original size.
6. Finally the minimum width/height of the window is set to the non-resizable portion plus the width/height of the `resizable()` widget as calculated above.

In simple words:

- It is assumed that the `resizable()` widget can be indefinitely enlarged and/or shrunk to a minimum width/height of 100 unless it is smaller than that, which is then considered the minimum.
- The window's `size_range()` minimum values are set to the sum of the non-resizable portion of the window and the previously calculated minimum size of the `resizable()` widget.

Examples:

```
Fl_Window win(400, 400);
win.resizable(win);
// win.size_range(100, 100, 0, 0);
```

The minimum size of the resizable is 100, hence the minimum size of the total window is also 100 in both directions.

```
Fl_Window win(400, 400);
Fl_Box box(20, 20, 360, 360);
win.resizable(box);
// win.size_range(140, 140, 0, 0);
```

The calculated minimum width and height would be 20 + 100 + 20 in both dimensions.

```
Fl_Window win(400, 400);
Fl_Box box(200, 0, 500, 300); // note: width 500 too large: clipped
win.resizable(box);
// win.size_range(300, 200, 0, 0);
```

The width of the resizable is clipped to 200, hence the minimum size of the total window is also 200 (fix) + 100 (min. resizable) in x direction. The minimum value in y direction is 100 (resizable) + 100 (fixed part).

The calculation is based on clipping the resizable widget to the window area to prevent programming errors and the assumption that the resizable widget can be shrunk to 100x100 or its original size, whichever is smaller.

If this is not what you want, please use `Fl_Window::size_range()` explicitly so you can set any appropriate range.

Since

1.4.0

11.164.3.21 default_xclass() [1/2]

```
const char * Fl_Window::default_xclass () [static]
```

Returns the default xclass.

See also

[Fl_Window::default_xclass\(const char *\)](#)

11.164.3.22 default_xclass() [2/2]

```
void Fl_Window::default_xclass (
    const char * xc) [static]
```

Sets the default window xclass.

The default xclass is used for all windows that don't have their own xclass set before `show()` is called. You can change the default xclass whenever you want, but this only affects windows that are created (and shown) after this call.

The given string `xc` is copied. You can use a local variable or free the string immediately after this call.

If you don't call this, the default xclass for all windows will be "FLTK". You can reset the default xclass by specifying NULL for `xc`.

If you call `Fl_Window::xclass(const char *)` for any window, then this also sets the default xclass, unless it has been set before.

Parameters

in	xc	default xclass for all windows subsequently created

See also

[Fl_Window::xclass\(const char *\)](#)

11.164.3.23 draw()

```
void Fl_Window::draw () [protected], [virtual]
```

Draws the widget.

Never call this function directly. FLTK will schedule redrawing whenever needed. If your widget must be redrawn as soon as possible, call [redraw\(\)](#) instead.

Override this function to draw your own widgets.

If you ever need to call another widget's draw method *from within your own [draw\(\)](#) method*, e.g. for an embedded scrollbar, you can do it (because [draw\(\)](#) is virtual) like this:

```
Fl_Widget *s = &scrollbar; // scrollbar is an embedded Fl_Scrollbar
s->draw();                // calls Fl_Scrollbar::draw()
```

Reimplemented from [Fl_Group](#).

11.164.3.24 flush()

```
void Fl_Window::flush () [virtual]
```

Forces the window to be drawn, this window is also made current and calls [draw\(\)](#).

Reimplemented in [Fl_Double_Window](#), [Fl_Gl_Window](#), [Fl_Overlay_Window](#), and [Fl_Single_Window](#).

11.164.3.25 force_position() [1/2]

```
int Fl_Window::force_position () const [inline], [protected]
```

Returns the internal state of the window's FORCE_POSITION flag.

Return values

1	if flag is set
0	otherwise

See also

[force_position\(int\)](#)

11.164.3.26 force_position() [2/2]

```
void Fl_Window::force_position (
    int force) [inline], [protected]
```

Sets an internal flag that tells FLTK and the window manager to honor position requests.

This is used internally and should not be needed by user code.

Parameters

in	force	1 to set the FORCE_POSITION flag, 0 to clear it

11.164.3.27 free_icons()

```
void Fl_Window::free_icons () [protected]
```

Deletes all icons previously attached to the window.

See also

[Fl_Window::icons\(const Fl_RGB_Image *icons\[\], int count\)](#)

11.164.3.28 free_position()

```
void Fl_Window::free_position () [inline]
```

Undoes the effect of a previous [resize\(\)](#) or [show\(\)](#) so that the next time [show\(\)](#) is called the window manager is free to position the window.

This is for Forms compatibility only.

Deprecated please use [force_position\(0\)](#) instead

11.164.3.29 fullscreen()

```
void Fl_Window::fullscreen ()
```

Makes the window completely fill one or more screens, without any window manager border visible.

You must use [fullscreen_off\(\)](#) to undo this.

Note

On some platforms, this can result in the keyboard being grabbed. The window may also be recreated, meaning [hide\(\)](#) and [show\(\)](#) will be called.

See also

[void Fl_Window::fullscreen_screens\(\)](#)

11.164.3.30 fullscreen_screens()

```
void Fl_Window::fullscreen_screens (
    int top,
    int bottom,
    int left,
    int right)
```

Sets which screens should be used when this window is in fullscreen mode.

The window will be resized to the top of the screen with index `top`, the bottom of the screen with index `bottom`, etc.

If this method is never called, or if any argument is < 0 , then the window will be resized to fill the screen it is currently on.

See also

[void Fl_Window::fullscreen\(\)](#)

11.164.3.31 get_size_range()

```
uchar Fl_Window::get_size_range (
    int * minWidth,
    int * minHeight,
    int * maxWidth = NULL,
    int * maxHeight = NULL,
    int * deltaX = NULL,
    int * deltaY = NULL,
    int * aspectRatio = NULL)
```

Gets the allowable range to which the user can resize this window.

Parameters

out	<i>minWidth,minHeight,maxWidth,maxHeight,deltaX,deltaY,aspectRatio</i>	are all pointers to integers that will receive the current respective value during the call. Every pointer can be NULL if that value is not needed.

Return values

0	if size range not set
1	if the size range was explicitly set by a call to Fl_Window::size_range() or has been calculated

See also

[Fl_Window::size_range\(int minWidth, int minHeight, int maxWidth, int maxHeight, int deltaX, int deltaY, int aspectRatio\)](#)

Since

1.4.0

11.164.3.32 handle()

```
int Fl_Window::handle (
    int event) [virtual]
```

Handles the specified event.

You normally don't call this method directly, but instead let FLTK do it when the user interacts with the widget.

When implemented in a widget, this function must return 0 if the widget does not use the event or 1 otherwise.

Most of the time, you want to call the inherited [handle\(\)](#) method in your overridden method so that you don't short-circuit events that you don't handle. In this last case you should return the callee retval.

One exception to the rule in the previous paragraph is if you really want to *override* the behavior of the base class. This requires knowledge of the details of the inherited class.

In rare cases you may want to return 1 from your [handle\(\)](#) method although you don't really handle the event. The effect would be to *filter* event processing, for instance if you want to dismiss non-numeric characters (keypresses) in a numeric input widget. You may "ring the bell" or show another visual indication or drop the event silently. In such a case you must not call the [handle\(\)](#) method of the base class and tell FLTK that you *consumed* the event by returning 1 even if you didn't *do* anything with it.

Parameters

in	<i>event</i>	the kind of event received

Return values

0	if the event was not used or understood
1	if the event was used and can be deleted

See also

[Fl_Event](#)

Reimplemented from [Fl_Group](#).

11.164.3.33 hide()

```
void Fl_Window::hide () [virtual]
```

Removes the window from the screen.

If the window is already hidden or has not been shown then this does nothing and is harmless.

Reimplemented from [Fl_Widget](#).

11.164.3.34 hotspot()

```
void Fl_Window::hotspot (
    int x,
    int y,
    int offscreen = 0)
```

Positions the window so that the mouse is pointing at the given position, or at the center of the given widget, which may be the window itself.

If the optional offscreen parameter is non-zero, then the window is allowed to extend off the screen (this does not work with some X window managers).

See also

[position\(\)](#)

11.164.3.35 icon() [1/3]

```
const void * Fl_Window::icon () const
```

Gets the current icon window target dependent data.

Deprecated in 1.3.3

11.164.3.36 icon() [2/3]

```
void Fl_Window::icon (
    const Fl_RGB_Image * icon)
```

Sets or resets a single window icon.

A window icon *can* be changed while the window is shown, but this *may* be platform and/or window manager dependent. To be sure that the window displays the correct window icon you should always set the icon before the window is shown.

If a window icon has not been set for a particular window, then the default window icon (see links below) or the system default icon will be used.

This method makes an internal copy of the `icon` pixel buffer, so once set, the [Fl_RGB_Image](#) instance can be freed by the caller.

Parameters

in	<i>icon</i>	icon for this window, NULL to reset window icon.

See also

[Fl_Window::default_icon\(const Fl_RGB_Image *\)](#)

[Fl_Window::default_icons\(const Fl_RGB_Image *\[\], int\)](#)

[Fl_Window::icons\(const Fl_RGB_Image *\[\], int\)](#)

Note

See [Window icons](#) for the Wayland platform.

11.164.3.37 icon() [3/3]

```
void Fl_Window::icon (
    const void * ic)
```

Platform-specific method to set the window icon usable on Windows and X11 only.

See [Setting the Icon of a Window](#) for its use under X11, and [Setting the Icon of a Window](#) under Windows.

Deprecated in 1.3.3 in favor of platform-independent methods [Fl_Window::icon\(const Fl_RGB_Image *icon\)](#) and [Fl_Window::icons\(const Fl_RGB_Image *icons\[\], int count\)](#).

11.164.3.38 iconize()

```
void Fl_Window::iconize ()
```

Iconifies the window.

If you call this when [shown\(\)](#) is false it will [show\(\)](#) it as an icon. If the window is already iconified this does nothing. Call [show\(\)](#) to restore the window.

When a window is iconified/restored (either by these calls or by the user) the [handle\(\)](#) method is called with FL_HIDE and FL_SHOW events and [visible\(\)](#) is turned on and off.

There is no way to control what is drawn in the icon except with the string passed to [Fl_Window::xclass\(\)](#). You should not rely on window managers displaying the icons.

11.164.3.39 icons() [1/2]

```
void Fl_Window::icons (
    const Fl_RGB_Image * icons[],
    int count)
```

Sets the window icons.

You may set multiple window icons with different sizes. Dependent on the platform and system settings the best (or the first) icon will be chosen.

The given images in `icons` are copied. You can use a local variable or free the images immediately after this call. If `count` is zero, current icons are removed. If `count` is greater than zero (must not be negative), then `icons[]` must contain at least `count` valid image pointers (not NULL). Otherwise the behavior is undefined.

Parameters

in	<i>icons</i>	icons for this window
in	<i>count</i>	number of images in <code>icons</code> . Set to 0 to remove the current icons

See also

[Fl_Window::default_icon\(const Fl_RGB_Image *\)](#)
[Fl_Window::default_icons\(const Fl_RGB_Image *\[\], int\)](#)
[Fl_Window::icon\(const Fl_RGB_Image *\)](#)

Note

See [Window icons](#) for the Wayland platform.

11.164.3.40 icons() [2/2]

```
void Fl_Window::icons (
    HICON big_icon,
    HICON small_icon)
```

Sets the window icons using HICON handles (Windows platform only).

The given icons are copied. You can free the icons immediately after this call.

Parameters

in	<i>big_icon</i>	large window icon
in	<i>small_icon</i>	small window icon

11.164.3.41 is_a_rescale()

```
bool Fl_Window::is_a_rescale () [static]
```

Returns true when a window is being rescaled.

Since

1.4.0

11.164.3.42 is_resizable()

```
int Fl_Window::is_resizable () [protected]
```

Protected method to determine whether a window is resizable.

If [size_range\(\)](#) has not yet been called this method calculates the default size range values by calling [default_size_range\(\)](#).

This method is for internal use only. The returned value is a bit mask and non-zero if the window is resizable in at least one direction.

Returns

non-zero if the window is resizable

Return values

0	the window is not resizable
1	the window is resizable in horizontal direction (w)
2	the window is resizable in vertical direction (h)
3	the window is resizable in both directions (w and h)

See also

[default_size_range\(\)](#)

Since

1.4.0

11.164.3.43 make_current()

```
void Fl_Window::make_current ()
```

Sets things up so that the drawing functions in [<FL/fl_draw.H>](#) will go into this window.

This is useful for incremental update of windows, such as in an idle callback, which will make your program behave much better if it draws a slow graphic. **Danger: incremental update is very hard to debug and maintain!**

This method only works for the [Fl_Window](#) and [Fl_Gl_Window](#) derived classes.

11.164.3.44 maximize()

```
void Fl_Window::maximize ()
```

Maximizes a top-level window to its current screen.

This function is effective only with a [show\(\)](#)'n, resizable, top-level window. Bordered and borderless windows can be used. Fullscreen windows can't be used.

See also

[Fl_Window::un_maximize\(\)](#), [Fl_Window::maximize_active\(\)](#)

Since

1.4.0

11.164.3.45 modal()

```
unsigned int Fl_Window::modal () const [inline]
```

Returns true if this window is modal.

11.164.3.46 os_id()

```
fl_uintptr_t Fl_Window::os_id ()
```

Returns a platform-specific identification of a shown window, or 0 if not shown.

Note

This identification may differ from the platform-specific reference of an [Fl_Window](#) object used by functions [fl_x11_xid\(\)](#), [fl_mac_xid\(\)](#), [fl_x11_find\(\)](#), and [fl_mac_find\(\)](#).

- X11 platform: the window's XID.
- macOS platform: The window number of the window's window device.
- other platforms: 0.

Since

1.4.0

11.164.3.47 resize()

```
void Fl_Window::resize (
    int X,
    int Y,
    int W,
    int H) [virtual]
```

Changes the size and position of the window.

If [shown\(\)](#) is true, these changes are communicated to the window server (which may refuse that size and cause a further resize). If [shown\(\)](#) is false, the size and position are used when [show\(\)](#) is called. See [Fl_Group](#) for the effect of resizing on the child widgets.

You can also call the [Fl_Widget](#) methods [size\(x,y\)](#) and [position\(w,h\)](#), which are inline wrappers for this virtual function.

A top-level window can not force, but merely suggest a position and size to the operating system. The window manager may not be willing or able to display a window at the desired position or with the given dimensions. It is up to the application developer to verify window parameters after the resize request.

Reimplemented from [Fl_Group](#).

11.164.3.48 screen_num() [1/2]

```
int Fl_Window::screen_num ()
```

The number of the screen containing the mapped window.

This method returns the screen number (0 .. n) of the window if it is shown, otherwise the result value is undefined.

Note

The return value is undefined before the window is shown and if the window has been hidden after it was previously shown.

To position a window on a particular screen, use [Fl_Window::screen_num\(int\)](#) **before** you [show\(\)](#) it.

Returns

screen number of the window if it is shown

See also

[Fl_Window::screen_num\(int\)](#)

Since

1.4.0

11.164.3.49 screen_num() [2/2]

```
void Fl_Window::screen_num (
    int screen_num)
```

Set the number of the screen where to map the window.

Call this and set also the window's desired x/y position before [show\(\)](#)ing the window. This can be necessary if a system has several screens with distinct scaling factors because the window's [x\(\)](#) and [y\(\)](#) may not suffice to uniquely identify one screen.

Consider a system with two screens where the left screen is A pixels wide and has a scaling factor of 1 whereas the right screen has a scaling factor of 2. For the sake of simplicity, consider only the X coordinate of the window. FLTK coordinates translate directly to pixel coordinates on the left screen, whereas FLTK coordinates multiplied by 2 correspond to pixel coordinates on the right screen. Consequently, FLTK coordinates between A/2 + 1 and A-1 can map to both screens. Therefore both the window coordinates and the screen number are necessary to uniquely identify where a window is to be mapped.

The valid range of `screen_num` is 0 .. [Fl::screen_count\(\)](#) - 1.

This method does nothing if

- the window is already [shown\(\)](#) or
- the given screen number is out of range.

Parameters

<code>in</code>	<code>screen_num</code>	screen number where the window is to be mapped

See also

[Fl_Window::screen_num\(\)](#)

[Fl::screen_count\(\)](#)

Since

1.4.0

11.164.3.50 set_menu_window()

```
void Fl_Window::set_menu_window () [inline]
```

Marks the window as a menu window.

This is intended for internal use, but it can also be used if you write your own menu handling. However, this is not recommended.

This flag is used for correct "parenting" of windows in communication with the windowing system. Modern X window managers can use different flags to distinguish menu and tooltip windows from normal windows.

This must be called before the window is shown and cannot be changed later.

11.164.3.51 set_modal()

```
void Fl_Window::set_modal () [inline]
```

A "modal" window, when [shown\(\)](#), will prevent any events from being delivered to other windows in the same program, and will also remain on top of the other windows (if the X window manager supports the "transient for" property).

Several modal windows may be shown at once, in which case only the last one shown gets events. You can see which window (if any) is modal by calling [Fl::modal\(\)](#).

11.164.3.52 set_non_modal()

```
void Fl_Window::set_non_modal () [inline]
```

A "non-modal" window (terminology borrowed from Microsoft Windows) acts like a [modal\(\)](#) one in that it remains on top, but it has no effect on event delivery.

There are *three* states for a window: modal, non-modal, and normal.

11.164.3.53 set_tooltip_window()

```
void Fl_Window::set_tooltip_window () [inline]
```

Marks the window as a tooltip window.

This is intended for internal use, but it can also be used if you write your own tooltip handling. However, this is not recommended.

This flag is used for correct "parenting" of windows in communication with the windowing system. Modern X window managers can use different flags to distinguish menu and tooltip windows from normal windows.

This must be called before the window is shown and cannot be changed later.

Note

Since `Fl_Tooltip_Window` is derived from `Fl_Menu_Window`, this also **clears** the `menu_window()` state.

11.164.3.54 shape() [1/3]

```
const Fl_Image * Fl_Window::shape ()
```

Returns the image controlling the window shape or NULL.

Since

1.4.0

11.164.3.55 shape() [2/3]

```
void Fl_Window::shape (
    const Fl_Image & img)
```

Set the window's shape with an `Fl_Image`.

See also

void `shape(const Fl_Image* img)`

11.164.3.56 shape() [3/3]

```
void Fl_Window::shape (
    const Fl_Image * img)
```

Assigns a non-rectangular shape to the window.

This function gives an arbitrary shape (not just a rectangular region) to an `Fl_Window`. An `Fl_Image` of any dimension can be used as mask; it is rescaled to the window's dimension as needed.

The layout and widgets inside are unaware of the mask shape, and most will act as though the window's rectangular bounding box is available to them. It is up to you to make sure they adhere to the bounds of their masking shape.

The `img` argument can be an `Fl_Bitmap`, `Fl_Pixmap`, `Fl_RGB_Image` or `Fl_Shared_Image`:

- With `Fl_Bitmap` or `Fl_Pixmap`, the shaped window covers the image part where bitmap bits equal one, or where the pixmap is not fully transparent.
- With an `Fl_RGB_Image` with an alpha channel (depths 2 or 4), the shaped window covers the image part that is not fully transparent.
- With an `Fl_RGB_Image` of depth 1 (gray-scale) or 3 (RGB), the shaped window covers the non-black image part.
- With an `Fl_Shared_Image`, the shape is determined by rules above applied to the underlying image. The shared image should not have been scaled through `Fl_Image::scale()`.

Platform details:

- On the unix/linux platform, the SHAPE extension of the X server is required. This function does control the shape of `Fl_Gl_Window` instances.
- On the Windows platform, this function does nothing with class `Fl_Gl_Window`.

- On the Mac platform, OS version 10.4 or above is required. An 8-bit shape-mask is used when `img` is an [Fl_RGB_Image](#): with depths 2 or 4, the image alpha channel becomes the shape mask such that areas with alpha = 0 are out of the shaped window; with depths 1 or 3, white and black are in and out of the shaped window, respectively, and other colors give intermediate masking scores. This function does nothing with class [Fl_Gl_Window](#).

The window borders and caption created by the window system are turned off by default. They can be re-enabled by calling `Fl_Window::border(1)`.

A usage example is found at `example/shapedwindow.cxx`.

Version

1.3.3

11.164.3.57 `show()` [1/2]

```
void Fl_Window::show () [virtual]
```

Puts the window on the screen.

This has the side effect of opening the display, if not done before.

If the window is already shown then it is restored and raised to the top. This is really convenient because your program can call `show()` at any time, even if the window is already up. It also means that `show()` serves the purpose of `raise()` in other toolkits.

[Fl_Window::show\(int argc, char **argv\)](#) is used for top-level windows and allows standard arguments to be parsed from the command-line.

Note

For some obscure reasons [Fl_Window::show\(\)](#) resets the current group by calling `Fl_Group::current(0)`. The comments in the code say "get rid of very common user bug: forgot end()". Although this is true it may have unwanted side effects if you `show()` an unrelated window (maybe for an error message or warning) while building a window or any other group widget.

Todo Check if we can remove resetting the current group in a later FLTK version (after 1.3.x). This may break "already broken" programs though if they rely on this "feature".

See also

[Fl_Window::show\(int argc, char **argv\)](#)

Reimplemented from [Fl_Widget](#).

11.164.3.58 `show()` [2/2]

```
void Fl_Window::show (
    int argc,
    char ** argv)
```

Puts the window on the screen with `show()` and parses command-line arguments.

This call should be used for top-level windows, at least for the first (main) window. It allows standard arguments to be parsed, as done by [Fl::args\(int, char **\)](#), from the command-line. You can use `argc` and `argv` from `main(int argc, char **argv)` for this call.

This call also sets up some system-specific internal variables, that is, it sets `FL_SELECTION_COLOR` and calls [Fl::background\(\)](#), [Fl::background2\(\)](#), [Fl::foreground\(\)](#) with default or X resources-given values, and calls [Fl::scheme\(const char *\)](#) for the current scheme. On X11, it also calls [Fl::dnd_text_ops\(int\)](#), [Fl_Tooltip::enable\(int\)](#), [Fl::visible_focus\(int\)](#) with X resources-given values.

Parameters

<code>argc</code>	command-line argument count, usually from <code>main()</code>
<code>argv</code>	command-line argument vector, usually from <code>main()</code>

See also

```
virtual void Fl_Window::show()
Fl::args(int, char **)
```

11.164.3.59 show_next_window_iconic() [1/2]

```
static char Fl_Window::show_next_window_iconic () [inline], [static]
```

Returns the static flag whether the next window should be opened iconified.

Note

This is an **internal function**, you should not use this in user code.

Please use [Fl_Window::iconize\(\)](#) to iconify a window.

11.164.3.60 show_next_window_iconic() [2/2]

```
static void Fl_Window::show_next_window_iconic (
    char stat) [inline], [static]
```

Sets a static flag whether the next window should be opened iconified.

Note

This is an **internal function**, you should not use this in user code.

Please use [Fl_Window::iconize\(\)](#) instead.

11.164.3.61 shown()

```
int Fl_Window::shown () [inline]
```

Returns non-zero if [show\(\)](#) has been called (but not [hide\(\)](#)).

You can tell if a window is iconified with (w->[shown\(\)](#) && !w->[visible\(\)](#)).

11.164.3.62 size_range()

```
void Fl_Window::size_range (
    int minWidth,
    int minHeight,
    int maxWidth = 0,
    int maxHeight = 0,
    int deltaX = 0,
    int deltaY = 0,
    int aspectRatio = 0)
```

Sets the allowable range to which the user can resize this window.

We recommend to call [size_range\(\)](#) if you have a [resizable\(\)](#) widget in a main window, and to call it after setting the [resizable\(\)](#) and before [show\(\)](#)'ing the window for best cross platform compatibility.

If this function is **not** called, FLTK tries to figure out the range when the window is shown. Please see the protected method [default_size_range\(\)](#) for details.

It is undefined what happens if the current window size does not fit in the constraints passed to [size_range\(\)](#).

Note

This only works for top-level windows and the exact behavior can be platform specific. To work correctly across all platforms [size_range\(\)](#) must be called after setting the [resizable\(\)](#) widget of the window and before the window is [show\(\)](#)'n.

Calling [size_range\(\)](#) after the window has been shown may work on some but not all platforms. If you need to change the [size_range\(\)](#) after the window has been shown, then you should consider to [hide\(\)](#) and [show\(\)](#) the window again, i.e. call [hide\(\)](#), [size_range\(\)](#), and [show\(\)](#) in this order.

Typical usage: call

```
size_range(minWidth, minHeight);
```

after setting the resizable widget and before calling [show\(\)](#). This ensures that the window cannot be resized smaller than the given values by user interaction.

`maxWidth` and `maxHeight` might be useful in some special cases but less frequently used.

The other optional parameters `deltaX`, `deltaY`, and `aspectRatio` are not recommended because they may not work on all platforms and may even under X11 not be supported by all Window Managers.

Parameters

in	<i>minWidth,minHeight</i>	The smallest the window can be. Either value must be greater than 0.
in	<i>maxWidth,maxHeight</i>	The largest the window can be. If either is equal to the minimum then you cannot resize in that direction. If either is zero then FLTK picks a maximum size in that direction such that the window will fill the screen.
in	<i>deltaX,deltaY</i>	These are size increments. The window will be constrained to widths of <code>minWidth + N * deltaX</code> , where N is any non-negative integer. If these are less or equal to 1 they are ignored (this is always ignored on Windows).
in	<i>aspectRatio</i>	A flag that indicates that the window should preserve its aspect ratio. This only works if both the maximum and minimum have the same aspect ratio (ignored on Windows and by many X window managers).

11.164.3.63 `un_maximize()`

```
void Fl_Window::un_maximize ()
```

Returns a previously maximized top-level window to its previous size.

See also

[Fl_Window::maximize\(\)](#)

Since

1.4.0

11.164.3.64 `wait_for_expose()`

```
void Fl_Window::wait_for_expose ()
```

Waits for the window to be displayed after calling [show\(\)](#).

[Fl_Window::show\(\)](#) is not guaranteed to show and draw the window on all platforms immediately. Instead this is done in the background; particularly on X11 it will take a few messages (client server roundtrips) to display the window. Usually this small delay doesn't matter, but in some cases you may want to have the window instantiated and displayed synchronously.

Currently (as of FLTK 1.3.4) this method has an effect on X11 and Mac OS. On Windows, [show\(\)](#) is always synchronous. The effect of [show\(\)](#) varies with versions of Mac OS X: early versions have the window appear on the screen when [show\(\)](#) returns, later versions don't. If you want to write portable code and need this synchronous [show\(\)](#) feature, add `win->wait_for_expose()` on all platforms, and FLTK will just do the right thing.

This method can be used for displaying splash screens before calling [Fl::run\(\)](#) or for having exact control over which window has the focus after calling [show\(\)](#).

If the window is not [shown\(\)](#), this method does nothing.

Note

Depending on the platform and window manager [wait_for_expose\(\)](#) may not guarantee that the window is fully drawn when it is called. Under X11 it may only make sure that the window is **mapped**, i.e. the internal (OS dependent) window object was created (and maybe shown on the desktop as an empty frame or something like that). You may need to call [Fl::flush\(\)](#) after [wait_for_expose\(\)](#) to make sure the window and all its widgets are drawn and thus visible.

FLTK does the best it can do to make sure that all widgets get drawn if you call [wait_for_expose\(\)](#) and [Fl::flush\(\)](#). However, dependent on the window manager it can not be guaranteed that this does always happen synchronously. The only guaranteed behavior that all widgets are eventually drawn is if the FLTK event loop is run continuously, for instance with [Fl::run\(\)](#).

See also

virtual void [Fl_Window::show\(\)](#)

Example code for displaying a window before calling [Fl::run\(\)](#)

```
Fl_Double_Window win = new Fl_Double_Window(...);

// do more window initialization here ...

win->show();           // show window
win->wait_for_expose(); // wait, until displayed
Fl::flush();           // make sure everything gets drawn

// do more initialization work that needs some time here ...

Fl::run();              // start FLTK event loop
```

Note that the window will not be responsive until the event loop is started with [Fl::run\(\)](#).

11.164.3.65 xclass() [1/2]

const char * [Fl_Window::xclass](#) () const
Returns the xclass for this window, or a default.

See also

[Fl_Window::default_xclass\(const char *\)](#)
[Fl_Window::xclass\(const char *\)](#)

11.164.3.66 xclass() [2/2]

void [Fl_Window::xclass](#) (
 const char * xc)

Sets the xclass for this window.

A string used to tell the system what type of window this is. Mostly this identifies the picture to draw in the icon. This only works if called *before* calling [show\(\)](#).

Under X, this is turned into a XA_WM_CLASS pair by truncating at the first non-alphanumeric character and capitalizing the first character, and the second one if the first is 'x'. Thus "foo" turns into "foo, Foo", and "xprog.1" turns into "xprog, XProg".

Under Microsoft Windows, this string is used as the name of the WNDCLASS structure, though it is not clear if this can have any visible effect.

Since

FLTK 1.3 the passed string is copied. You can use a local variable or free the string immediately after this call. Note that FLTK 1.1 stores the *pointer* without copying the string.

If the default xclass has not yet been set, this also sets the default xclass for all windows created subsequently.

See also

[Fl_Window::default_xclass\(const char *\)](#)

11.164.4 Member Data Documentation

11.164.4.1 current_

[Fl_Window](#) * [Fl_Window::current_](#) [static], [protected]

Stores the last window that was made current.

See [current\(\)](#) const

The documentation for this class was generated from the following files:

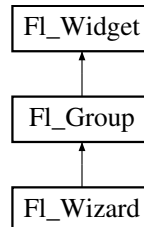
- [Fl_Window.H](#)
- [Fl_arg.cxx](#)
- [fl_cursor.cxx](#)
- [Fl_Window.cxx](#)
- [Fl_Window_fullscreen.cxx](#)
- [Fl_Window_hotspot.cxx](#)
- [Fl_Window_iconize.cxx](#)

11.165 FL_Wizard Class Reference

This widget is based off the [FL_Tabs](#) widget, but instead of displaying tabs it only changes "tabs" under program control.

```
#include <FL_Wizard.H>
```

Inheritance diagram for FL_Wizard:



Public Member Functions

- [FL_Wizard](#) (int, int, int, int, const char *=0)
The constructor creates the [FL_Wizard](#) widget at the specified position and size.
- void [next](#) ()
This method shows the next child of the wizard.
- void [prev](#) ()
Shows the previous child.
- [FL_Widget *](#) [value](#) ()
Gets the current visible child widget.
- void [value](#) ([FL_Widget *](#))
Sets the child widget that is visible.

Public Member Functions inherited from [FL_Group](#)

- [FL_Widget *](#) [_ddfdesign_kludge](#) ()
This is for forms compatibility only.
- void [add](#) ([FL_Widget &](#))
The widget is removed from its current group (if any) and then added to the end of this group.
- void [add](#) ([FL_Widget *o](#))
See void [FL_Group::add\(FL_Widget &w\)](#)
- void [add_resizable](#) ([FL_Widget &o](#))
Adds a widget to the group and makes it the resizable widget.
- [FL_Widget *](#) [const * array](#) () const
Returns a pointer to the internal array of children.
- [FL_Group](#) const * [as_group](#) () const [FL_OVERRIDE](#)
- [FL_Group *](#) [as_group](#) () [FL_OVERRIDE](#)
Returns an [FL_Group](#) pointer if this widget is an [FL_Group](#).
- void [begin](#) ()
Sets the current group so you can build the widget tree by just constructing the widgets.
- [FL_Widget *](#) [child](#) (int n) const
Returns the n'th child.
- int [children](#) () const
Returns how many child widgets the group has.
- void [clear](#) ()
Deletes all child widgets from memory recursively.
- unsigned int [clip_children](#) ()

- Returns the current clipping mode.*
- void `clip_children` (int c)
 - Controls whether the group widget clips the drawing of child widgets to its bounding box.*
- virtual int `delete_child` (int n)
 - Removes the widget at `index` from the group and deletes it.*
- void `end` ()
 - Exactly the same as `current(this->parent())`.*
- int `find` (const `FL_Widget` &o) const
 - See int `FL_Group::find(const FL_Widget *w)` const.*
- int `find` (const `FL_Widget` *) const
 - Searches the children for the widget and returns the index.*
- `FL_Group` (int, int, int, int, const char *s=0)
 - Creates a new `FL_Group` widget using the given position, size, and label string.*
- void `focus` (`FL_Widget` *W)
- void `forms_end` ()
 - This is for forms compatibility only.*
- int `handle` (int) `FL_OVERRIDE`
 - Handles the specified event.*
- void `init_sizes` ()
 - Resets the internal array of widget sizes and positions.*
- void `insert` (`FL_Widget` &, int i)
 - The widget is removed from its current group (if any) and then inserted into this group.*
- void `insert` (`FL_Widget` &o, `FL_Widget` *before)
 - This does `insert(w, find(before))`.*
- void `remove` (`FL_Widget` &)
 - Removes a widget from the group but does not delete it.*
- void `remove` (`FL_Widget` *o)
 - Removes the widget o from the group.*
- void `remove` (int index)
 - Removes the widget at `index` from the group but does not delete it.*
- `FL_Widget` * `resizable` () const
 - Returns the group's resizable widget.*
- void `resizable` (`FL_Widget` &o)
 - Sets the group's resizable widget.*
- void `resizable` (`FL_Widget` *o)
 - The resizable widget defines both the resizing box and the resizing behavior of the group and its children.*
- void `resize` (int, int, int, int) `FL_OVERRIDE`
 - Resizes the `FL_Group` widget and all of its children.*
- virtual `~FL_Group` ()
 - The destructor also deletes all the children.*

Public Member Functions inherited from `FL_Widget`

- void `_clear_fullscreen` ()
- void `_set_fullscreen` ()
- void `activate` ()
 - Activates the widget.*
- unsigned int `active` () const
 - Returns whether the widget is active.*
- int `active_r` () const
 - Returns whether the widget and all of its parents are active.*

- [Fl_Align align](#) () const
Gets the label alignment.
- void [align](#) ([Fl_Align](#) alignment)
Sets the label alignment.
- long [argument](#) () const
Gets the current user data (long) argument that is passed to the callback function.
- void [argument](#) (long v)
Sets the current user data (long) argument that is passed to the callback function.
- virtual class [Fl_Gl_Window](#) * [as_gl_window](#) ()
Returns an [Fl_Gl_Window](#) pointer if this widget is an [Fl_Gl_Window](#).
- virtual class [Fl_Gl_Window](#) const * [as_gl_window](#) () const
- virtual [Fl_Window](#) * [as_window](#) ()
Returns an [Fl_Window](#) pointer if this widget is an [Fl_Window](#).
- virtual [Fl_Window](#) const * [as_window](#) () const
- void [bind_deimage](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the inactive state.
- void [bind_deimage](#) (int f)
Bind the inactive image to the widget, so the widget will delete the image when it is no longer needed.
- void [bind_image](#) ([Fl_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- void [bind_image](#) (int f)
Bind the image to the widget, so the widget will delete the image when it is no longer needed.
- [Fl_Boxtype box](#) () const
Gets the box type of the widget.
- void [box](#) ([Fl_Boxtype](#) new_box)
Sets the box type for the widget.
- [Fl_Callback_p callback](#) () const
Gets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, [Fl_Callback_User_Data](#) *p, bool auto_free)
Sets the current callback function and managed user data for the widget.
- void [callback](#) ([Fl_Callback](#) *cb, void *p)
Sets the current callback function and data for the widget.
- void [callback](#) ([Fl_Callback0](#) *cb)
Sets the current callback function for the widget.
- void [callback](#) ([Fl_Callback1](#) *cb, long p=0)
Sets the current callback function for the widget.
- unsigned int [changed](#) () const
Checks if the widget value changed since the last callback.
- void [clear_active](#) ()
Marks the widget as inactive without sending events or changing focus.
- void [clear_changed](#) ()
Marks the value of the widget as unchanged.
- void [clear_damage](#) (uchar c=0)
Clears or sets the damage flags.
- void [clear_output](#) ()
Sets a widget to accept input.
- void [clear_visible](#) ()
Hides the widget.
- void [clear_visible_focus](#) ()

- Disables keyboard focus navigation with this widget.*

 - `FL_Color color () const`

Gets the background color of the widget.

- `void color (FL_Color bg)`

Sets the background color of the widget.

- `void color (FL_Color bg, FL_Color sel)`

Sets the background and selection color of the widget.

- `FL_Color color2 () const`

For back compatibility only.

- `void color2 (unsigned a)`

For back compatibility only.

- `int contains (const FL_Widget *w) const`

Checks if w is a child of this widget.

- `void copy_label (const char *new_label)`

Sets the current label.

- `void copy_tooltip (const char *text)`

Sets the current tooltip text.

- `uchar damage () const`

Returns non-zero if draw() needs to be called.

- `void damage (uchar c)`

Sets the damage bits for the widget.

- `void damage (uchar c, int x, int y, int w, int h)`

Sets the damage bits for an area inside the widget.

- `int damage_resize (int, int, int, int)`

Internal use only.

- `void deactivate ()`

Deactivates the widget.

- `FL_Image * deimage ()`

Gets the image that is used as part of the widget label when in the inactive state.

- `const FL_Image * deimage () const`

Gets the image that is used as part of the widget label when in the inactive state.

- `void deimage (FL_Image &img)`

Sets the image to use as part of the widget label when in the inactive state.

- `void deimage (FL_Image *img)`

Sets the image to use as part of the widget label when in the inactive state.

- `int deimage_bound () const`

Returns whether the inactive image is managed by the widget.

- `void do_callback (FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with default arguments.

- `void do_callback (FL_Widget *widget, long arg, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.

- `void do_callback (FL_Widget *widget, void *arg=0, FL_Callback_Reason reason=FL_REASON_UNKNOWN)`

Calls the widget callback function with arbitrary arguments.

- `void draw_label (int, int, int, int, FL_Align) const`

Draws the label in an arbitrary bounding box with an arbitrary alignment.

- `int h () const`

Gets the widget height.

- `virtual void hide ()`

Makes a widget invisible.

- `int horizontal_label_margin ()`

Get the spacing between the label and the horizontal edge of the widget.

- void [horizontal_label_margin](#) (int px)
Set the spacing between the label and the horizontal edge of the widget.
- [FL_Image](#) * [image](#) ()
Gets the image that is used as part of the widget label when in the active state.
- const [FL_Image](#) * [image](#) () const
Gets the image that is used as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) &img)
Sets the image to use as part of the widget label when in the active state.
- void [image](#) ([FL_Image](#) *img)
Sets the image to use as part of the widget label when in the active state.
- int [image_bound](#) () const
Returns whether the image is managed by the widget.
- int [inside](#) (const [FL_Widget](#) *wgt) const
Checks if this widget is a child of wgt.
- int [is_label_copied](#) () const
Returns whether the current label was assigned with [copy_label\(\)](#).
- const char * [label](#) () const
Gets the current label text.
- void [label](#) (const char *text)
Sets the current label pointer.
- void [label](#) ([FL_Labeltype](#) a, const char *b)
Shortcut to set the label text and type in one call.
- int [label_image_spacing](#) ()
Return the gap size between the label and the image.
- void [label_image_spacing](#) (int gap)
Set the gap between the label and the image in pixels.
- [FL_Color](#) [labelcolor](#) () const
Gets the label color.
- void [labelcolor](#) ([FL_Color](#) c)
Sets the label color.
- [FL_Font](#) [labelfont](#) () const
Gets the font to use.
- void [labelfont](#) ([FL_Font](#) f)
Sets the font to use.
- [FL_Fonsize](#) [labelsize](#) () const
Gets the font size in pixels.
- void [labelsize](#) ([FL_Fonsize](#) pix)
Sets the font size in pixels.
- [FL_Labeltype](#) [labeltype](#) () const
Gets the label type.
- void [labeltype](#) ([FL_Labeltype](#) a)
Sets the label type.
- void [measure_label](#) (int &ww, int &hh) const
Sets width ww and height hh accordingly with the label size.
- bool [needs_keyboard](#) () const
Returns whether this widget needs a keyboard.
- void [needs_keyboard](#) (bool needs)
Sets whether this widget needs a keyboard.
- unsigned int [output](#) () const
Returns if a widget is used for output only.
- [FL_Group](#) * [parent](#) () const

- Returns a pointer to the parent widget.*

 - void [parent](#) ([Fl_Group](#) *p)

Internal use only - "for hacks only".
- void [position](#) (int X, int Y)

Repositions the window or widget.
- void [redraw](#) ()

Schedules the drawing of the widget.
- void [redraw_label](#) ()

Schedules the drawing of the label.
- [Fl_Color](#) [selection_color](#) () const

Gets the selection color.
- void [selection_color](#) ([Fl_Color](#) a)

Sets the selection color.
- void [set_active](#) ()

Marks the widget as active without sending events or changing focus.
- void [set_changed](#) ()

Marks the value of the widget as changed.
- void [set_output](#) ()

Sets a widget to output only.
- void [set_visible](#) ()

Makes the widget visible.
- void [set_visible_focus](#) ()

Enables keyboard focus navigation with this widget.
- int [shortcut_label](#) () const

Returns whether the widget's label uses '&' to indicate shortcuts.
- void [shortcut_label](#) (int value)

Sets whether the widget's label uses '&' to indicate shortcuts.
- virtual void [show](#) ()

Makes a widget visible.
- void [size](#) (int W, int H)

Changes the size of the widget.
- int [take_focus](#) ()

Gives the widget the keyboard focus.
- unsigned int [takeevents](#) () const

Returns if the widget is able to take events.
- int [test_shortcut](#) ()

Returns true if the widget's label contains the entered '&x' shortcut.
- const char * [tooltip](#) () const

Gets the current tooltip text.
- void [tooltip](#) (const char *text)

Sets the current tooltip text.
- [Fl_Window](#) * [top_window](#) () const

Returns a pointer to the top-level window for the widget.
- [Fl_Window](#) * [top_window_offset](#) (int &xoff, int &yoff) const

Finds the x/y offset of the current widget relative to the top-level window.
- [uchar](#) [type](#) () const

Gets the widget type.
- void [type](#) ([uchar](#) t)

Sets the widget type.
- int [use_accents_menu](#) ()

Returns non zero if MAC_USE_ACCENTS_MENU flag is set, 0 otherwise.

- void * [user_data](#) () const
Gets the user data for this widget.
- void **user_data** ([Fl_Callback_User_Data](#) *v, bool auto_free)
Sets the user data for this widget.
- void **user_data** (void *v)
Sets the user data for this widget.
- int [vertical_label_margin](#) ()
Get the spacing between the label and the vertical edge of the widget.
- void [vertical_label_margin](#) (int px)
Set the spacing between the label and the vertical edge of the widget.
- unsigned int [visible](#) () const
Returns whether a widget is visible.
- unsigned int [visible_focus](#) () const
Checks whether this widget has a visible focus.
- void [visible_focus](#) (int v)
Modifies keyboard focus navigation.
- int [visible_r](#) () const
Returns whether a widget and all its parents are visible.
- int [w](#) () const
Gets the widget width.
- [Fl_When](#) [when](#) () const
Returns the conditions under which the callback is called.
- void [when](#) (uchar i)
Sets the flags used to decide when a callback is called.
- [Fl_Window](#) * [window](#) () const
Returns a pointer to the nearest parent window up the widget hierarchy.
- int [x](#) () const
Gets the widget position in its window.
- int [y](#) () const
Gets the widget position in its window.
- virtual ~[Fl_Widget](#) ()
Destroys the widget.

Protected Member Functions

- void [draw](#) () [FL_OVERRIDE](#)
Draws the wizard border and visible child.

Protected Member Functions inherited from [Fl_Group](#)

- [Fl_Rect](#) * [bounds](#) ()
Returns the internal array of widget sizes and positions.
- void [draw_child](#) ([Fl_Widget](#) &widget) const
Forces a child to redraw.
- void [draw_children](#) ()
Draws all children of the group.
- void **draw_outside_label** (const [Fl_Widget](#) &widget) const
Parents normally call this to draw outside labels of child widgets.
- virtual int [on_insert](#) ([Fl_Widget](#) *, int)
Allow derived groups to act when a widget is added as a child.
- virtual int [on_move](#) (int, int)
Allow derived groups to act when a widget is moved within the group.

- virtual void **on_remove** (int)
Allow derived groups to act when a child widget is removed from the group.
- int * **sizes** ()
Returns the internal array of widget sizes and positions.
- void **update_child** (FI_Widget &widget) const
Draws a child only if it needs it.

Protected Member Functions inherited from FI_Widget

- void **clear_flag** (unsigned int c)
Clears a flag in the flags mask.
- void **draw_backdrop** () const
If FL_ALIGN_IMAGE_BACKDROP is set, the image or deimage will be drawn.
- void **draw_box** () const
Draws the widget box according its box style.
- void **draw_box** (FI_Boxtype t, FI_Color c) const
Draws a box of type t, of color c at the widget's position and size.
- void **draw_box** (FI_Boxtype t, int x, int y, int w, int h, FI_Color c) const
Draws a box of type t, of color c at the position X,Y and size W,H.
- void **draw_focus** () const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int X, int Y, int W, int H) const
Draws a focus rectangle around the widget.
- void **draw_focus** (FI_Boxtype t, int x, int y, int w, int h, FI_Color bg) const
Draws a focus box for the widget at the given position and size.
- void **draw_label** () const
Draws the widget's label at the defined label position.
- void **draw_label** (int, int, int, int) const
Draws the label in an arbitrary bounding box.
- FI_Widget (int x, int y, int w, int h, const char *label=0L)
Creates a widget at the given position and size.
- unsigned int **flags** () const
Gets the widget flags mask.
- void **h** (int v)
Internal use only.
- void **set_flag** (unsigned int c)
Sets a flag in the flags mask.
- void **w** (int v)
Internal use only.
- void **x** (int v)
Internal use only.
- void **y** (int v)
Internal use only.

Additional Inherited Members

Static Public Member Functions inherited from FI_Group

- static FI_Group * **current** ()
Returns the currently active group.
- static void **current** (FI_Group *g)
Sets the current group.

Static Public Member Functions inherited from [Fl_Widget](#)

- static void [default_callback](#) ([Fl_Widget](#) *widget, void *data)
The default callback for all widgets that don't set a callback.
- static unsigned int [label_shortcut](#) (const char *t)
Returns the Unicode value of the '&x' shortcut in a given text.
- static int [test_shortcut](#) (const char *, const bool require_alt=false)
Returns true if the given text t contains the entered '&x' shortcut.

Protected Types inherited from [Fl_Widget](#)

- enum {
[INACTIVE](#) = 1<<0 , [INVISIBLE](#) = 1<<1 , [OUTPUT](#) = 1<<2 , [NOBORDER](#) = 1<<3 ,
[FORCE_POSITION](#) = 1<<4 , [NON_MODAL](#) = 1<<5 , [SHORTCUT_LABEL](#) = 1<<6 , [CHANGED](#) = 1<<7
, [OVERRIDE](#) = 1<<8 , [VISIBLE_FOCUS](#) = 1<<9 , [COPIED_LABEL](#) = 1<<10 , [CLIP_CHILDREN](#) = 1<<11
, [MENU_WINDOW](#) = 1<<12 , [TOOLTIP_WINDOW](#) = 1<<13 , [MODAL](#) = 1<<14 , [NO_OVERLAY](#) = 1<<15
, [GROUP_RELATIVE](#) = 1<<16 , [COPIED_TOOLTIP](#) = 1<<17 , [FULLSCREEN](#) = 1<<18 , [MAC_USE_ACCENTS_MENU](#)
= 1<<19 ,
[NEEDS_KEYBOARD](#) = 1<<20 , [IMAGE_BOUND](#) = 1<<21 , [DEIMAGE_BOUND](#) = 1<<22 ,
[AUTO_DELETE_USER_DATA](#) = 1<<23 ,
[MAXIMIZED](#) = 1<<24 , [POPUP](#) = 1<<25 , [USERFLAG3](#) = 1<<29 , [USERFLAG2](#) = 1<<30 ,
[USERFLAG1](#) = 1<<31 }
flags possible values enumeration.

11.165.1 Detailed Description

This widget is based off the [Fl_Tabs](#) widget, but instead of displaying tabs it only changes "tabs" under program control.

Its primary purpose is to support "wizards" that step a user through configuration or troubleshooting tasks.

As with [Fl_Tabs](#), wizard panes are composed of child (usually [Fl_Group](#)) widgets. Navigation buttons must be added separately.

11.165.2 Constructor & Destructor Documentation

11.165.2.1 [Fl_Wizard\(\)](#)

```
Fl_Wizard::Fl_Wizard (
    int xx,
    int yy,
    int ww,
    int hh,
    const char * l = 0)
```

The constructor creates the [Fl_Wizard](#) widget at the specified position and size.

The inherited destructor destroys the widget and its children.

11.165.3 Member Function Documentation

11.165.3.1 [draw\(\)](#)

```
void Fl_Wizard::draw (
    void ) [protected], [virtual]
```

Draws the wizard border and visible child.

Reimplemented from [Fl_Group](#).

11.165.3.2 next()

```
void Fl_Wizard::next ()
```

This method shows the next child of the wizard.

If the last child is already visible, this function does nothing.

The documentation for this class was generated from the following files:

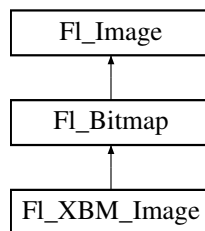
- Fl_Wizard.H
- Fl_Wizard.cxx

11.166 Fl_XBM_Image Class Reference

The [Fl_XBM_Image](#) class supports loading, caching, and drawing of X Bitmap (XBM) bitmap files.

```
#include <Fl_XBM_Image.H>
```

Inheritance diagram for Fl_XBM_Image:



Public Member Functions

- [Fl_XBM_Image](#) (const char *filename)
The constructor loads the named XBM file from the given name filename.

Public Member Functions inherited from [Fl_Bitmap](#)

- int **cache_h** ()
- int **cache_w** ()
- [Fl_Image](#) * **copy** () const
- [Fl_Image](#) * **copy** (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void **draw** (int X, int Y)
- void **draw** (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [Fl_Bitmap](#) (const char *bits, int bits_length, int W, int H)
The constructors create a new bitmap from the specified bitmap data.
- [Fl_Bitmap](#) (const char *bits, int W, int H)
The constructors create a new bitmap from the specified bitmap data.
- [Fl_Bitmap](#) (const uchar *bits, int bits_length, int W, int H)
The constructors create a new bitmap from the specified bitmap data.
- [Fl_Bitmap](#) (const uchar *bits, int W, int H)
The constructors create a new bitmap from the specified bitmap data.
- void **label** ([Fl_Menu_Item](#) *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.
- void **label** ([Fl_Widget](#) *w) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a widget or menu item.
- void **uncache** () [FL_OVERRIDE](#)
If the image has been cached for display, delete the cache data.
- virtual ~[Fl_Bitmap](#) ()
The destructor frees all memory and server resources that are used by the bitmap.

Public Member Functions inherited from [FI_Image](#)

- virtual class [FI_Shared_Image](#) * [as_shared_image](#) ()
Returns whether an image is an [FI_Shared_Image](#) or not.
- virtual void [color_average](#) ([FI_Color](#) c, float i)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [FI_Image](#) * [copy](#) () const
Creates a copy of the image in the same size.
- int [count](#) () const
Returns the number of data values associated with the image.
- int [d](#) () const
Returns the image depth.
- const char *const * [data](#) () const
Returns a pointer to the current image data array.
- int [data_h](#) () const
Returns the height of the image data.
- int [data_w](#) () const
Returns the width of the image data.
- virtual void [desaturate](#) ()
The [desaturate\(\)](#) method converts an image to grayscale.
- void [draw](#) (int X, int Y)
Draws the image to the current drawing surface.
- int [fail](#) () const
Returns a value that is not 0 if there is currently no image available.
- [FI_Image](#) (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int [h](#) () const
Returns the current image drawing height in FLTK units.
- void [inactive](#) ()
The [inactive\(\)](#) method calls [color_average\(FL_BACKGROUND_COLOR, 0.33f\)](#) to produce an image that appears grayed out.
- int [ld](#) () const
Returns the current line data size in bytes.
- virtual void [release](#) ()
Releases an [FI_Image](#) - the same as 'delete this'.
- virtual void [scale](#) (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- int [w](#) () const
Returns the current image drawing width in FLTK units.
- virtual ~[FI_Image](#) ()
The destructor is a virtual method that frees all memory used by the image.

Additional Inherited Members

Static Public Member Functions inherited from [FI_Image](#)

- static [FI_Labeltype](#) [define_FL_IMAGE_LABEL](#) ()
- static [FI_RGB_Scaling](#) [RGB_scaling](#) ()
Returns the currently used RGB image scaling method.
- static void [RGB_scaling](#) ([FI_RGB_Scaling](#))
Sets the RGB image scaling method used for copy(int, int).
- static [FI_RGB_Scaling](#) [scaling_algorithm](#) ()
Gets what algorithm is used when resizing a source image to draw it.
- static void [scaling_algorithm](#) ([FI_RGB_Scaling](#) algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Public Attributes inherited from FI_Bitmap

- int **alloc_array**
Non-zero if array points to bitmap data allocated internally.
- const uchar * **array**
pointer to raw bitmap data

Static Public Attributes inherited from FI_Image

- static const int **ERR_FILE_ACCESS** = -2
- static const int **ERR_FORMAT** = -3
- static const int **ERR_MEMORY_ACCESS** = -4
- static const int **ERR_NO_IMAGE** = -1
- static bool **register_images_done** = false
True after [fl_register_images\(\)](#) was called, false before.

Protected Member Functions inherited from FI_Image

- void **d** (int D)
Sets the current image depth.
- void **data** (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void **draw_empty** (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int **draw_scaled** (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void **h** (int H)
Sets the height of the image data.
- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Static Protected Member Functions inherited from FI_Image

- static void **labeltype** (const FI_Label *lo, int lx, int ly, int lw, int lh, FI_Align la)
- static void **measure** (const FI_Label *lo, int &lw, int &lh)

11.166.1 Detailed Description

The [FI_XBM_Image](#) class supports loading, caching, and drawing of X Bitmap (XBM) bitmap files.

11.166.2 Constructor & Destructor Documentation

11.166.2.1 FI_XBM_Image()

```
FI_XBM_Image::FI_XBM_Image (
    const char * name)
```

The constructor loads the named XBM file from the given name filename.

The destructor frees all memory and server resources that are used by the image.

The documentation for this class was generated from the following files:

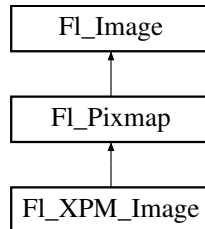
- FI_XBM_Image.H
- FI_XBM_Image.cxx

11.167 Fl_XPM_Image Class Reference

The [Fl_XPM_Image](#) class supports loading, caching, and drawing of X Pixmap (XPM) images, including transparency.

```
#include <Fl_XPM_Image.H>
```

Inheritance diagram for [Fl_XPM_Image](#):



Public Member Functions

- [Fl_XPM_Image](#) (const char *filename)
The constructor loads the XPM image from the name filename.

Public Member Functions inherited from [Fl_Pixmap](#)

- int **cache_h** ()
- int **cache_w** ()
- void [color_average](#) (Fl_Color c, float i) [FL_OVERRIDE](#)
The [color_average\(\)](#) method averages the colors in the image with the provided FLTK color value.
- [Fl_Image](#) * **copy** () const
- [Fl_Image](#) * [copy](#) (int W, int H) const [FL_OVERRIDE](#)
Creates a resized copy of the image.
- void [desaturate](#) () [FL_OVERRIDE](#)
The [desaturate\(\)](#) method converts an image to grayscale.
- void **draw** (int X, int Y)
- void [draw](#) (int X, int Y, int W, int H, int cx=0, int cy=0) [FL_OVERRIDE](#)
Draws the image to the current drawing surface with a bounding box.
- [Fl_Pixmap](#) (char *const *D)
The constructors create a new pixmap from the specified XPM data.
- [Fl_Pixmap](#) (const char *const *D)
The constructors create a new pixmap from the specified XPM data.
- [Fl_Pixmap](#) (const uchar *const *D)
The constructors create a new pixmap from the specified XPM data.
- [Fl_Pixmap](#) (uchar *const *D)
The constructors create a new pixmap from the specified XPM data.
- void [label](#) (Fl_Menu_Item *m) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a menu item.
- void [label](#) (Fl_Widget *w) [FL_OVERRIDE](#)
This method is an obsolete way to set the image attribute of a widget or menu item.
- void [uncache](#) () [FL_OVERRIDE](#)
If the image has been cached for display, delete the cache data.
- virtual ~**Fl_Pixmap** ()
The destructor frees all memory and server resources that are used by the pixmap.

Public Member Functions inherited from FI_Image

- virtual class FI_Shared_Image * **as_shared_image** ()
Returns whether an image is an FI_Shared_Image or not.
- FI_Image * **copy** () const
Creates a copy of the image in the same size.
- int **count** () const
Returns the number of data values associated with the image.
- int **d** () const
Returns the image depth.
- const char *const * **data** () const
Returns a pointer to the current image data array.
- int **data_h** () const
Returns the height of the image data.
- int **data_w** () const
Returns the width of the image data.
- void **draw** (int X, int Y)
Draws the image to the current drawing surface.
- int **fail** () const
Returns a value that is not 0 if there is currently no image available.
- FI_Image (int W, int H, int D)
The constructor creates an empty image with the specified width, height, and depth.
- int **h** () const
Returns the current image drawing height in FLTK units.
- void **inactive** ()
The inactive() method calls color_average(FL_BACKGROUND_COLOR, 0.33f) to produce an image that appears grayed out.
- int **ld** () const
Returns the current line data size in bytes.
- virtual void **release** ()
Releases an FI_Image - the same as 'delete this'.
- virtual void **scale** (int width, int height, int proportional=1, int can_expand=0)
Sets the drawing size of the image.
- int **w** () const
Returns the current image drawing width in FLTK units.
- virtual ~FI_Image ()
The destructor is a virtual method that frees all memory used by the image.

Additional Inherited Members

Static Public Member Functions inherited from FI_Image

- static FI_Labeltype **define_FL_IMAGE_LABEL** ()
- static FI_RGB_Scaling **RGB_scaling** ()
Returns the currently used RGB image scaling method.
- static void **RGB_scaling** (FI_RGB_Scaling)
Sets the RGB image scaling method used for copy(int, int).
- static FI_RGB_Scaling **scaling_algorithm** ()
Gets what algorithm is used when resizing a source image to draw it.
- static void **scaling_algorithm** (FI_RGB_Scaling algorithm)
Sets what algorithm is used when resizing a source image to draw it.

Public Attributes inherited from [FI_Pixmap](#)

- int **alloc_data**

Static Public Attributes inherited from [FI_Image](#)

- static const int **ERR_FILE_ACCESS** = -2
- static const int **ERR_FORMAT** = -3
- static const int **ERR_MEMORY_ACCESS** = -4
- static const int **ERR_NO_IMAGE** = -1
- static bool **register_images_done** = false
True after [fl_register_images\(\)](#) was called, false before.

Protected Member Functions inherited from [FI_Pixmap](#)

- void **measure** ()

Protected Member Functions inherited from [FI_Image](#)

- void **d** (int D)
Sets the current image depth.
- void **data** (const char *const *p, int c)
Sets the current data pointer and count of pointers in the array.
- void **draw_empty** (int X, int Y)
The protected method [draw_empty\(\)](#) draws a box with an X in it.
- int **draw_scaled** (int X, int Y, int W, int H)
Draw the image to the current drawing surface rescaled to a given width and height.
- void **h** (int H)
Sets the height of the image data.
- void **ld** (int LD)
Sets the current line data size in bytes.
- void **w** (int W)
Sets the width of the image data.

Static Protected Member Functions inherited from [FI_Image](#)

- static void **labeltype** (const [FI_Label](#) *lo, int lx, int ly, int lw, int lh, [FI_Align](#) la)
- static void **measure** (const [FI_Label](#) *lo, int &lw, int &lh)

11.167.1 Detailed Description

The [FI_XPM_Image](#) class supports loading, caching, and drawing of X Pixmap (XPM) images, including transparency.

11.167.2 Constructor & Destructor Documentation

11.167.2.1 [FI_XPM_Image](#)()

```
FI_XPM_Image::FI_XPM_Image (
    const char * name)
```

The constructor loads the XPM image from the name filename.

The destructor frees all memory and server resources that are used by the image.

The documentation for this class was generated from the following files:

- [FI_XPM_Image.H](#)
- [FI_XPM_Image.cxx](#)

11.168 FI_GIF_Image::GIF_FRAME Struct Reference

Classes

- struct [CPAL](#)

Public Member Functions

- void **colors** (int nclrs, int bg, int tp)
- void **disposal** (int mode, int time)
- **GIF_FRAME** (int frame, int W, int H, int fx, int fy, int fw, int fh, [uchar](#) *data)
- **GIF_FRAME** (int frame, [uchar](#) *data)

Public Attributes

- int **bkgd**
- const [uchar](#) * **bptr**
- int **clrs**
- const struct [FI_GIF_Image::GIF_FRAME::CPAL](#) * **cpal**
- int **delay**
- int **dispose**
- int **h**
- int **height**
- int **ifrm**
- int **trans**
- int **w**
- int **width**
- int **x**
- int **y**

The documentation for this struct was generated from the following file:

- FI_GIF_Image.H

11.169 FI_ICO_Image::IconDirEntry Struct Reference

Windows ICONDIRENTRY structure

```
#include <FI_ICO_Image.H>
```

Public Attributes

- int **bColorCount**
Number of colors (0 if 8bpp)
- int **bHeight**
Image height.
- int **bReserved**
Reserved.
- int **bWidth**
Image width.
- int **dwBytesInRes**
Resource size in bytes.
- int **dwImageOffset**
Offset to the image.
- int **wBitCount**
Bits per pixel.
- int **wPlanes**
Color Planes.

11.169.1 Detailed Description

Windows ICONDIRENTRY structure

The documentation for this struct was generated from the following file:

- [FI_ICO_Image.H](#)

11.170 FI_Text_Editor::Key_Binding Struct Reference

Simple linked list item associating a key/state to a function.

```
#include <Fl_Text_Editor.H>
```

Public Attributes

- [Key_Func](#) **function**
associated function
- int **key**
the key pressed
- [Key_Binding](#) * **next**
next key binding in the list
- int **state**
the state of key modifiers

11.170.1 Detailed Description

Simple linked list item associating a key/state to a function.

The documentation for this struct was generated from the following file:

- [FI_Text_Editor.H](#)

11.171 FI_Terminal::Margin Class Reference

Public Member Functions

- void **bottom** (int val)
- int **bottom** (void) const
- void **left** (int val)
- int **left** (void) const
- void **right** (int val)
- int **right** (void) const
- void **top** (int val)
- int **top** (void) const

The documentation for this class was generated from the following file:

- [FI_Terminal.H](#)

11.172 FI_Preferences::Name Class Reference

'[Name](#)' provides a simple method to create numerical or more complex procedural names for entries and groups on the fly.

```
#include <Fl_Preferences.H>
```


Public Member Functions

- [Name](#) (const char *format,...)
Creates a group name or entry name on the fly.
- [Name](#) (unsigned int n)
Creates a group name or entry name on the fly.
- **operator const char *** ()
Return the [Name](#) as a "C" string.

11.172.1 Detailed Description

'[Name](#)' provides a simple method to create numerical or more complex procedural names for entries and groups on the fly.

Example: `prefs.set(Fl_Preferences::Name("File%d",i),file[i]);`

See `test/preferences.cxx` as a sample for writing arrays into preferences.

'[Name](#)' is actually implemented as a class inside [Fl_Preferences](#). It casts into `const char*` and gets automatically destroyed after the enclosing call ends.

11.172.2 Constructor & Destructor Documentation

11.172.2.1 Name() [1/2]

```
Fl_Preferences::Name::Name (
    unsigned int n)
```

Creates a group name or entry name on the fly.

This version creates a simple unsigned integer as an entry name.

```
int n, i;
Fl_Preferences prev( appPrefs, "PreviousFiles" );
prev.get( "n", 0 );
for ( i=0; i<n; i++ )
    prev.get( Fl_Preferences::Name(i), prevFile[i], "" );
```

11.172.2.2 Name() [2/2]

```
Fl_Preferences::Name::Name (
    const char * format,
    ...)
```

Creates a group name or entry name on the fly.

This version creates entry names as in 'printf'.

```
int n, i;
Fl_Preferences prefs( USER, "matthiasm.com", "test" );
prev.get( "nFiles", 0 );
for ( i=0; i<n; i++ )
    prev.get( Fl_Preferences::Name( "File%d", i ), prevFile[i], "" );
```

The documentation for this class was generated from the following files:

- `Fl_Preferences.H`
- `Fl_Preferences.cxx`

11.173 Fl_Preferences::Node Class Reference

Public Member Functions

- void **add** (const char *line)
- [Node](#) * **addChild** (const char *path)
- const char * **child** (int ix)
- [Node](#) * **childNodes** (int ix)
- void **clearDirtyFlags** ()
- void **deleteAllChildren** ()
- void **deleteAllEntries** ()
- char **deleteEntry** (const char *name)

- char **dirty** ()
- [Entry](#) & **entry** (int i)
- [Node](#) * **find** (const char *path)
- [RootNode](#) * **findRoot** ()
- const char * **get** (const char *name)
- int **getEntry** (const char *name)
- const char * **name** ()
- int **nChildren** ()
- int **nEntry** ()
- **Node** (const char *path)
- [Node](#) * **parent** ()
- const char * **path** ()
- char **remove** ()
- [Node](#) * **search** (const char *path, int offset=0)
- void **set** (const char *line)
- void **set** (const char *name, const char *value)
- void **setParent** ([Node](#) *parent)
- void **setRoot** ([RootNode](#) *r)
- int **write** (FILE *f)

Static Public Attributes

- static int **lastEntrySet** = -1

The documentation for this class was generated from the following files:

- [Fl_Preferences.H](#)
- [Fl_Preferences.cxx](#)

11.174 [Fl_Paged_Device::page_format](#) Struct Reference

width, height and name of a page format

```
#include <Fl_Paged_Device.H>
```

Public Attributes

- int **height**
height in points
- const char * **name**
format name
- int **width**
width in points

11.174.1 Detailed Description

width, height and name of a page format

The documentation for this struct was generated from the following file:

- [Fl_Paged_Device.H](#)

11.175 FI_Terminal::PartialUtf8Buf Class Reference

Public Member Functions

- bool **append** (const char *p, int len)
- const char * **buf** (void) const
- int **buflen** (void) const
- void **clear** (void)
- bool **is_complete** (void) const
- bool **is_continuation** (char c)

The documentation for this class was generated from the following file:

- [FI_Terminal.H](#)

11.176 FI_Terminal::RingBuffer Class Reference

Public Member Functions

- void **change_disp_cols** (int dcols, const [CharStyle](#) &style)
- void **change_disp_rows** (int drows, const [CharStyle](#) &style)
- void **clear** (void)
- void **clear_disp_rows** (int sdrow, int edrow, const [CharStyle](#) &style)
- void **clear_hist** (void)
- void **create** (int drows, int dcols, int hrows)
- int **disp_cols** (void) const
- int **disp_erow** (void) const
- void **disp_rows** (int val)
- int **disp_rows** (void) const
- int **disp_srow** (void) const
- int **hist_cols** (void) const
- int **hist_erow** (void) const
- void **hist_rows** (int val)
- int **hist_rows** (void) const
- int **hist_srow** (void) const
- void **hist_use** (int val)
- int **hist_use** (void) const
- int **hist_use_srow** (void) const
- bool **is_disp_ring_row** (int grow) const
- bool **is_hist_ring_row** (int grow) const
- void **move_disp_row** (int src_row, int dst_row)
- int **offset** (void) const
- void **offset_adjust** (int rows)
- void **resize** (int drows, int dcols, int hrows, const [CharStyle](#) &style)
- [Utf8Char](#) * **ring_chars** (void)
- [Utf8Char](#) * **ring_chars** (void) const
- int **ring_cols** (void) const
- int **ring_erow** (void) const
- int **ring_rows** (void) const
- int **ring_srow** (void) const
- **RingBuffer** (int drows, int dcols, int hrows)
- void **scroll** (int rows, const [CharStyle](#) &style)
- [Utf8Char](#) * **u8c_disp_row** (int drow)
- const [Utf8Char](#) * **u8c_disp_row** (int drow) const
- [Utf8Char](#) * **u8c_hist_row** (int hrow)
- const [Utf8Char](#) * **u8c_hist_row** (int hrow) const

- [Utf8Char](#) * **u8c_hist_use_row** (int hurow)
- const [Utf8Char](#) * **u8c_hist_use_row** (int hurow) const
- [Utf8Char](#) * **u8c_ring_row** (int row)
- const [Utf8Char](#) * **u8c_ring_row** (int row) const

The documentation for this class was generated from the following files:

- [Fl_Terminal.H](#)
- [Fl_Terminal.cxx](#)

11.177 [Fl_Preferences::RootNode](#) Class Reference

Public Member Functions

- char * **filename** ()
- char **getPath** (char *[path](#), int pathlen)
- int **read** ()
- [Root](#) **root** ()
- [RootNode](#) ([Fl_Preferences](#) *)
- [RootNode](#) ([Fl_Preferences](#) *, const char *[path](#), const char *vendor, const char *application, [Root](#) flags)
- [RootNode](#) ([Fl_Preferences](#) *, [Root](#) root, const char *vendor, const char *application)
- int **write** ()

The documentation for this class was generated from the following files:

- [Fl_Preferences.H](#)
- [Fl_Preferences.cxx](#)

11.178 [Fl_Scroll::ScrollInfo](#) Struct Reference

Structure to manage scrollbar and widget interior sizes.

```
#include <Fl_Scroll.H>
```

Public Attributes

- [Fl_Region_LRTB](#) **child**
child bounding box: left/right/top/bottom
- int **hneeded**
horizontal scrollbar visibility
- [Fl_Scrollbar_Data](#) **hscroll**
horizontal scrollbar region + values
- [Fl_Region_XYWH](#) **innerbox**
widget's inner box, excluding scrollbars
- [Fl_Region_XYWH](#) **innerchild**
widget's inner box, including scrollbars
- int **scrollsize**
the effective scrollbar thickness (local or global)
- int **vneeded**
vertical scrollbar visibility
- [Fl_Scrollbar_Data](#) **vscroll**
vertical scrollbar region + values

11.178.1 Detailed Description

Structure to manage scrollbar and widget interior sizes.

This is filled out by `recalc_scrollbars()` for use in calculations that need to know the visible scroll area size, etc.

Version

1.3.3

The documentation for this struct was generated from the following file:

- `Fl_Scroll.H`

11.179 `Fl_Terminal::Selection` Class Reference

Public Member Functions

- `bool clear` (void)
- `bool dragged_off` (int row, int col, bool char_right)
- `int ecol` (void) const
- `void end` (void)
- `int erow` (void) const
- `bool extend` (int row, int col, bool char_right)
- `bool get_selection` (int &srow, int &scol, int &erow, int &ecol) const
Return selection start/end.
- `bool is_selection` (void) const
- `void push_clear` ()
- `void push_rowcol` (int row, int col, bool char_right)
- `int scol` (void) const
- `void scroll` (int nrows)
- `void select` (int srow, int scol, int erow, int ecol)
- `Selection` (`Fl_Terminal` *terminal)
- `void selectionbgcolor` (`Fl_Color` val)
- `Fl_Color selectionbgcolor` (void) const
- `void selectionfgcolor` (`Fl_Color` val)
- `Fl_Color selectionfgcolor` (void) const
- `int srow` (void) const
- `bool start` (int row, int col, bool char_right)
- `void start_push` ()
- `int state` (void) const

11.179.1 Member Function Documentation

11.179.1.1 `get_selection()`

```
bool Fl_Terminal::Selection::get_selection (
    int & srow,
    int & scol,
    int & erow,
    int & ecol) const
```

Return selection start/end.

Ensures (start < end) to allow walking 'forward' thru selection, left-to-right, top-to-bottom.

Returns:

- true – valid selection values returned
- false – no selection was made, returned values undefined

The documentation for this class was generated from the following files:

- `Fl_Terminal.H`
- `Fl_Terminal.cxx`

11.180 FI_Tile::Size_Range Struct Reference

Public Attributes

- int **maxh**
- int **maxw**
- int **minh**
- int **minw**

The documentation for this struct was generated from the following file:

- FI_Tile.H

11.181 FI_Text_Display::Style_Table_Entry Struct Reference

This structure associates the color, font, and font size of a string to draw with an attribute mask matching attr.

```
#include <FI_Text_Display.H>
```

Public Attributes

- unsigned **attr**
further attributes for the text style (see ATTR_BGCOLOR, etc.)
- [FI_Color](#) **bgcolor**
text background color if ATTR_BGCOLOR or ATTR_BGCOLOR_EXT is set
- [FI_Color](#) **color**
text color
- [FI_Font](#) **font**
text font
- [FI_Fonsize](#) **size**
text font size

11.181.1 Detailed Description

This structure associates the color, font, and font size of a string to draw with an attribute mask matching attr.

There must be one entry for each style that can be used in an [FI_Text_Display](#) for displaying text. The style table is an array of struct [Style_Table_Entry](#).

The style table is associated with an [FI_Text_Display](#) by using [FI_Text_Display::highlight_data\(\)](#).

See also

[FI_Text_Display::highlight_data\(\)](#)

The documentation for this struct was generated from the following file:

- FI_Text_Display.H

11.182 FI_Terminal::Utf8Char Class Reference

Public Member Functions

- [FI_Color](#) **attr_bg_color** (const [FI_Widget](#) *grp) const
- [FI_Color](#) **attr_fg_color** (const [FI_Widget](#) *grp) const
- [uchar](#) **attrib** (void) const
- [FI_Color](#) **bgcolor** (void) const
- [uchar](#) **charflags** (void) const
- void **clear** (const [CharStyle](#) &style)
- [FI_Color](#) **fgcolor** (void) const
- void **fl_font_set** (const [CharStyle](#) &style) const

- bool **is_char** (char c) const
- int **length** (void) const
- int **max_utf8** () const
- [Utf8Char](#) & **operator=** (const [Utf8Char](#) &o)
- double **pwidth** (void) const
- int **pwidth_int** (void) const
- void **show_char** (void) const
- void **show_char_info** (void) const
- void **text_ascii** (char c, const [CharStyle](#) &style)
- void **text_utf8** (const char *[text](#), int len, const [CharStyle](#) &style)
- const char * **text_utf8** (void) const
- **Utf8Char** (const [Utf8Char](#) &o)

The documentation for this class was generated from the following files:

- [Fl_Terminal.H](#)
- [Fl_Terminal.cxx](#)

Chapter 12

File Documentation

12.1 Enumerations.H File Reference

This file contains type definitions and general enumerations.

```
#include <FL/fl_config.h>
#include "Fl_Export.H"
#include "fl_types.h"
#include <FL/platform_types.h>
```

Macros

- `#define FL_IMAGE_WITH_ALPHA 0x40000000`

Version Numbers

FLTK defines some constants to help the programmer to find out, for which FLTK version a program is compiled.

The following constants are defined:

- `#define FL_ABI_VERSION FL_API_VERSION`
The FLTK ABI (Application Binary Interface) version number as an int.
- `#define FL_API_VERSION (FL_MAJOR_VERSION*10000 + FL_MINOR_VERSION*100 + FL_PATCH_VERSION)`
The FLTK API version number as an int.
- `#define FL_MAJOR_VERSION 1`
The major release version of this FLTK library.
- `#define FL_MINOR_VERSION 5`
The minor release version for this library.
- `#define FL_PATCH_VERSION 0`
The patch version for this library.
- `#define FL_VERSION`
The FLTK version number as a double.

Names of Non-ASCII keys and mouse buttons

The following constants define the names of non-ASCII keys on the keyboard and of mouse buttons for `FL_KEYBOARD` and `FL_SHORTCUT` events.

See also

[`Fl::event\_key\(\)`](#) and [`Fl::get\_key\(int\)`](#) (use ASCII letters for all other keys):

- `#define FL_Alt_Gr 0xfe03`
The AltGr key on some international keyboards.
- `#define FL_Alt_L 0xffe9`
The left alt key.
- `#define FL_Alt_R 0xffea`
The right alt key.

- **#define FL_Back** 0xEF26
Like back on a browser.
- **#define FL_BackSpace** 0xff08
The backspace key.
- **#define FL_Button** 0xfee8
A mouse button; use FL_Button + n for mouse button n.
- **#define FL_Caps_Lock** 0xffe5
The caps lock key.
- **#define FL_Control_L** 0xffe3
The lefthand control key.
- **#define FL_Control_R** 0xffe4
The righthand control key.
- **#define FL_Delete** 0xffff
The delete key.
- **#define FL_Down** 0xff54
The down arrow key.
- **#define FL_Eisu** 0xff2f
The Eisu key of JIS keyboards.
- **#define FL_End** 0xff57
The end key.
- **#define FL_Enter** 0xff0d
The enter key.
- **#define FL_Escape** 0xff1b
The escape key.
- **#define FL_F** 0xffbd
One of the function keys; use FL_F + n for function key n.
- **#define FL_F_Last** 0xffe0
The last function key; use to range-check function keys.
- **#define FL_Favorites** 0xEF30
Show favorite locations.
- **#define FL_Forward** 0xEF27
Like forward on a browser.
- **#define FL_Help** 0xff68
The 'help' key on Mac keyboards.
- **#define FL_Home** 0xff50
The home key.
- **#define FL_Home_Page** 0xEF18
Display user's home page.
- **#define FL_Insert** 0xff63
The insert key.
- **#define FL_Iso_Key** 0xff0c
The additional key of ISO keyboards.
- **#define FL_JIS_Underscore** 0xff31
The underscore key of JIS keyboards.
- **#define FL_Kana** 0xff2e
The Kana key of JIS keyboards.
- **#define FL_KP** 0xff80
One of the keypad numbers; use FL_KP + 'n' for digit n.
- **#define FL_KP_Enter** 0xff8d
The enter key on the keypad, same as FL_KP+'r'.
- **#define FL_KP_Last** 0xffbd
The last keypad key; use to range-check keypad.
- **#define FL_Left** 0xff51
The left arrow key.
- **#define FL_Mail** 0xEF19
Invoke user's mail program.
- **#define FL_Media_Next** 0xEF17
Next track.
- **#define FL_Media_Play** 0xEF14

- *Start playing of audio.*
- #define **FL_Media_Prev** 0xEF16
- *Previous track.*
- #define **FL_Media_Stop** 0xEF15
- *Stop playing audio.*
- #define **FL_Menu** 0xff67
- *The menu key.*
- #define **FL_Meta_L** 0xffe7
- *The left meta/Windows key.*
- #define **FL_Meta_R** 0xffe8
- *The right meta/Windows key.*
- #define **FL_Num_Lock** 0xff7f
- *The num lock key.*
- #define **FL_Page_Down** 0xff56
- *The page-down key.*
- #define **FL_Page_Up** 0xff55
- *The page-up key.*
- #define **FL_Pause** 0xff13
- *The pause key.*
- #define **FL_Print** 0xff61
- *The print (or print-screen) key.*
- #define **FL_Refresh** 0xEF29
- *Refresh the page.*
- #define **FL_Right** 0xff53
- *The right arrow key.*
- #define **FL_Scroll_Lock** 0xff14
- *The scroll lock key.*
- #define **FL_Search** 0xEF1B
- *Search.*
- #define **FL_Shift_L** 0xffe1
- *The lefthand shift key.*
- #define **FL_Shift_R** 0xffe2
- *The righthand shift key.*
- #define **FL_Sleep** 0xEF2F
- *Put system to sleep.*
- #define **FL_Stop** 0xEF28
- *Stop current operation.*
- #define **FL_Tab** 0xff09
- *The tab key.*
- #define **FL_Up** 0xff52
- *The up arrow key.*
- #define **FL_Volume_Down** 0xEF11
- *Volume control down.*
- #define **FL_Volume_Mute** 0xEF12
- *Mute sound from the system.*
- #define **FL_Volume_Up** 0xEF13
- *Volume control up.*
- #define **FL_Yen** 0xff30
- *The Yen key of JIS keyboards.*

Mouse Buttons

These constants define the button numbers for *FL_PUSH* and *FL_RELEASE* events.

See also

[Fl::event_button\(\)](#)

- **#define FL_BACK_MOUSE 4**
The back mouse button (side button 1)
- **#define FL_FORWARD_MOUSE 5**
The forward mouse button (side button 2)
- **#define FL_LEFT_MOUSE 1**
The left mouse button.
- **#define FL_MIDDLE_MOUSE 2**
The middle mouse button.
- **#define FL_RIGHT_MOUSE 3**
The right mouse button.

Event States

The following constants define bits in the [Fl::event_state\(\)](#) value.

- **#define FL_ALT 0x00080000**
One of the alt keys is down.
- **#define FL_BUTTON(n)**
Mouse button n ($n = 1..5$) is pushed, undefined if n outside $1..5$.
- **#define FL_BUTTON1 0x01000000**
Mouse button 1 is pushed (L)
- **#define FL_BUTTON2 0x02000000**
Mouse button 2 is pushed (M)
- **#define FL_BUTTON3 0x04000000**
Mouse button 3 is pushed (R)
- **#define FL_BUTTON4 0x08000000**
Mouse button 4 is pushed (BACK)
- **#define FL_BUTTON5 0x10000000**
Mouse button 5 is pushed (FORWARD)
- **#define FL_BUTTONS 0x1f000000**
Bitmask: any mouse button (1-5) is pushed.
- **#define FL_CAPS_LOCK 0x00020000**
The caps lock is on.
- **#define FL_CTRL 0x00040000**
One of the ctrl keys is down.
- **#define FL_KEY_MASK 0x0000ffff**
All keys are 16 bit for now.
- **#define FL_META 0x00400000**
One of the meta/Windows keys is down.
- **#define FL_NUM_LOCK 0x00100000**
The num lock is on.
- **#define FL_SCROLL_LOCK 0x00800000**
The scroll lock is on.
- **#define FL_SHIFT 0x00010000**
One of the shift keys is down.

Typedefs

- typedef int [Fl_Fontsize](#)
Size of a font in pixels.

Enumerations

- enum { `FL_READ` = 1 , `FL_WRITE` = 4 , `FL_EXCEPT` = 8 }
FD "when" conditions.
- enum `Fl_Arrow_Type` { `FL_ARROW_SINGLE` = 0x01 , `FL_ARROW_DOUBLE` = 0x02 , `FL_ARROW_CHOICE` = 0x03 , `FL_ARROW_RETURN` = 0x04 }
Arrow types define the type of arrow drawing function.
- enum `Fl_Damage` {
`FL_DAMAGE_CHILD` = 0x01 , `FL_DAMAGE_EXPOSE` = 0x02 , `FL_DAMAGE_SCROLL` = 0x04 ,
`FL_DAMAGE_OVERLAY` = 0x08 ,
`FL_DAMAGE_USER1` = 0x10 , `FL_DAMAGE_USER2` = 0x20 , `FL_DAMAGE_ALL` = 0x80 }
Damage masks.
- enum `Fl_Event` {
`FL_NO_EVENT` = 0 , `FL_PUSH` = 1 , `FL_RELEASE` = 2 , `FL_ENTER` = 3 ,
`FL_LEAVE` = 4 , `FL_DRAG` = 5 , `FL_FOCUS` = 6 , `FL_UNFOCUS` = 7 ,
`FL_KEYDOWN` = 8 , `FL_KEYBOARD` = 8 , `FL_KEYUP` = 9 , `FL_CLOSE` = 10 ,
`FL_MOVE` = 11 , `FL_SHORTCUT` = 12 , `FL_DEACTIVATE` = 13 , `FL_ACTIVATE` = 14 ,
`FL_HIDE` = 15 , `FL_SHOW` = 16 , `FL_PASTE` = 17 , `FL_SELECTIONCLEAR` = 18 ,
`FL_MOUSEWHEEL` = 19 , `FL_DND_ENTER` = 20 , `FL_DND_DRAG` = 21 , `FL_DND_LEAVE` = 22 ,
`FL_DND_RELEASE` = 23 , `FL_SCREEN_CONFIGURATION_CHANGED` = 24 , `FL_FULLSCREEN` = 25 ,
`FL_ZOOM_GESTURE` = 26 ,
`FL_ZOOM_EVENT` = 27 }
Every time a user moves the mouse pointer, clicks a button, or presses a key, an event is generated and sent to your application.
- enum `Fl_Labeltype` {
`FL_NORMAL_LABEL` = 0 , `FL_NO_LABEL` , `_FL_SHADOW_LABEL` , `_FL_ENGRAVED_LABEL` ,
`_FL_EMBOSSSED_LABEL` , `_FL_MULTI_LABEL` , `_FL_ICON_LABEL` , `_FL_IMAGE_LABEL` ,
`FL_FREE_LABELTYPE` }
The `labeltype()` method sets the type of the label.
- enum `Fl_Mode` {
`FL_RGB` = 0 , `FL_INDEX` = 1 , `FL_SINGLE` = 0 , `FL_DOUBLE` = 2 ,
`FL_ACCUM` = 4 , `FL_ALPHA` = 8 , `FL_DEPTH` = 16 , `FL_STENCIL` = 32 ,
`FL_RGB8` = 64 , `FL_MULTISAMPLE` = 128 , `FL_STEREO` = 256 , `FL_FAKE_SINGLE` = 512 ,
`FL_OPENGL3` = 1024 }
visual types and `Fl_Gl_Window::mode()` (values match Glut)
- enum `Fl_Orientation` {
`FL_ORIENT_NONE` = 0x00 , `FL_ORIENT_RIGHT` = 0x00 , `FL_ORIENT_NE` = 0x01 , `FL_ORIENT_UP` = 0x02 ,
`FL_ORIENT_NW` = 0x03 , `FL_ORIENT_LEFT` = 0x04 , `FL_ORIENT_SW` = 0x05 , `FL_ORIENT_DOWN` = 0x06 ,
`FL_ORIENT_SE` = 0x07 }
`Fl_Orientation` describes the orientation of a GUI element.

When Conditions

- enum `Fl_When` {
`FL_WHEN_NEVER` = 0 , `FL_WHEN_CHANGED` = 1 , `FL_WHEN_NOT_CHANGED` = 2 , `FL_WHEN_RELEASE` = 4 ,
`FL_WHEN_RELEASE_ALWAYS` = 6 , `FL_WHEN_ENTER_KEY` = 8 , `FL_WHEN_ENTER_KEY_ALWAYS` = 10 , `FL_WHEN_ENTER_KEY_CHANGED` = 11 ,
`FL_WHEN_CLOSED` = 16 }
These constants determine when a callback is performed.

Callback Reasons

- enum `Fl_Callback_Reason` {
`FL_REASON_UNKNOWN` = 0 , `FL_REASON_SELECTED` , `FL_REASON_DESELECTED` , `FL_REASON_RESELECTED`

```
,
FL_REASON_OPENED , FL_REASON_CLOSED , FL_REASON_DRAGGED , FL_REASON_CANCELLED
,
FL_REASON_CHANGED , FL_REASON_GOT_FOCUS , FL_REASON_LOST_FOCUS , FL_REASON_RELEASED
,
FL_REASON_ENTER_KEY , FL_REASON_USER = 32 }
```

These constants describe why a callback is performed.

Cursors

- enum `Fl_Cursor` {
`FL_CURSOR_DEFAULT` = 0 , `FL_CURSOR_ARROW` = 35 , `FL_CURSOR_CROSS` = 66 ,
`FL_CURSOR_WAIT` = 76 ,
`FL_CURSOR_INSERT` = 77 , `FL_CURSOR_HAND` = 31 , `FL_CURSOR_HELP` = 47 , `FL_CURSOR_MOVE`
= 27 ,
`FL_CURSOR_NS` = 78 , `FL_CURSOR_WE` = 79 , `FL_CURSOR_NWSE` = 80 , `FL_CURSOR_NESW` =
81 ,
`FL_CURSOR_N` = 70 , `FL_CURSOR_NE` = 69 , `FL_CURSOR_E` = 49 , `FL_CURSOR_SE` = 8 ,
`FL_CURSOR_S` = 9 , `FL_CURSOR_SW` = 7 , `FL_CURSOR_W` = 36 , `FL_CURSOR_NW` = 68 ,
`FL_CURSOR_NONE` = 255 }

The following constants define the mouse cursors that are available in FLTK.

Variables

- `Fl_Fontsize FL_NORMAL_SIZE`

normal font size

Box Types

FLTK standard box types

This enum defines the standard box types included with FLTK.

Note

The documented enum `Fl_Boxtype` contains some values (names) with leading underscores, e.g. `_FL_SHADOW_BOX`. This is due to technical reasons - please use the same values (names) without the leading underscore in your code! Enum values with leading underscores are reserved for internal use and subject to change without notice!

`FL_NO_BOX` means nothing is drawn at all, so whatever is already on the screen remains. The `FL_..._FRAME` types only draw their edges, leaving the interior unchanged. The blue color in the image below is the area that is

not drawn by the frame types.

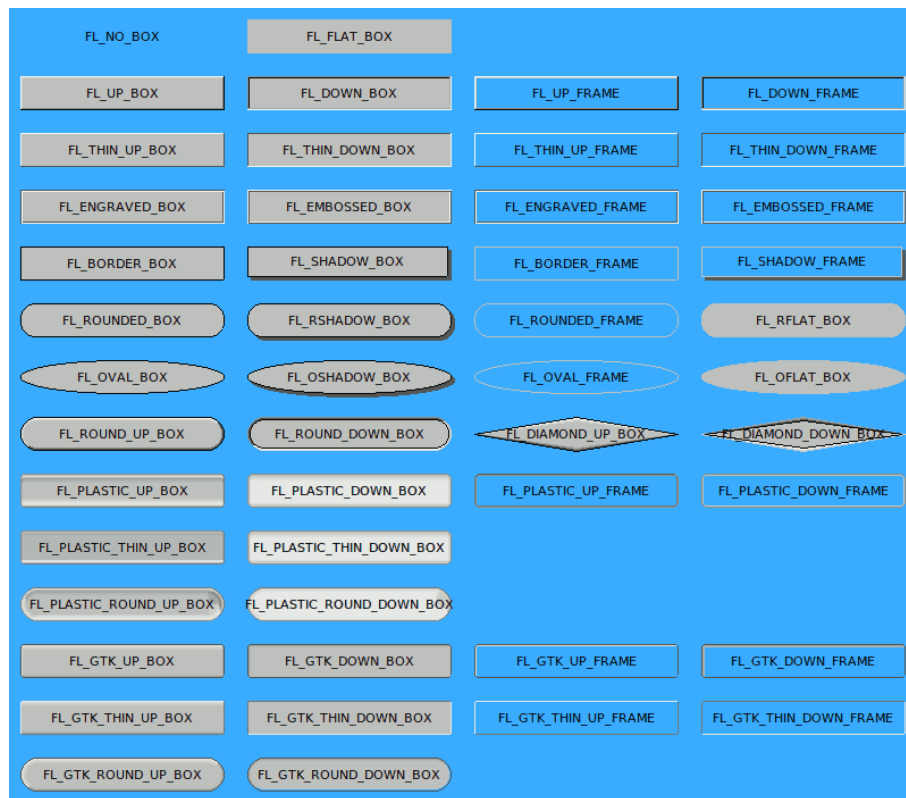


Figure 12.1 FLTK Standard Box Types

Note

Not all box types are depicted in the figure above. See enum [Fl_Boxtype](#) below for the complete list of box types.

See also

[Fl::get_system_colors\(\)](#)

- [Fl_Boxtype fl_box](#) ([Fl_Boxtype b](#))
Get the filled version of a frame.
- enum [Fl_Boxtype](#) {
[FL_NO_BOX](#) = 0, [FL_FLAT_BOX](#), [FL_UP_BOX](#), [FL_DOWN_BOX](#),
[FL_UP_FRAME](#), [FL_DOWN_FRAME](#), [FL_THIN_UP_BOX](#), [FL_THIN_DOWN_BOX](#),
[FL_THIN_UP_FRAME](#), [FL_THIN_DOWN_FRAME](#), [FL_ENGRAVED_BOX](#), [FL_EMBOSSSED_BOX](#),
[FL_ENGRAVED_FRAME](#), [FL_EMBOSSSED_FRAME](#), [FL_BORDER_BOX](#), [FL_SHADOW_BOX](#),
[FL_BORDER_FRAME](#), [FL_SHADOW_FRAME](#), [FL_ROUNDED_BOX](#), [FL_RSHADOW_BOX](#),
[FL_ROUNDED_FRAME](#), [FL_RFLAT_BOX](#), [FL_ROUND_UP_BOX](#), [FL_ROUND_DOWN_BOX](#),
[FL_DIAMOND_UP_BOX](#), [FL_DIAMOND_DOWN_BOX](#), [FL_OVAL_BOX](#), [FL_OSHADOW_BOX](#),
[FL_OVAL_FRAME](#), [FL_OFLAT_BOX](#), [FL_PLASTIC_UP_BOX](#), [FL_PLASTIC_DOWN_BOX](#),
[FL_PLASTIC_UP_FRAME](#), [FL_PLASTIC_DOWN_FRAME](#), [FL_PLASTIC_THIN_UP_BOX](#),
[FL_PLASTIC_THIN_DOWN_BOX](#),
[FL_PLASTIC_ROUND_UP_BOX](#), [FL_PLASTIC_ROUND_DOWN_BOX](#), [FL_GTK_UP_BOX](#),
[FL_GTK_DOWN_BOX](#),
[FL_GTK_UP_FRAME](#), [FL_GTK_DOWN_FRAME](#), [FL_GTK_THIN_UP_BOX](#), [FL_GTK_THIN_DOWN_BOX](#),
[FL_GTK_THIN_UP_FRAME](#), [FL_GTK_THIN_DOWN_FRAME](#), [FL_GTK_ROUND_UP_BOX](#),
[FL_GTK_ROUND_DOWN_BOX](#),
[FL_GLEAM_UP_BOX](#), [FL_GLEAM_DOWN_BOX](#), [FL_GLEAM_UP_FRAME](#), [FL_GLEAM_DOWN_FRAME](#)

```

,
_FL_GLEAM_THIN_UP_BOX , _FL_GLEAM_THIN_DOWN_BOX , _FL_GLEAM_ROUND_UP_BOX ,
_FL_GLEAM_ROUND_DOWN_BOX ,
_FL_OXY_UP_BOX , _FL_OXY_DOWN_BOX , _FL_OXY_UP_FRAME , _FL_OXY_DOWN_FRAME ,
_FL_OXY_THIN_UP_BOX , _FL_OXY_THIN_DOWN_BOX , _FL_OXY_THIN_UP_FRAME , _FL_OXY_THIN_DOWN_FRAME
,
_FL_OXY_ROUND_UP_BOX , _FL_OXY_ROUND_DOWN_BOX , _FL_OXY_BUTTON_UP_BOX ,
_FL_OXY_BUTTON_DOWN_BOX ,
FL_FREE_BOXTYPE , FL_MAX_BOXTYPE = 255 }

    FLTK standard box types.
• #define FL_CIRCLE_BOX FL_ROUND_DOWN_BOX
• #define FL_DIAMOND_BOX FL_DIAMOND_DOWN_BOX
• Fl\_Boxtype fl\_down (Fl\_Boxtype b)
    Get the "pressed" or "down" version of a box.
• #define FL_FRAME FL\_ENGRAVED\_FRAME
• Fl\_Boxtype fl\_frame (Fl\_Boxtype b)
    Get the unfilled, frame only version of a box.
• #define FL_FRAME_BOX FL\_ENGRAVED\_BOX

• Fl\_Labeltype fl\_define\_FL\_EMBOSSED\_LABEL ()
    Initializes the internal table entry for FL_EMBOSSED_LABEL and returns its internal value.
• Fl\_Labeltype fl\_define\_FL\_ENGRAVED\_LABEL ()
    Initializes the internal table entry for FL_ENGRAVED_LABEL and returns its internal value.
• Fl\_Labeltype fl\_define\_FL\_ICON\_LABEL ()
    Initializes the internal table entry for FL_ICON_LABEL and returns its internal value.
• Fl\_Labeltype fl\_define\_FL\_IMAGE\_LABEL ()
    Initializes the internal table entry for FL_IMAGE_LABEL and returns its internal value.
• Fl\_Labeltype fl\_define\_FL\_MULTI\_LABEL ()
    Initializes the internal table entry for FL_MULTI_LABEL and returns its internal value.
• Fl\_Labeltype fl\_define\_FL\_SHADOW\_LABEL ()
    Initializes the internal table entry for FL_SHADOW_LABEL and returns its internal value.
• #define FL_EMBOSSED_LABEL fl\_define\_FL\_EMBOSSED\_LABEL()
    Draws a label with embossed text.
• #define FL_ENGRAVED_LABEL fl\_define\_FL\_ENGRAVED\_LABEL()
    Draws a label with engraved text.
• #define FL_ICON_LABEL fl\_define\_FL\_ICON\_LABEL()
    Draws an icon as the label.
• #define FL_IMAGE_LABEL fl\_define\_FL\_IMAGE\_LABEL()
    Draws an image (Fl\_Image) as the label.
• #define FL_MULTI_LABEL fl\_define\_FL\_MULTI\_LABEL()
    Draws a label that can comprise several parts like text and images.
• #define FL_SHADOW_LABEL fl\_define\_FL\_SHADOW\_LABEL()
    Draws a label with shadows behind the text.
• #define FL_SYMBOL_LABEL FL\_NORMAL\_LABEL
    Sets the current label type and returns its corresponding Fl\_Labeltype value.

```

Colors

The [Fl_Color](#) type holds an FLTK color value.

Colors are either 8-bit indexes into a [virtual colormap](#) or 24-bit RGB color values. (See [Colors](#) for the default FLTK colormap)

Color indices occupy the lower 8 bits of the value, while RGB colors occupy the upper 24 bits, for a byte organization of RGBI.


```

Fl_Color => 0xrrggbbii
          | | | |
          | | | +--- index between 0 and 255
          | | +----- blue color component (8 bit)
          | +----- green component (8 bit)
          +----- red component (8 bit)

```

A color can have either an index or an rgb value. Colors with rgb set and an index >0 are reserved for special use.

- const [Fl_Color](#) **FL_BACKGROUND2_COLOR** = 7
the default background color for text, list, and valuator widgets
- const [Fl_Color](#) **FL_BACKGROUND_COLOR** = 49
Default background color.
- const [Fl_Color](#) **FL_BLACK** = 56
- const [Fl_Color](#) **FL_BLUE** = 216
- typedef unsigned int **Fl_Color**
An FLTK color value; see also [Colors](#)
- [Fl_Color](#) **fl_color_average** ([Fl_Color](#) c1, [Fl_Color](#) c2, float weight)
Returns the weighted average color between the two given colors.
- #define **FL_COLOR_CUBE** ([Fl_Color](#))56
- [Fl_Color](#) **fl_color_cube** (int r, int g, int b)
Returns a color out of the color cube.
- [Fl_Color](#) **fl_contrast** ([Fl_Color](#) fg, [Fl_Color](#) bg, int context=0, int size=0)
Returns a color that contrasts with the background color.
- typedef [Fl_Color](#) **Fl_Contrast_Function**([Fl_Color](#), [Fl_Color](#), int, int)
Type of a custom [fl_contrast\(\)](#) function.
- void **fl_contrast_function** ([Fl_Contrast_Function](#) *f)
Register a custom contrast function.
- int **fl_contrast_level** ()
Get the contrast level (sensitivity) of the [fl_contrast\(\)](#) method.
- void **fl_contrast_level** (int level)
Set the contrast level (sensitivity) of the [fl_contrast\(\)](#) method.
- enum **Fl_Contrast_Mode** {
 [FL_CONTRAST_NONE](#) = 0, [FL_CONTRAST_LEGACY](#), [FL_CONTRAST_CIELAB](#), [FL_CONTRAST_CUSTOM](#),
 [FL_CONTRAST_LAST](#) }
Define the possible modes to calculate [fl_contrast\(\)](#).
- int **fl_contrast_mode** ()
Return the current contrast algorithm (mode).
- void **fl_contrast_mode** (int mode)
Set the contrast algorithm (mode).
- const [Fl_Color](#) **FL_CYAN** = 223
- const [Fl_Color](#) **FL_DARK1** = 47
- const [Fl_Color](#) **FL_DARK2** = 45
- const [Fl_Color](#) **FL_DARK3** = 39
- const [Fl_Color](#) **FL_DARK_BLUE** = 136
- const [Fl_Color](#) **FL_DARK_CYAN** = 140
- const [Fl_Color](#) **FL_DARK_GREEN** = 60
- const [Fl_Color](#) **FL_DARK_MAGENTA** = 152
- const [Fl_Color](#) **FL_DARK_RED** = 72
- const [Fl_Color](#) **FL_DARK_YELLOW** = 76
- [Fl_Color](#) **fl_darker** ([Fl_Color](#) c)

- Returns a darker version of the specified color.*
- const `Fl_Color` **FL_FOREGROUND_COLOR** = 0
the default foreground color (0) used for labels and text
- #define **FL_FREE_COLOR** (`Fl_Color`)16
Colors numbered between FL_FREE_COLOR and FL_FREE_COLOR + FL_NUM_FREE_COLOR - 1 are free for the user to be given any value using `Fl::set_color()`.
- #define **FL_GRAY** `FL_BACKGROUND_COLOR`
- const `Fl_Color` **FL_GRAY0** = 32
- #define **FL_GRAY_RAMP** (`Fl_Color`)32
- `Fl_Color` **fl_gray_ramp** (int i)
Returns a gray color value from black (i == 0) to white (i == FL_NUM_GRAY - 1).
- const `Fl_Color` **FL_GREEN** = 63
- `Fl_Color` **fl_inactive** (`Fl_Color` c)
Returns the inactive, dimmed version of the given color.
- const `Fl_Color` **FL_INACTIVE_COLOR** = 8
the inactive foreground color
- const `Fl_Color` **FL_LIGHT1** = 50
- const `Fl_Color` **FL_LIGHT2** = 52
- const `Fl_Color` **FL_LIGHT3** = 54
- `Fl_Color` **fl_lighter** (`Fl_Color` c)
Returns a lighter version of the specified color.
- double **fl_lightness** (`Fl_Color` color)
Return the perceived lightness of a color.
- double **fl_luminance** (`Fl_Color` color)
Return the raw / physical luminance of a color.
- const `Fl_Color` **FL_MAGENTA** = 248
- #define **FL_NUM_BLUE** 5
- #define **FL_NUM_FREE_COLOR** 16
- #define **FL_NUM_GRAY** 24
- #define **FL_NUM_GREEN** 8
- #define **FL_NUM_RED** 5
- const `Fl_Color` **FL_RED** = 88
- `Fl_Color` **fl_rgb_color** (`uchar` g)
Returns the 24-bit color value closest to g (grayscale).
- `Fl_Color` **fl_rgb_color** (`uchar` r, `uchar` g, `uchar` b)
Returns the 24-bit color value closest to r, g, b.
- const `Fl_Color` **FL_SELECTION_COLOR** = 15
the default selection/highlight color
- const `Fl_Color` **FL_WHITE** = 255
- const `Fl_Color` **FL_YELLOW** = 95

Alignment Flags

Flags to control the label alignment.

This controls how the label is displayed next to or inside the widget. The default value is `FL_ALIGN_CENTER` (0) for most widgets, which centers the label inside the widget.

All alignment flags use the common prefix "`FL_ALIGN_`". In the following descriptions this prefix is sometimes omitted for brevity.

Flags can be or'd to achieve a combination of alignments, but there are some "*magic values*" (e.g. combinations of TOP and BOTTOM and of LEFT and RIGHT) that have special meanings (see below). For instance:

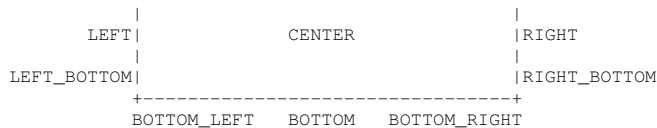
`FL_ALIGN_TOP_LEFT == (FL_ALIGN_TOP | FL_ALIGN_LEFT) != FL_ALIGN_LEFT_TOP.`

Outside alignments (`FL_ALIGN_INSIDE` is not set):

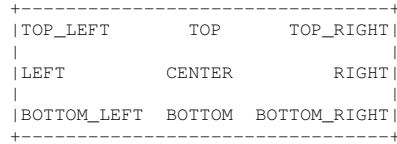
```

      TOP_LEFT      TOP      TOP_RIGHT
      +-----+
LEFT_TOP |                               | RIGHT_TOP

```



Inside alignments (`FL_ALIGN_INSIDE` is set):



See also

[Fl_Align](#), [FL_ALIGN_CENTER](#), etc.

Note

1. Bit positions not defined in the following constants of type `Fl_Align` are reserved for future extensions. Do not use.
2. The "magic values" (`FL_ALIGN_`)`LEFT_TOP`, `RIGHT_TOP`, `LEFT_BOTTOM`, and `RIGHT_BOTTOM` must not be used together with `FL_ALIGN_INSIDE`. Use `TOP_LEFT`, `TOP_RIGHT`, `BOTTOM_LEFT`, or `BOTTOM_RIGHT` instead.
3. Although bits can be or'd together there are some unused/illegal combinations, for instance:
 - setting both `FL_ALIGN_TOP` and `FL_ALIGN_BOTTOM` in combinations other than those given in the `Fl_Align` constants below (magic values)
 - setting both `FL_ALIGN_LEFT` and `FL_ALIGN_RIGHT` in combinations other than those given in the `Fl_Align` constants below (magic values)
 - using one of the "magic values" (2) together with `FL_ALIGN_INSIDE`

Using illegal bit combinations or undefined bits may yield unexpected behavior, and this behavior may be changed without notice in future FLTK versions.

- typedef unsigned **Fl_Align**
FLTK type for alignment control.
- const [Fl_Align](#) **FL_ALIGN_BOTTOM** = 0x0002
Align the label at the bottom of the widget.
- const [Fl_Align](#) **FL_ALIGN_BOTTOM_LEFT** = [FL_ALIGN_BOTTOM](#) | [FL_ALIGN_LEFT](#)
- const [Fl_Align](#) **FL_ALIGN_BOTTOM_RIGHT** = [FL_ALIGN_BOTTOM](#) | [FL_ALIGN_RIGHT](#)
- const [Fl_Align](#) **FL_ALIGN_CENTER** = 0x0000
Align the label horizontally in the middle.
- const [Fl_Align](#) **FL_ALIGN_CLIP** = 0x0040
All parts of the label that are larger than the widget will not be drawn.
- const [Fl_Align](#) **FL_ALIGN_IMAGE_BACKDROP** = 0x0200
If the label contains an image, draw the image or deimage in the background.
- const [Fl_Align](#) **FL_ALIGN_IMAGE_MASK** = 0x0320
Mask value to test for image alignment flags.
- const [Fl_Align](#) **FL_ALIGN_IMAGE_NEXT_TO_TEXT** = 0x0100
If the label contains an image, draw the text to the right of the image.
- const [Fl_Align](#) **FL_ALIGN_IMAGE_OVER_TEXT** = 0x0000
If the label contains an image, draw the text below the image.
- const [Fl_Align](#) **FL_ALIGN_INSIDE** = 0x0010
Draw the label inside of the widget.
- const [Fl_Align](#) **FL_ALIGN_LEFT** = 0x0004
Align the label at the left of the widget.
- const [Fl_Align](#) **FL_ALIGN_LEFT_BOTTOM** = 0x000d

- Outside only, left of widget, bottom position, magic value: TOP | LEFT | RIGHT.*

 - const `FL_Align FL_ALIGN_LEFT_TOP` = 0x0007
- Outside only, left of widget, top position, magic value: TOP | BOTTOM | LEFT.*

 - const `FL_Align FL_ALIGN_NOWRAP` = 0x0000
- Nothing, same as FL_ALIGN_CENTER, for back compatibility.*

 - const `FL_Align FL_ALIGN_POSITION_MASK` = 0x000f
- Mask value to test for TOP, BOTTOM, LEFT, and RIGHT flags.*

 - const `FL_Align FL_ALIGN_RIGHT` = 0x0008
- Align the label to the right of the widget.*

 - const `FL_Align FL_ALIGN_RIGHT_BOTTOM` = 0x000e
- Outside only, right of widget, bottom position, magic value: BOTTOM | LEFT | RIGHT.*

 - const `FL_Align FL_ALIGN_RIGHT_TOP` = 0x000b
- Outside only, right of widget, top position, magic value: TOP | BOTTOM | RIGHT.*

 - const `FL_Align FL_ALIGN_TEXT_NEXT_TO_IMAGE` = 0x0120
- If the label contains an image, draw the text to the left of the image.*

 - const `FL_Align FL_ALIGN_TEXT_OVER_IMAGE` = 0x0020
- If the label contains an image, draw the text on top of the image.*

 - const `FL_Align FL_ALIGN_TOP` = 0x0001
- Align the label at the top of the widget.*

 - const `FL_Align FL_ALIGN_TOP_LEFT` = `FL_ALIGN_TOP` | `FL_ALIGN_LEFT`
 - const `FL_Align FL_ALIGN_TOP_RIGHT` = `FL_ALIGN_TOP` | `FL_ALIGN_RIGHT`
 - const `FL_Align FL_ALIGN_WRAP` = 0x0080
- Wrap text that does not fit the width of the widget.*

Font Numbers

The following constants define the standard FLTK fonts:

- const `FL_Font FL_BOLD` = 1
- add this to Helvetica, Courier, or Times*
- const `FL_Font FL_BOLD_ITALIC` = 3
- add this to Helvetica, Courier, or Times*
- const `FL_Font FL_COURIER` = 4
- Courier normal.*
- const `FL_Font FL_COURIER_BOLD` = 5
- Courier bold.*
- const `FL_Font FL_COURIER_BOLD_ITALIC` = 7
- Courier bold-italic.*
- const `FL_Font FL_COURIER_ITALIC` = 6
- Courier italic.*
- typedef int `FL_Font`
- A font number is an index into the internal font table.*
- const `FL_Font FL_FREE_FONT` = 16
- first one to allocate*
- const `FL_Font FL_HELVETICA` = 0
- Helvetica (or Arial) normal (0)*
- const `FL_Font FL_HELVETICA_BOLD` = 1
- Helvetica (or Arial) bold.*
- const `FL_Font FL_HELVETICA_BOLD_ITALIC` = 3
- Helvetica (or Arial) bold-oblique.*
- const `FL_Font FL_HELVETICA_ITALIC` = 2
- Helvetica (or Arial) oblique.*

- const `FL_Font FL_ITALIC` = 2
add this to Helvetica, Courier, or Times
- const `FL_Font FL_SCREEN` = 13
Default monospaced screen font.
- const `FL_Font FL_SCREEN_BOLD` = 14
Default monospaced bold screen font.
- const `FL_Font FL_SYMBOL` = 12
Standard symbol font.
- const `FL_Font FL_TIMES` = 8
Times roman.
- const `FL_Font FL_TIMES_BOLD` = 9
Times roman bold.
- const `FL_Font FL_TIMES_BOLD_ITALIC` = 11
Times roman bold-italic.
- const `FL_Font FL_TIMES_ITALIC` = 10
Times roman italic.
- const `FL_Font FL_ZAPF_DINGBATS` = 15
Zapf-dingbats font.

12.1.1 Detailed Description

This file contains type definitions and general enumerations.

12.1.2 Macro Definition Documentation

12.1.2.1 FL_ABI_VERSION

```
#define FL_ABI_VERSION FL_API_VERSION
```

The FLTK ABI (Application Binary Interface) version number as an *int*.

FL_ABI_VERSION is an *int* that describes the major, minor, and patch ABI version numbers in the same format as FL_API_VERSION.

The ABI version number FL_ABI_VERSION is usually the same as the API version FL_API_VERSION with the last two digits set to '00'.

FLTK retains the ABI (Application Binary Interface) during patch releases of the same major and minor versions. Examples:

FLTK Version	FL_API_VERSION	FL_ABI_VERSION	FL_VERSION (deprecated)
1.3.0	10300	10300	1.0300
1.3.4	10304	10300	1.0304

Version 1.2.3 is actually stored as 10203 to allow for more than 9 minor and patch releases.

The FL_MAJOR_VERSION, FL_MINOR_VERSION, and FL_PATCH_VERSION constants give the integral values for the major, minor, and patch releases respectively.

To enable new ABI-breaking features in patch releases you can configure FLTK to use a higher FL_ABI_VERSION.

See also

README.abi-version.txt

12.1.2.2 FL_API_VERSION

```
#define FL_API_VERSION (FL_MAJOR_VERSION*10000 + FL_MINOR_VERSION*100 + FL_PATCH_VERSION)
```

The FLTK API version number as an *int*.

FL_API_VERSION is an *int* that describes the major, minor, and patch version numbers.

Version 1.2.3 is actually stored as 10203 to allow for more than 9 minor and patch releases.

The FL_MAJOR_VERSION, FL_MINOR_VERSION, and FL_PATCH_VERSION constants give the integral values for the major, minor, and patch releases respectively.

Note

FL_API_VERSION is intended to replace the deprecated *double* FL_VERSION.

See also

[Fl::api_version\(\)](#)

12.1.2.3 FL_BUTTON

```
#define FL_BUTTON(  
    n)
```

Value:

```
(0x00800000«(n))
```

Mouse button *n* (*n* = 1..5) is pushed, *undefined* if *n* outside 1..5.

12.1.2.4 FL_IMAGE_LABEL

```
#define FL_IMAGE_LABEL fl_define_FL_IMAGE_LABEL()
```

Draws an image ([Fl_Image](#)) as the label.

This is useful for one particular part of an [Fl_Multi_Label](#). Use [Fl_Widget::image\(\)](#) and/or [Fl_Widget::deimage\(\)](#) for normal widgets with images as labels.

12.1.2.5 FL_MAJOR_VERSION

```
#define FL_MAJOR_VERSION 1
```

The major release version of this FLTK library.

See also

[FL_VERSION](#)

12.1.2.6 FL_MINOR_VERSION

```
#define FL_MINOR_VERSION 5
```

The minor release version for this library.

FLTK remains mostly source-code compatible between minor version changes.

12.1.2.7 FL_MULTI_LABEL

```
#define FL_MULTI_LABEL fl_define_FL_MULTI_LABEL()
```

Draws a label that can comprise several parts like text and images.

See also

[Fl_Multi_Label](#)

12.1.2.8 FL_PATCH_VERSION

```
#define FL_PATCH_VERSION 0
```

The patch version for this library.

FLTK remains binary compatible between patches.

12.1.2.9 FL_SYMBOL_LABEL

```
#define FL_SYMBOL_LABEL FL_NORMAL_LABEL
```

Sets the current label type and returns its corresponding [Fl_Labeltype](#) value.

FL_SYMBOL_LABEL is an alias for FL_NORMAL_LABEL.

'@' symbols can be drawn with normal labels as well.

This definition is for historical reasons only (forms compatibility). You should use FL_NORMAL_LABEL instead.

12.1.2.10 FL_VERSION

```
#define FL_VERSION
```

Value:

```
( (double)FL_MAJOR_VERSION + \
(double)FL_MINOR_VERSION * 0.01 + \
(double)FL_PATCH_VERSION * 0.0001 )
```

The FLTK version number as a *double*.

FL_VERSION is a *double* that describes the major, minor, and patch version numbers.

Version 1.2.3 is actually stored as 1.0203 to allow for more than 9 minor and patch releases.

Deprecated This *double* version number is retained for compatibility with existing program code. New code should use *int* FL_API_VERSION instead. FL_VERSION is deprecated because comparisons of floating point values may fail due to rounding errors. However, there are currently no plans to remove this deprecated constant.

FL_VERSION is equivalent to $(double)FL\_API\_VERSION / 10000$.

See also

[Fl::version\(\)](#) (deprecated as well)

[FL_API_VERSION](#)

[Fl::api_version\(\)](#)

12.1.3 Typedef Documentation

12.1.3.1 Fl_Contrast_Function

```
typedef Fl_Color Fl_Contrast_Function(Fl_Color, Fl_Color, int, int)
```

Type of a custom [fl_contrast\(\)](#) function.

Use this signature to define your own custom [fl_contrast\(\)](#) function together with [fl_contrast_mode\(FL_↵ CONTRAST_CUSTOM\)](#). Example:

```
Fl_Color my_contrast(Fl_Color fg, Fl_Color bg, int context, int size) {
    // calculate contrast and ...
    return color;
}
// call this early in your main() program:
fl_contrast_function(my_contrast);
fl_contrast_mode(FL_CONTRAST_CUSTOM);
fl_contrast_level(50); // optional, must be called after fl_contrast_mode()
```

For parameters and types see [fl_contrast\(Fl_Color, Fl_Color, int, int\)](#).

See also

[fl_contrast\(Fl_Color, Fl_Color, int, int\)](#)

[fl_contrast_mode\(int\)](#)

12.1.3.2 Fl_Fontsize

```
typedef int Fl_Fontsize
```

Size of a font in pixels.

This is the approximate height of a font in pixels.

12.1.4 Enumeration Type Documentation

12.1.4.1 anonymous enum

```
anonymous enum
```

FD "when" conditions.

Enumerator

FL_READ	Call the callback when there is data to be read.
FL_WRITE	Call the callback when data can be written without blocking.
FL_EXCEPT	Call the callback if an exception occurs on the file.

12.1.4.2 Fl_Arrow_Type

enum [Fl_Arrow_Type](#)

Arrow types define the type of arrow drawing function.

FLTK schemes can draw several graphical elements in their particular way. One of these elements is an arrow type that can be in different GUI elements like scrollbars, choice buttons, and FLTK's [Fl_Return_Button](#).

Note

This enum is not yet stable (as of FLTK 1.4.0) and may be changed without notice as necessary.

Since

1.4.0

Enumerator

FL_ARROW_SINGLE	Single arrow, e.g. in Fl_Scrollbar .
FL_ARROW_DOUBLE	Double arrow, e.g. in Fl_Counter .
FL_ARROW_CHOICE	Dropdown box, e.g. in Fl_Choice .
FL_ARROW_RETURN	Return arrow, e.g. in Fl_Return_Button .

12.1.4.3 Fl_Boxtype

enum [Fl_Boxtype](#)

FLTK standard box types.

This enum defines the standard box types included with FLTK.

Note

The documented enum [Fl_Boxtype](#) contains some values (names) with leading underscores, e.g. **`_FL_SHADOW_BOX`**. This is due to technical reasons - please use the same values (names) without the leading underscore in your code! Enum values with leading underscores are reserved for internal use and subject to change without notice!

Enumerator

FL_NO_BOX	nothing is drawn at all, this box is invisible
FL_FLAT_BOX	a flat box
FL_UP_BOX	see figure Standard Box Types
FL_DOWN_BOX	see figure Standard Box Types
FL_UP_FRAME	see figure Standard Box Types
FL_DOWN_FRAME	see figure Standard Box Types
FL_THIN_UP_BOX	see figure Standard Box Types
FL_THIN_DOWN_BOX	see figure Standard Box Types
FL_THIN_UP_FRAME	see figure Standard Box Types
FL_THIN_DOWN_FRAME	see figure Standard Box Types
FL_ENGRAVED_BOX	see figure Standard Box Types
FL_EMBOSSSED_BOX	see figure Standard Box Types
FL_ENGRAVED_FRAME	see figure Standard Box Types
FL_EMBOSSSED_FRAME	see figure Standard Box Types
FL_BORDER_BOX	see figure Standard Box Types
_FL_SHADOW_BOX	see figure Standard Box Types , use FL_SHADOW_BOX
FL_BORDER_FRAME	see figure Standard Box Types

Enumerator

_FL_SHADOW_FRAME	see figure Standard Box Types , use FL_SHADOW_FRAME
_FL_ROUNDED_BOX	see figure Standard Box Types , use FL_ROUNDED_BOX
_FL_RSHADOW_BOX	see figure Standard Box Types , use FL_RSHADOW_BOX
_FL_ROUNDED_FRAME	see figure Standard Box Types , use FL_ROUNDED_FRAME
_FL_RFLAT_BOX	see figure Standard Box Types , use FL_RFLAT_BOX
_FL_ROUND_UP_BOX	see figure Standard Box Types , use FL_ROUND_UP_BOX
_FL_ROUND_DOWN_BOX	see figure Standard Box Types , use FL_ROUND_DOWN_BOX
_FL_DIAMOND_UP_BOX	see figure Standard Box Types , use FL_DIAMOND_UP_BOX
_FL_DIAMOND_DOWN_BOX	see figure Standard Box Types , use FL_DIAMOND_DOWN_BOX
_FL_OVAL_BOX	see figure Standard Box Types , use FL_OVAL_BOX
_FL_OSHADOW_BOX	see figure Standard Box Types , use FL_OSHADOW_BOX
_FL_OVAL_FRAME	see figure Standard Box Types , use FL_OVAL_FRAME
_FL_OFLAT_BOX	see figure Standard Box Types , use FL_OFLAT_BOX
_FL_PLASTIC_UP_BOX	plastic version of FL_UP_BOX, use FL_PLASTIC_UP_BOX
_FL_PLASTIC_DOWN_BOX	plastic version of FL_DOWN_BOX, use FL_PLASTIC_DOWN_BOX
_FL_PLASTIC_UP_FRAME	plastic version of FL_UP_FRAME, use FL_PLASTIC_UP_FRAME
_FL_PLASTIC_DOWN_FRAME	plastic version of FL_DOWN_FRAME, use FL_PLASTIC_DOWN_FRAME
_FL_PLASTIC_THIN_UP_BOX	plastic version of FL_THIN_UP_BOX, use FL_PLASTIC_THIN_UP_BOX
_FL_PLASTIC_THIN_DOWN_BOX	plastic version of FL_THIN_DOWN_BOX, use FL_PLASTIC_THIN_DOWN_BOX
_FL_PLASTIC_ROUND_UP_BOX	plastic version of FL_ROUND_UP_BOX, use FL_PLASTIC_ROUND_UP_BOX
_FL_PLASTIC_ROUND_DOWN_BOX	plastic version of FL_ROUND_DOWN_BOX, use FL_PLASTIC_ROUND_DOWN_BOX
_FL_GTK_UP_BOX	gtk+ version of FL_UP_BOX, use FL_GTK_UP_BOX
_FL_GTK_DOWN_BOX	gtk+ version of FL_DOWN_BOX, use FL_GTK_DOWN_BOX
_FL_GTK_UP_FRAME	gtk+ version of FL_UP_FRAME, use FL_GTK_UP_FRAME
_FL_GTK_DOWN_FRAME	gtk+ version of FL_DOWN_FRAME, use FL_GTK_DOWN_FRAME
_FL_GTK_THIN_UP_BOX	gtk+ version of FL_THIN_UP_BOX, use FL_GTK_THIN_UP_BOX
_FL_GTK_THIN_DOWN_BOX	gtk+ version of FL_THIN_DOWN_BOX, use FL_GTK_THIN_DOWN_BOX
_FL_GTK_THIN_UP_FRAME	gtk+ version of FL_THIN_UP_FRAME, use FL_GTK_THIN_UP_FRAME
_FL_GTK_THIN_DOWN_FRAME	gtk+ version of FL_THIN_DOWN_FRAME, use FL_GTK_THIN_DOWN_FRAME
_FL_GTK_ROUND_UP_BOX	gtk+ version of FL_ROUND_UP_BOX, use FL_GTK_ROUND_UP_BOX
_FL_GTK_ROUND_DOWN_BOX	gtk+ version of FL_ROUND_DOWN_BOX, use FL_GTK_ROUND_DOWN_BOX
_FL_GLEAM_UP_BOX	gleam version of FL_UP_BOX, use FL_GLEAM_UP_BOX
_FL_GLEAM_DOWN_BOX	gleam version of FL_DOWN_BOX, use FL_GLEAM_DOWN_BOX
_FL_GLEAM_UP_FRAME	gleam version of FL_UP_FRAME, use FL_GLEAM_UP_FRAME
_FL_GLEAM_DOWN_FRAME	gleam version of FL_DOWN_FRAME, use FL_GLEAM_DOWN_FRAME
_FL_GLEAM_THIN_UP_BOX	gleam version of FL_THIN_UP_BOX, use FL_GLEAM_THIN_UP_BOX

Enumerator

<code>_FL_GLEAM_THIN_DOWN_BOX</code>	gleam version of <code>FL_THIN_DOWN_BOX</code> , use <code>FL_GLEAM_THIN_DOWN_BOX</code>
<code>_FL_GLEAM_ROUND_UP_BOX</code>	gleam version of <code>FL_ROUND_UP_BOX</code> , use <code>FL_GLEAM_ROUND_UP_BOX</code>
<code>_FL_GLEAM_ROUND_DOWN_BOX</code>	gleam version of <code>FL_ROUND_DOWN_BOX</code> , use <code>FL_GLEAM_ROUND_DOWN_BOX</code>
<code>_FL_OXY_UP_BOX</code>	oxy version of <code>FL_UP_BOX</code> , use <code>FL_OXY_UP_BOX</code>
<code>_FL_OXY_DOWN_BOX</code>	oxy version of <code>FL_DOWN_BOX</code> , use <code>FL_OXY_DOWN_BOX</code>
<code>_FL_OXY_UP_FRAME</code>	oxy version of <code>FL_UP_FRAME</code> , use <code>FL_OXY_UP_FRAME</code>
<code>_FL_OXY_DOWN_FRAME</code>	oxy version of <code>FL_DOWN_FRAME</code> , use <code>FL_OXY_DOWN_FRAME</code>
<code>_FL_OXY_THIN_UP_BOX</code>	oxy version of <code>FL_THIN_UP_BOX</code> , use <code>FL_OXY_THIN_UP_BOX</code>
<code>_FL_OXY_THIN_DOWN_BOX</code>	oxy version of <code>FL_THIN_DOWN_BOX</code> , use <code>FL_OXY_THIN_DOWN_BOX</code>
<code>_FL_OXY_THIN_UP_FRAME</code>	oxy version of <code>FL_THIN_UP_FRAME</code> , use <code>FL_OXY_THIN_UP_FRAME</code>
<code>_FL_OXY_THIN_DOWN_FRAME</code>	oxy version of <code>FL_THIN_DOWN_FRAME</code> , use <code>FL_OXY_THIN_DOWN_FRAME</code>
<code>_FL_OXY_ROUND_UP_BOX</code>	oxy version of <code>FL_ROUND_UP_BOX</code> , use <code>FL_OXY_ROUND_UP_BOX</code>
<code>_FL_OXY_ROUND_DOWN_BOX</code>	oxy version of <code>FL_ROUND_DOWN_BOX</code> , use <code>FL_OXY_ROUND_DOWN_BOX</code>
<code>_FL_OXY_BUTTON_UP_BOX</code>	<code>FL_OXY_BUTTON_UP_BOX</code> (new boxtype ?), use <code>FL_OXY_BUTTON_UP_BOX</code> .
<code>_FL_OXY_BUTTON_DOWN_BOX</code>	<code>FL_OXY_BUTTON_DOWN_BOX</code> (new boxtype ?), use <code>FL_OXY_BUTTON_DOWN_BOX</code> .
<code>FL_FREE_BOXTYPE</code>	the first free box type for creation of new box types
<code>FL_MAX_BOXTYPE</code>	highest legal index for a box type

12.1.4.4 `Fl_Callback_Reason`

enum `Fl_Callback_Reason`

These constants describe why a callback is performed.

See also

[Fl::callback_reason\(\)](#), [Fl_Widget::when\(\)](#), [Fl_When](#)

Enumerator

<code>FL_REASON_UNKNOWN</code>	unknown or unset reason
<code>FL_REASON_SELECTED</code>	an item was selected
<code>FL_REASON_DESELECTED</code>	an item was de-selected
<code>FL_REASON_RESELECTED</code>	an item was re-selected (double-clicked).
<code>FL_REASON_OPENED</code>	an item was opened
<code>FL_REASON_CLOSED</code>	an item was closed
<code>FL_REASON_DRAGGED</code>	an item was dragged into a new place
<code>FL_REASON_CANCELLED</code>	a dialog was cancelled
<code>FL_REASON_CHANGED</code>	the value of the widget was modified
<code>FL_REASON_GOT_FOCUS</code>	a widget received focus
<code>FL_REASON_LOST_FOCUS</code>	a widget lost focus
<code>FL_REASON_RELEASED</code>	the mouse button was released

Enumerator

FL_REASON_ENTER_KEY	user finished input pressing Enter
FL_REASON_USER	user defined callback reasons

12.1.4.5 Fl_Contrast_Mode

enum [Fl_Contrast_Mode](#)

Define the possible modes to calculate [fl_contrast\(\)](#).

Enumerator

FL_CONTRAST_NONE	always return foreground color
FL_CONTRAST_LEGACY	legacy (FLTK 1.3.x) contrast function
FL_CONTRAST_CIELAB	new (FLTK 1.4.0) default function
FL_CONTRAST_CUSTOM	optional custom contrast function
FL_CONTRAST_LAST	internal use only (invalid contrast mode)

12.1.4.6 Fl_Cursor

enum [Fl_Cursor](#)

The following constants define the mouse cursors that are available in FLTK.

Cursors are provided by the system when available, or bitmaps built into FLTK as a fallback.

Enumerator

FL_CURSOR_DEFAULT	the default cursor, usually an arrow:
FL_CURSOR_ARROW	an arrow pointer:
FL_CURSOR_CROSS	crosshair:
FL_CURSOR_WAIT	busy indicator (for instance hourglass): ,
FL_CURSOR_INSERT	I-beam:
FL_CURSOR_HAND	pointing hand:
FL_CURSOR_HELP	question mark pointer: ?
FL_CURSOR_MOVE	4-pointed arrow or hand: ,
FL_CURSOR_NS	up/down resize:
FL_CURSOR_WE	left/right resize:
FL_CURSOR_NWSE	diagonal resize:
FL_CURSOR_NESW	diagonal resize:
FL_CURSOR_N	upwards resize:
FL_CURSOR_NE	upwards, right resize:
FL_CURSOR_E	rightwards resize:
FL_CURSOR_SE	downwards, right resize:
FL_CURSOR_S	downwards resize:
FL_CURSOR_SW	downwards, left resize:
FL_CURSOR_W	leftwards resize:
FL_CURSOR_NW	upwards, left resize:
FL_CURSOR_NONE	invisible.

12.1.4.7 Fl_Damage

enum [Fl_Damage](#)

Damage masks.

Enumerator

FL_DAMAGE_CHILD	A child needs to be redrawn.
FL_DAMAGE_EXPOSE	The window was exposed.
FL_DAMAGE_SCROLL	The Fl_Scroll widget was scrolled. Used by other widgets for other widget specific damages.
FL_DAMAGE_OVERLAY	The overlay planes need to be redrawn.
FL_DAMAGE_USER1	First user-defined damage bit.
FL_DAMAGE_USER2	Second user-defined damage bit.
FL_DAMAGE_ALL	Everything needs to be redrawn.

12.1.4.8 Fl_Event

enum [Fl_Event](#)

Every time a user moves the mouse pointer, clicks a button, or presses a key, an event is generated and sent to your application.

Events can also come from other programs like the window manager.

Events are identified by the integer argument passed to the [Fl_Widget::handle\(\)](#) virtual method. Other information about the most recent event is stored in static locations and acquired by calling the [Fl::event_*](#)() methods. This static information remains valid until the next event is read from the window system, so it is ok to look at it outside of the [handle\(\)](#) method.

Event numbers can be converted to their actual names using the [fl_eventnames\[\]](#) array defined in `#include <FL/names.h>`

See also

[Fl::event_text\(\)](#), [Fl::event_key\(\)](#), class [Fl::](#)

Enumerator

FL_NO_EVENT	No event.
FL_PUSH	A mouse button has gone down with the mouse pointing at this widget. You can find out what button by calling Fl::event_button(). You find out the mouse position by calling Fl::event_x() and Fl::event_y(). A widget indicates that it "wants" the mouse click by returning non-zero from its Fl_Widget::handle() method. It will then become the Fl::pushed() widget and will get FL_DRAG and the matching FL_RELEASE events. If Fl_Widget::handle() returns zero then FLTK will try sending the FL_PUSH to another widget.
FL_RELEASE	A mouse button has been released. You can find out what button by calling Fl::event_button(). In order to receive the FL_RELEASE event, the widget must return non-zero when handling FL_PUSH.
FL_ENTER	The mouse has been moved to point at this widget. This can be used for highlighting feedback. If a widget wants to highlight or otherwise track the mouse, it indicates this by returning non-zero from its handle() method. It then becomes the Fl::belowmouse() widget and will receive FL_MOVE and FL_LEAVE events.

Enumerator

FL_LEAVE	The mouse has moved out of the widget. In order to receive the FL_LEAVE event, the widget must return non-zero when handling FL_ENTER.
FL_DRAG	The mouse has moved with a button held down. The current button state is in Fl::event_state() . The mouse position is in Fl::event_x() and Fl::event_y() . In order to receive FL_DRAG events, the widget must return non-zero when handling FL_PUSH.
FL_FOCUS	This indicates an <i>attempt</i> to give a widget the keyboard focus. If a widget wants the focus, it should change itself to display the fact that it has the focus, and return non-zero from its <code>handle()</code> method. It then becomes the Fl::focus() widget and gets FL_KEYDOWN, FL_KEYUP, and FL_UNFOCUS events. The focus will change either because the window manager changed which window gets the focus, or because the user tried to navigate using tab, arrows, or other keys. You can check Fl::event_key() to figure out why it moved. For navigation it will be the key pressed and for interaction with the window manager it will be zero.
FL_UNFOCUS	This event is sent to the previous Fl::focus() widget when another widget gets the focus or the window loses focus.
FL_KEYDOWN	A key was pressed (FL_KEYDOWN) or released (FL_KEYUP). FL_KEYBOARD is a synonym for FL_KEYDOWN. The key can be found in Fl::event_key() . The text that the key should insert can be found with Fl::event_text() and its length is in Fl::event_length() . If you use the key <code>handle()</code> should return 1. If you return zero then FLTK assumes you ignored the key and will then attempt to send it to a parent widget. If none of them want it, it will change the event into a FL_SHORTCUT event. To receive FL_KEYBOARD events you must also respond to the FL_FOCUS and FL_UNFOCUS events. If you are writing a text-editing widget you may also want to call the Fl::compose() function to translate individual keystrokes into non-ASCII characters. FL_KEYUP events are sent to the widget that currently has focus. This is not necessarily the same widget that received the corresponding FL_KEYDOWN event because focus may have changed between events.
FL_KEYBOARD	Equivalent to FL_KEYDOWN. See also FL_KEYDOWN
FL_KEYUP	Key release event. See also FL_KEYDOWN
FL_CLOSE	The user clicked the close button of a window. This event is used internally only to trigger the callback of Fl_Window derived classed. The default callback closes the window calling Fl_Window::hide() .

Enumerator

FL_MOVE	The mouse has moved without any mouse buttons held down. This event is sent to the Fl::belowmouse() widget. In order to receive FL_MOVE events, the widget must return non-zero when handling FL_ENTER.
FL_SHORTCUT	If the Fl::focus() widget is zero or ignores an FL_KEYBOARD event then FLTK tries sending this event to every widget it can, until one of them returns non-zero. FL_SHORTCUT is first sent to the Fl::belowmouse() widget, then its parents and siblings, and eventually to every widget in the window, trying to find an object that returns non-zero. FLTK tries really hard to not to ignore any keystrokes! You can also make "global" shortcuts by using Fl::add_handler() . A global shortcut will work no matter what windows are displayed or which one has the focus.
FL_DEACTIVATE	This widget is no longer active, due to Fl_Widget::deactivate() being called on it or one of its parents. Fl_Widget::active() may still be true after this, the widget is only active if Fl_Widget::active() is true on it and all its parents (use Fl_Widget::active_r() to check this).
FL_ACTIVATE	This widget is now active, due to Fl_Widget::activate() being called on it or one of its parents.
FL_HIDE	This widget is no longer visible, due to Fl_Widget::hide() being called on it or one of its parents, or due to a parent window being minimized. Fl_Widget::visible() may still be true after this, but the widget is visible only if visible() is true for it and all its parents (use Fl_Widget::visible_r() to check this).
FL_SHOW	This widget is visible again, due to Fl_Widget::show() being called on it or one of its parents, or due to a parent window being restored. Child Fl_Windows respond to this by actually creating the window if not done already, so if you subclass a window, be sure to pass FL_SHOW to the base class Fl_Widget::handle() method!
FL_PASTE	You should get this event some time after you call Fl::paste() . The contents of Fl::event_text() is the text to insert and the number of characters is in Fl::event_length() .
FL_SELECTIONCLEAR	The Fl::selection_owner() will get this event before the selection is moved to another widget. This indicates that some other widget or program has claimed the selection. Motif programs used this to clear the selection indication. Most modern programs ignore this.
FL_MOUSEWHEEL	The user has moved the mouse wheel. The Fl::event_dx() and Fl::event_dy() methods can be used to find the amount to scroll horizontally and vertically.
FL_DND_ENTER	The mouse has been moved to point at this widget. A widget that is interested in receiving drag'n'drop data must return 1 to receive FL_DND_DRAG, FL_DND_LEAVE and FL_DND_RELEASE events.
FL_DND_DRAG	The mouse has been moved inside a widget while dragging data. A widget that is interested in receiving drag'n'drop data should indicate the possible drop position.
FL_DND_LEAVE	The mouse has moved out of the widget.
FL_DND_RELEASE	The user has released the mouse button dropping data into the widget. If the widget returns 1, it will receive the data in the immediately following FL_PASTE event.

Enumerator

FL_SCREEN_CONFIGURATION_CHANGED	The screen configuration (number, positions) was changed. Use Fl::add_handler() to be notified of this event.
FL_FULLSCREEN	The fullscreen state of the window has changed. This event is sent to the window's handle method.
FL_ZOOM_GESTURE	The user has made a zoom/pinch/magnification gesture (Mac OS platform only). The Fl::event_dy() method can be used to find magnification amount, Fl::event_x() and Fl::event_y() are set as well. This event is sent to the window's handle method.
FL_ZOOM_EVENT	A zoom event (ctrl+/-/0/ or cmd+/-/0/) was processed. Use Fl::add_handler() to be notified of this event.

12.1.4.9 Fl_Labeltype

enum [Fl_Labeltype](#)

The [labeltype\(\)](#) method sets the type of the label.

Note

The documented `enum Fl_Labeltype` contains some values (names) with leading underscores, e.g. `↔` `_FL_IMAGE_LABEL`. This is due to technical reasons - please use the same values (names) without the leading underscore in your code! Enum values with leading underscores are reserved for internal use and subject to change without notice!

The following standard label types are included:

Enumerator

FL_NORMAL_LABEL	draws the text (0)
FL_NO_LABEL	does nothing
_FL_SHADOW_LABEL	draws a drop shadow under the text
_FL_ENGRAVED_LABEL	draws edges as though the text is engraved
_FL_EMBOSSSED_LABEL	draws edges as though the text is raised
_FL_MULTI_LABEL	draws a composite label See also Fl_Multi_Label
_FL_ICON_LABEL	draws the icon associated with the text
_FL_IMAGE_LABEL	the label displays an "icon" based on a Fl_Image
FL_FREE_LABELTYPE	first free labeltype to use for creating own labeltypes

12.1.4.10 Fl_Orientation

enum [Fl_Orientation](#)

`Fl_Orientation` describes the orientation of a GUI element.

FLTK schemes can draw several graphical elements, for instance arrows, pointing at different directions. This enum defines the direction to use for drawing a particular GUI element.

The definition of this enum was chosen such that the enum value can be multiplied by 45 to get a rotation angle in degrees starting at the horizontal axis (0 = right, 1 = NE, 2 = up, ...) that can be used with [fl_rotate\(\)](#). Note: angle is counter-clockwise in degrees.

The 'unspecified' value `FL_ORIENT_NONE` shall be used for elements that would usually not be rotated, like the return arrow of the [Fl_Return_Button](#). It can still be used as an angle though since it is the same value as `FL_↔` `ORIENT_RIGHT` (0 degrees).

Note

This enum is not yet stable (as of FLTK 1.4.0) and may be changed without notice as necessary.

Since

1.4.0

Enumerator

FL_ORIENT_NONE	GUI element direction is unspecified.
FL_ORIENT_RIGHT	GUI element pointing right (0°)
FL_ORIENT_NE	GUI element pointing NE (45°)
FL_ORIENT_UP	GUI element pointing up (90°)
FL_ORIENT_NW	GUI element pointing NW (135°)
FL_ORIENT_LEFT	GUI element pointing left (180°)
FL_ORIENT_SW	GUI element pointing SW (225°)
FL_ORIENT_DOWN	GUI element pointing down (270°)
FL_ORIENT_SE	GUI element pointing SE (315°)

12.1.4.11 Fl_When

enum [Fl_When](#)

These constants determine when a callback is performed.

Fl_When is a bit field. Some values are merely shortcuts for common bit combinations. New flags may be added in the future, so it's important to mask the required bit when reading via `when()`.

Note

Some widgets may not fully support FL_WHEN_... flags.

See also

[Fl_Widget::when\(\)](#), [Fl::callback_reason\(\)](#), [Fl_Callback_Reason](#), [Fl_Widget::do_callback\(\)](#)

Enumerator

FL_WHEN_NEVER	Never call the callback.
FL_WHEN_CHANGED	Do the callback only when the widget value changes.
FL_WHEN_NOT_CHANGED	Do the callback whenever the user interacts with the widget.
FL_WHEN_RELEASE	Do the callback when the button or key is released and the value changes.
FL_WHEN_RELEASE_ALWAYS	Do the callback when the button or key is released, even if the value doesn't change.
FL_WHEN_ENTER_KEY	Do the callback when the user presses the ENTER key and the value changes.
FL_WHEN_ENTER_KEY_ALWAYS	Do the callback when the user presses the ENTER key, even if the value doesn't change.
FL_WHEN_ENTER_KEY_CHANGED	Do callbacks whether the value changed or not, and when the ENTER key is pressed.
FL_WHEN_CLOSED	Do the callback when a child of Fl_Tabs is closed.

12.1.5 Function Documentation

12.1.5.1 fl_box()

```
Fl_Boxtype fl_box (
    Fl_Boxtype b) [inline]
```

Get the filled version of a frame.

If no filled version of a given frame exists, the behavior of this function is undefined and some random box or frame is returned.

12.1.5.2 fl_color_cube()

```
Fl_Color fl_color_cube (
    int r,
    int g,
    int b) [inline]
```

Returns a color out of the color cube.

r must be in the range 0 to FL_NUM_RED (5) minus 1, *g* must be in the range 0 to FL_NUM_GREEN (8) minus 1, *b* must be in the range 0 to FL_NUM_BLUE (5) minus 1.

To get the closest color to a 8-bit set of R,G,B values use:

```
fl_color_cube(R * (FL_NUM_RED - 1) / 255,
    G * (FL_NUM_GREEN - 1) / 255,
    B * (FL_NUM_BLUE - 1) / 255);
```

12.1.5.3 fl_define_FL_EMBOSSED_LABEL()

```
Fl_Labeltype fl_define_FL_EMBOSSED_LABEL () [extern]
```

Initializes the internal table entry for FL_EMBOSSED_LABEL and returns its internal value.

Internal use only.

12.1.5.4 fl_define_FL_ENGRAVED_LABEL()

```
Fl_Labeltype fl_define_FL_ENGRAVED_LABEL () [extern]
```

Initializes the internal table entry for FL_ENGRAVED_LABEL and returns its internal value.

Internal use only.

12.1.5.5 fl_define_FL_ICON_LABEL()

```
Fl_Labeltype fl_define_FL_ICON_LABEL () [extern]
```

Initializes the internal table entry for FL_ICON_LABEL and returns its internal value.

Internal use only.

12.1.5.6 fl_define_FL_IMAGE_LABEL()

```
Fl_Labeltype fl_define_FL_IMAGE_LABEL () [extern]
```

Initializes the internal table entry for FL_IMAGE_LABEL and returns its internal value.

Internal use only.

12.1.5.7 fl_define_FL_MULTI_LABEL()

```
Fl_Labeltype fl_define_FL_MULTI_LABEL () [extern]
```

Initializes the internal table entry for FL_MULTI_LABEL and returns its internal value.

Internal use only.

12.1.5.8 fl_define_FL_SHADOW_LABEL()

```
Fl_Labeltype fl_define_FL_SHADOW_LABEL () [extern]
```

Initializes the internal table entry for FL_SHADOW_LABEL and returns its internal value.

Internal use only.

12.1.5.9 fl_down()

```
Fl_Boxtype fl_down (
    Fl_Boxtype b) [inline]
```

Get the "pressed" or "down" version of a box.

If no "down" version of a given box exists, the behavior of this function is undefined and some random box or frame is returned.

12.1.5.10 fl_frame()

```
Fl_Boxtype fl_frame (
    Fl_Boxtype b) [inline]
```

Get the unfilled, frame only version of a box.

If no frame version of a given box exists, the behavior of this function is undefined and some random box or frame is returned.

12.1.5.11 fl_gray_ramp()

```
Fl_Color fl_gray_ramp (
    int i) [inline]
```

Returns a gray color value from black (i == 0) to white (i == FL_NUM_GRAY - 1).

FL_NUM_GRAY is defined to be 24 in the current FLTK release. To get the closest FLTK gray value to an 8-bit grayscale color 'I' use:

```
fl_gray_ramp(I * (FL_NUM_GRAY - 1) / 255)
```

12.1.6 Variable Documentation

12.1.6.1 FL_ALIGN_LEFT

```
const Fl_Align FL_ALIGN_LEFT = 0x0004
```

Align the label at the left of the widget.

Inside labels appear left-justified starting at the left side of the widget, outside labels are right-justified and drawn to the left of the widget.

12.1.6.2 FL_ALIGN_TOP

```
const Fl_Align FL_ALIGN_TOP = 0x0001
```

Align the label at the top of the widget.

Inside labels appear below the top, outside labels are drawn on top of the widget.

12.1.6.3 FL_NORMAL_SIZE

```
Fl_Fontsize FL_NORMAL_SIZE [extern]
```

normal font size

normal font size

12.2 Enumerations.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Enumerations for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2025 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
```

```

00021 #ifndef FL_Enumerations_H
00022 #define FL_Enumerations_H
00023
00024 /*
00025 *****
00026 * FL_ABI_VERSION is defined by configure or CMake since FLTK 1.3.4.
00027 * It is written to FL/fl_config.h and #included here.
00028 *****
00029 * For more informations on FL_ABI_VERSION see README.abi-version.txt.
00030 *****
00031 */
00032
00033 #include <FL/fl_config.h>
00034
00035 # include "Fl_Export.H"
00036 # include "fl_types.h"
00037 # include <FL/platform_types.h> // for FL_COMMAND and FL_CONTROL
00038
00039 // Keep the following comment in sync with the values below for searching
00040 // Current FLTK version: 1.5.0
00041
00055 #define FL_MAJOR_VERSION      1
00056
00062 #define FL_MINOR_VERSION      5
00063
00069 #define FL_PATCH_VERSION      0
00070
00092 #define FL_VERSION            ( (double)FL_MAJOR_VERSION + \
00093                               (double)FL_MINOR_VERSION * 0.01 + \
00094                               (double)FL_PATCH_VERSION * 0.0001 )
00095
00114 #define FL_API_VERSION (FL_MAJOR_VERSION*10000 + FL_MINOR_VERSION*100 + FL_PATCH_VERSION)
00115
00146 #ifndef FL_ABI_VERSION
00147 #define FL_ABI_VERSION (FL_MAJOR_VERSION*10000 + FL_MINOR_VERSION*100)
00148 #endif
00149
00150 /*
00151  Check if FL_ABI_VERSION is out of allowed range; redefine if necessary.
00152
00153  This is done to prevent users from defining an illegal ABI version.
00154
00155  Rule: FL_MAJOR_VERSION * 10000 + FL_MINOR_VERSION * 100
00156        <= FL_ABI_VERSION <= FL_API_VERSION.
00157
00158  Example (FLTK 1.3.4):
00159
00160        10300 <= FL_ABI_VERSION <= 10304
00161
00162  Note: configure + CMake can be used to define FL_ABI_VERSION, but they
00163  do not check validity. This is done here.
00164 */
00165
00166 #if FL_ABI_VERSION < FL_MAJOR_VERSION*10000 + FL_MINOR_VERSION*100
00167
00168 # undef FL_ABI_VERSION
00169 # define FL_ABI_VERSION (FL_MAJOR_VERSION*10000 + FL_MINOR_VERSION*100)
00170
00171 #elif FL_ABI_VERSION > FL_API_VERSION
00172
00173 # undef FL_ABI_VERSION
00174 # define FL_ABI_VERSION FL_API_VERSION
00175
00176 #endif
00177
00178 // group: Version Numbers
00179
00198 // DEV NOTE: Keep this list in sync with FL/names.h
00199 enum Fl_Event { // events
00201     FL_NO_EVENT      = 0,
00202
00214     FL_PUSH          = 1,
00215
00222     FL_RELEASE       = 2,
00223
00231     FL_ENTER         = 3,
00232
00237     FL_LEAVE        = 4,
00238
00246     FL_DRAG          = 5,
00247
00261     FL_FOCUS         = 6,
00262
00266     FL_UNFOCUS      = 7,
00267
00288     FL_KEYDOWN       = 8,
00289

```

```

00293 FL_KEYBOARD          = 8,
00294
00298 FL_KEYUP            = 9,
00299
00305 FL_CLOSE              = 10,
00306
00313 FL_MOVE               = 11,
00314
00327 FL_SHORTCUT         = 12,
00328
00334 FL_DEACTIVATE        = 13,
00335
00339 FL_ACTIVATE          = 14,
00340
00347 FL_HIDE             = 15,
00348
00355 FL_SHOW               = 16,
00356
00361 FL_PASTE             = 17,
00362
00368 FL_SELECTIONCLEAR    = 18,
00369
00373 FL_MOUSEWHEEL        = 19,
00374
00379 FL_DND_ENTER         = 20,
00380
00385 FL_DND_DRAG          = 21,
00386
00389 FL_DND_LEAVE         = 22,
00390
00395 FL_DND_RELEASE       = 23,
00399 FL_SCREEN_CONFIGURATION_CHANGED = 24,
00403 FL_FULLSCREEN        = 25,
00409 FL_ZOOM_GESTURE       = 26,
00413 FL_ZOOM_EVENT         = 27
00414 // DEV NOTE: Keep this list in sync with FL/names.h
00415 };
00416
00429 enum Fl_When { // Fl_Widget::when():
00430     FL_WHEN_NEVER        = 0,
00431     FL_WHEN_CHANGED      = 1,
00432     FL_WHEN_NOT_CHANGED  = 2,
00433     FL_WHEN_RELEASE      = 4,
00434     FL_WHEN_RELEASE_ALWAYS = 6,
00435     FL_WHEN_ENTER_KEY    = 8,
00436     FL_WHEN_ENTER_KEY_ALWAYS = 10,
00437     FL_WHEN_ENTER_KEY_CHANGED = 11,
00438     FL_WHEN_CLOSED       = 16
00439 };
00440 // group: When Conditions
00441
00442
00449 enum Fl_Callback_Reason {
00450     FL_REASON_UNKNOWN=0,
00451     FL_REASON_SELECTED,
00452     FL_REASON_DESELECTED,
00453     FL_REASON_RESELECTED,
00454     FL_REASON_OPENED,
00455     FL_REASON_CLOSED,
00456     FL_REASON_DRAGGED,
00457     FL_REASON_CANCELLED,
00458     FL_REASON_CHANGED,
00459     FL_REASON_GOT_FOCUS,
00460     FL_REASON_LOST_FOCUS,
00461     FL_REASON_RELEASED,
00462     FL_REASON_ENTER_KEY,
00463     FL_REASON_USER = 32
00464 };
00465 // group: Callback Reasons
00466
00467
00478 // FIXME: These codes collide with valid Unicode keys
00479
00480 #define FL_Button          0xfee8
00481 #define FL_BackSpace      0xff08
00482 #define FL_Tab            0xff09
00483 #define FL_Iso_Key        0xff0c
00484 #define FL_Enter          0xff0d
00485 #define FL_Pause          0xff13
00486 #define FL_Scroll_Lock    0xff14
00487 #define FL_Escape         0xff1b
00488 #define FL_Kana           0xff2e
00489 #define FL_Eisu           0xff2f
00490 #define FL_Yen            0xff30
00491 #define FL_JIS_Underscore 0xff31
00492 #define FL_Home           0xff50
00493 #define FL_Left           0xff51

```

```
00494 #define FL_Up          0xff52
00495 #define FL_Right        0xff53
00496 #define FL_Down        0xff54
00497 #define FL_Page_Up      0xff55
00498 #define FL_Page_Down    0xff56
00499 #define FL_End         0xff57
00500 #define FL_Print        0xff61
00501 #define FL_Insert       0xff63
00502 #define FL_Menu         0xff67
00503 #define FL_Help         0xff68
00504 #define FL_Num_Lock     0xff7f
00505 #define FL_KP           0xff80
00506 #define FL_KP_Enter     0xff8d
00507 #define FL_KP_Last      0xffbd
00508 #define FL_F            0xffbd
00509 #define FL_F_Last       0xfe0
00510 #define FL_Shift_L      0xfe1
00511 #define FL_Shift_R      0xfe2
00512 #define FL_Control_L    0xfe3
00513 #define FL_Control_R    0xfe4
00514 #define FL_Caps_Lock    0xfe5
00515 #define FL_Meta_L       0xfe7
00516 #define FL_Meta_R       0xfe8
00517 #define FL_Alt_L        0xfe9
00518 #define FL_Alt_R        0xfea
00519 #define FL_Delete       0xffff
00520 #define FL_Alt_Gr       0xfe03
00521
00522 // These use the Private Use Area (PUA) of the Basic Multilingual Plane
00523 // of Unicode. Guaranteed not to conflict with a proper Unicode character.
00524
00525 // These primarily map to the XFree86 keysym range
00526 #define FL_Volume_Down  0xEF11
00527 #define FL_Volume_Mute  0xEF12
00528 #define FL_Volume_Up    0xEF13
00529 #define FL_Media_Play   0xEF14
00530 #define FL_Media_Stop   0xEF15
00531 #define FL_Media_Prev   0xEF16
00532 #define FL_Media_Next   0xEF17
00533 #define FL_Home_Page    0xEF18
00534 #define FL_Mail         0xEF19
00535 #define FL_Search       0xEF1B
00536 #define FL_Back         0xEF26
00537 #define FL_Forward      0xEF27
00538 #define FL_Stop         0xEF28
00539 #define FL_Refresh      0xEF29
00540 #define FL_Sleep        0xEF2F
00541 #define FL_Favorites    0xEF30
00542
00543 // group: Non-ASCII key names
00544
00554 #define FL_LEFT_MOUSE   1
00555 #define FL_MIDDLE_MOUSE 2
00556 #define FL_RIGHT_MOUSE  3
00557 #define FL_BACK_MOUSE   4
00558 #define FL_FORWARD_MOUSE 5
00559
00560 // group: Mouse Buttons
00561
00562
00563 // group: Event States
00564
00565
00570 // FIXME: it would be nice to have the modifiers in the upper 8 bits so that
00571 // a unicode key (21 bits) can be sent as an unsigned with the modifiers.
00572
00573 #define FL_SHIFT         0x00010000
00574 #define FL_CAPS_LOCK     0x00020000
00575 #define FL_CTRL          0x00040000
00576 #define FL_ALT           0x00080000
00577 #define FL_NUM_LOCK      0x00100000
00578
00579 #define FL_META          0x00400000 // most X servers do this?
00580
00581 #define FL_SCROLL_LOCK   0x00800000 // correct for XFree86
00582
00583 // Mouse buttons // correct for XFree86
00584
00585 #define FL_BUTTON1       0x01000000
00586 #define FL_BUTTON2       0x02000000
00587 #define FL_BUTTON3       0x04000000
00588 #define FL_BUTTON4       0x08000000
00589 #define FL_BUTTON5       0x10000000
00590 #define FL_BUTTONS       0x1f000000
00591
00592 #define FL_BUTTON(n)     (0x00800000«(n))
00593
00594
00595 #define FL_KEY_MASK      0x0000ffff
```

```
00596                                     //  FIXME: Unicode needs 21 bits!
00597
00598     // group: Event States
00599
00625     // group: Box Types
00626
00638     enum Fl_Boxtype { // boxtypes (if you change these you must also change fl_boxtype.cxx):
00639
00640         FL_NO_BOX = 0,
00641         FL_FLAT_BOX,
00642         FL_UP_BOX,
00643         FL_DOWN_BOX,
00644         FL_UP_FRAME,
00645         FL_DOWN_FRAME,
00646         FL_THIN_UP_BOX,
00647         FL_THIN_DOWN_BOX,
00648         FL_THIN_UP_FRAME,
00649         FL_THIN_DOWN_FRAME,
00650         FL_ENGRAVED_BOX,
00651         FL_EMBOSSSED_BOX,
00652         FL_ENGRAVED_FRAME,
00653         FL_EMBOSSSED_FRAME,
00654         FL_BORDER_BOX,
00655         _FL_SHADOW_BOX,
00656         FL_BORDER_FRAME,
00657         _FL_SHADOW_FRAME,
00658         _FL_ROUNDED_BOX,
00659         _FL_RSHADOW_BOX,
00660         _FL_ROUNDED_FRAME,
00661         _FL_RFLAT_BOX,
00662         _FL_ROUND_UP_BOX,
00663         _FL_ROUND_DOWN_BOX,
00664         _FL_DIAMOND_UP_BOX,
00665         _FL_DIAMOND_DOWN_BOX,
00666         _FL_OVAL_BOX,
00667         _FL_OSHADOW_BOX,
00668         _FL_OVAL_FRAME,
00669         _FL_OFLAT_BOX,
00670         _FL_PLASTIC_UP_BOX,
00671         _FL_PLASTIC_DOWN_BOX,
00672         _FL_PLASTIC_UP_FRAME,
00673         _FL_PLASTIC_DOWN_FRAME,
00674         _FL_PLASTIC_THIN_UP_BOX,
00675         _FL_PLASTIC_THIN_DOWN_BOX,
00676         _FL_PLASTIC_ROUND_UP_BOX,
00677         _FL_PLASTIC_ROUND_DOWN_BOX,
00678         _FL_GTK_UP_BOX,
00679         _FL_GTK_DOWN_BOX,
00680         _FL_GTK_UP_FRAME,
00681         _FL_GTK_DOWN_FRAME,
00682         _FL_GTK_THIN_UP_BOX,
00683         _FL_GTK_THIN_DOWN_BOX,
00684         _FL_GTK_THIN_UP_FRAME,
00685         _FL_GTK_THIN_DOWN_FRAME,
00686         _FL_GTK_ROUND_UP_BOX,
00687         _FL_GTK_ROUND_DOWN_BOX,
00688         _FL_GLEAM_UP_BOX,
00689         _FL_GLEAM_DOWN_BOX,
00690         _FL_GLEAM_UP_FRAME,
00691         _FL_GLEAM_DOWN_FRAME,
00692         _FL_GLEAM_THIN_UP_BOX,
00693         _FL_GLEAM_THIN_DOWN_BOX,
00694         _FL_GLEAM_ROUND_UP_BOX,
00695         _FL_GLEAM_ROUND_DOWN_BOX,
00696         _FL_OXY_UP_BOX,
00697         _FL_OXY_DOWN_BOX,
00698         _FL_OXY_UP_FRAME,
00699         _FL_OXY_DOWN_FRAME,
00700         _FL_OXY_THIN_UP_BOX,
00701         _FL_OXY_THIN_DOWN_BOX,
00702         _FL_OXY_THIN_UP_FRAME,
00703         _FL_OXY_THIN_DOWN_FRAME,
00704         _FL_OXY_ROUND_UP_BOX,
00705         _FL_OXY_ROUND_DOWN_BOX,
00706         _FL_OXY_BUTTON_UP_BOX,
00707         _FL_OXY_BUTTON_DOWN_BOX,
00708         FL_FREE_BOXTYPE,
00709         FL_MAX_BOXTYPE = 255
00710     };
00711
00712     #ifndef FL_DOXYGEN
00713
00714     extern FL_EXPORT Fl_Boxtype fl_define_FL_ROUND_UP_BOX();
00715     #define FL_ROUND_UP_BOX fl_define_FL_ROUND_UP_BOX()
00716     #define FL_ROUND_DOWN_BOX (Fl_Boxtype) (fl_define_FL_ROUND_UP_BOX()+1)
00717     extern FL_EXPORT Fl_Boxtype fl_define_FL_SHADOW_BOX();
00718     #define FL_SHADOW_BOX fl_define_FL_SHADOW_BOX()
```

```

00719 #define FL_SHADOW_FRAME (Fl_Boxtype)(fl_define_FL_SHADOW_BOX()+2)
00720 extern FL_EXPORT Fl_Boxtype fl_define_FL_ROUNDED_BOX();
00721 #define FL_ROUNDED_BOX fl_define_FL_ROUNDED_BOX()
00722 #define FL_ROUNDED_FRAME (Fl_Boxtype)(fl_define_FL_ROUNDED_BOX()+2)
00723 extern FL_EXPORT Fl_Boxtype fl_define_FL_RFLAT_BOX();
00724 #define FL_RFLAT_BOX fl_define_FL_RFLAT_BOX()
00725 extern FL_EXPORT Fl_Boxtype fl_define_FL_RSHADOW_BOX();
00726 #define FL_RSHADOW_BOX fl_define_FL_RSHADOW_BOX()
00727 extern FL_EXPORT Fl_Boxtype fl_define_FL_DIAMOND_BOX();
00728 #define FL_DIAMOND_UP_BOX fl_define_FL_DIAMOND_BOX()
00729 #define FL_DIAMOND_DOWN_BOX (Fl_Boxtype)(fl_define_FL_DIAMOND_BOX()+1)
00730 extern FL_EXPORT Fl_Boxtype fl_define_FL_OVAL_BOX();
00731 #define FL_OVAL_BOX fl_define_FL_OVAL_BOX()
00732 #define FL_OSHADOW_BOX (Fl_Boxtype)(fl_define_FL_OVAL_BOX()+1)
00733 #define FL_OVAL_FRAME (Fl_Boxtype)(fl_define_FL_OVAL_BOX()+2)
00734 #define FL_OFLAT_BOX (Fl_Boxtype)(fl_define_FL_OVAL_BOX()+3)
00735
00736 extern FL_EXPORT Fl_Boxtype fl_define_FL_PLASTIC_UP_BOX();
00737 #define FL_PLASTIC_UP_BOX fl_define_FL_PLASTIC_UP_BOX()
00738 #define FL_PLASTIC_DOWN_BOX (Fl_Boxtype)(fl_define_FL_PLASTIC_UP_BOX()+1)
00739 #define FL_PLASTIC_UP_FRAME (Fl_Boxtype)(fl_define_FL_PLASTIC_UP_BOX()+2)
00740 #define FL_PLASTIC_DOWN_FRAME (Fl_Boxtype)(fl_define_FL_PLASTIC_UP_BOX()+3)
00741 #define FL_PLASTIC_THIN_UP_BOX (Fl_Boxtype)(fl_define_FL_PLASTIC_UP_BOX()+4)
00742 #define FL_PLASTIC_THIN_DOWN_BOX (Fl_Boxtype)(fl_define_FL_PLASTIC_UP_BOX()+5)
00743 #define FL_PLASTIC_ROUND_UP_BOX (Fl_Boxtype)(fl_define_FL_PLASTIC_UP_BOX()+6)
00744 #define FL_PLASTIC_ROUND_DOWN_BOX (Fl_Boxtype)(fl_define_FL_PLASTIC_UP_BOX()+7)
00745
00746 extern FL_EXPORT Fl_Boxtype fl_define_FL_GTK_UP_BOX();
00747 #define FL_GTK_UP_BOX fl_define_FL_GTK_UP_BOX()
00748 #define FL_GTK_DOWN_BOX (Fl_Boxtype)(fl_define_FL_GTK_UP_BOX()+1)
00749 #define FL_GTK_UP_FRAME (Fl_Boxtype)(fl_define_FL_GTK_UP_BOX()+2)
00750 #define FL_GTK_DOWN_FRAME (Fl_Boxtype)(fl_define_FL_GTK_UP_BOX()+3)
00751 #define FL_GTK_THIN_UP_BOX (Fl_Boxtype)(fl_define_FL_GTK_UP_BOX()+4)
00752 #define FL_GTK_THIN_DOWN_BOX (Fl_Boxtype)(fl_define_FL_GTK_UP_BOX()+5)
00753 #define FL_GTK_THIN_UP_FRAME (Fl_Boxtype)(fl_define_FL_GTK_UP_BOX()+6)
00754 #define FL_GTK_THIN_DOWN_FRAME (Fl_Boxtype)(fl_define_FL_GTK_UP_BOX()+7)
00755 #define FL_GTK_ROUND_UP_BOX (Fl_Boxtype)(fl_define_FL_GTK_UP_BOX()+8)
00756 #define FL_GTK_ROUND_DOWN_BOX (Fl_Boxtype)(fl_define_FL_GTK_UP_BOX()+9)
00757
00758 extern FL_EXPORT Fl_Boxtype fl_define_FL_GLEAM_UP_BOX();
00759 #define FL_GLEAM_UP_BOX fl_define_FL_GLEAM_UP_BOX()
00760 #define FL_GLEAM_DOWN_BOX (Fl_Boxtype)(fl_define_FL_GLEAM_UP_BOX()+1)
00761 #define FL_GLEAM_UP_FRAME (Fl_Boxtype)(fl_define_FL_GLEAM_UP_BOX()+2)
00762 #define FL_GLEAM_DOWN_FRAME (Fl_Boxtype)(fl_define_FL_GLEAM_UP_BOX()+3)
00763 #define FL_GLEAM_THIN_UP_BOX (Fl_Boxtype)(fl_define_FL_GLEAM_UP_BOX()+4)
00764 #define FL_GLEAM_THIN_DOWN_BOX (Fl_Boxtype)(fl_define_FL_GLEAM_UP_BOX()+5)
00765 #define FL_GLEAM_ROUND_UP_BOX (Fl_Boxtype)(fl_define_FL_GLEAM_UP_BOX()+6)
00766 #define FL_GLEAM_ROUND_DOWN_BOX (Fl_Boxtype)(fl_define_FL_GLEAM_UP_BOX()+7)
00767
00768 extern FL_EXPORT Fl_Boxtype fl_define_FL_OXY_UP_BOX();
00769 #define FL_OXY_UP_BOX fl_define_FL_OXY_UP_BOX()
00770 #define FL_OXY_DOWN_BOX (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+1)
00771 #define FL_OXY_UP_FRAME (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+2)
00772 #define FL_OXY_DOWN_FRAME (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+3)
00773 #define FL_OXY_THIN_UP_BOX (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+4)
00774 #define FL_OXY_THIN_DOWN_BOX (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+5)
00775 #define FL_OXY_THIN_UP_FRAME (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+6)
00776 #define FL_OXY_THIN_DOWN_FRAME (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+7)
00777 #define FL_OXY_ROUND_UP_BOX (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+8)
00778 #define FL_OXY_ROUND_DOWN_BOX (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+9)
00779 #define FL_OXY_BUTTON_UP_BOX (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+10)
00780 #define FL_OXY_BUTTON_DOWN_BOX (Fl_Boxtype)(fl_define_FL_OXY_UP_BOX()+11)
00781
00782 #endif // ! FL_DOXYGEN
00783
00784 // conversions of box types to other boxtypes:
00790 inline Fl_Boxtype fl_box(Fl_Boxtype b) {
00791     return (Fl_Boxtype)((b<FL_UP_BOX)?b:(b-2));
00792 }
00798 inline Fl_Boxtype fl_down(Fl_Boxtype b) {
00799     return (Fl_Boxtype)((b<FL_UP_BOX)?b:(b-1));
00800 }
00806 inline Fl_Boxtype fl_frame(Fl_Boxtype b) {
00807     return (Fl_Boxtype)((b%4<2)?b:(b+2));
00808 }
00809
00810 // back-compatibility box types:
00811 #define FL_FRAME FL_ENGRAVED_FRAME
00812 #define FL_FRAME_BOX FL_ENGRAVED_BOX
00813 #define FL_CIRCLE_BOX FL_ROUND_DOWN_BOX
00814 #define FL_DIAMOND_BOX FL_DIAMOND_DOWN_BOX
00815
00816 // group: Box Types
00817
00829 enum Fl_Labeltype { // labeltypes:
00830     FL_NORMAL_LABEL = 0,
00831     FL_NO_LABEL,

```

```
00832     _FL_SHADOW_LABEL,  
00833     _FL_ENGRAVED_LABEL,  
00834     _FL_EMBOSSED_LABEL,  
00835     _FL_MULTI_LABEL,  
00836     _FL_ICON_LABEL,  
00837     _FL_IMAGE_LABEL,  
00838  
00839     FL_FREE_LABELTYPE  
00840 };  
00841  
00853 #define FL_SYMBOL_LABEL FL_NORMAL_LABEL  
00858 extern Fl_Labeltype FL_EXPORT fl_define_FL_SHADOW_LABEL();  
00862 #define FL_SHADOW_LABEL fl_define_FL_SHADOW_LABEL()  
00863  
00868 extern Fl_Labeltype FL_EXPORT fl_define_FL_ENGRAVED_LABEL();  
00872 #define FL_ENGRAVED_LABEL fl_define_FL_ENGRAVED_LABEL()  
00873  
00878 extern Fl_Labeltype FL_EXPORT fl_define_FL_EMBOSSED_LABEL();  
00882 #define FL_EMBOSSED_LABEL fl_define_FL_EMBOSSED_LABEL()  
00883  
00888 extern Fl_Labeltype FL_EXPORT fl_define_FL_MULTI_LABEL();  
00893 #define FL_MULTI_LABEL fl_define_FL_MULTI_LABEL()  
00894  
00899 extern Fl_Labeltype FL_EXPORT fl_define_FL_ICON_LABEL();  
00903 #define FL_ICON_LABEL fl_define_FL_ICON_LABEL()  
00904  
00909 extern Fl_Labeltype FL_EXPORT fl_define_FL_IMAGE_LABEL();  
00916 #define FL_IMAGE_LABEL fl_define_FL_IMAGE_LABEL()  
00917  
00980 typedef unsigned Fl_Align;  
00981  
00983 const Fl_Align FL_ALIGN_CENTER      = 0x0000;  
00984  
00987 const Fl_Align FL_ALIGN_TOP        = 0x0001;  
00988  
00990 const Fl_Align FL_ALIGN_BOTTOM     = 0x0002;  
00991  
00995 const Fl_Align FL_ALIGN_LEFT       = 0x0004;  
00996  
00998 const Fl_Align FL_ALIGN_RIGHT      = 0x0008;  
00999  
01001 const Fl_Align FL_ALIGN_INSIDE     = 0x0010;  
01002  
01004 const Fl_Align FL_ALIGN_TEXT_OVER_IMAGE = 0x0020;  
01005  
01007 const Fl_Align FL_ALIGN_IMAGE_OVER_TEXT = 0x0000;  
01008  
01010 const Fl_Align FL_ALIGN_CLIP       = 0x0040;  
01011  
01013 const Fl_Align FL_ALIGN_WRAP        = 0x0080;  
01014  
01016 const Fl_Align FL_ALIGN_IMAGE_NEXT_TO_TEXT = 0x0100;  
01017  
01019 const Fl_Align FL_ALIGN_TEXT_NEXT_TO_IMAGE = 0x0120;  
01020  
01022 const Fl_Align FL_ALIGN_IMAGE_BACKDROP = 0x0200;  
01023  
01024 const Fl_Align FL_ALIGN_TOP_LEFT    = FL_ALIGN_TOP | FL_ALIGN_LEFT;  
01025 const Fl_Align FL_ALIGN_TOP_RIGHT   = FL_ALIGN_TOP | FL_ALIGN_RIGHT;  
01026 const Fl_Align FL_ALIGN_BOTTOM_LEFT = FL_ALIGN_BOTTOM | FL_ALIGN_LEFT;  
01027 const Fl_Align FL_ALIGN_BOTTOM_RIGHT = FL_ALIGN_BOTTOM | FL_ALIGN_RIGHT;  
01028  
01030 const Fl_Align FL_ALIGN_LEFT_TOP    = 0x0007;  
01031  
01033 const Fl_Align FL_ALIGN_RIGHT_TOP   = 0x000b;  
01034  
01036 const Fl_Align FL_ALIGN_LEFT_BOTTOM = 0x000d;  
01037  
01039 const Fl_Align FL_ALIGN_RIGHT_BOTTOM = 0x000e;  
01040  
01042 const Fl_Align FL_ALIGN_NOWRAP      = 0x0000;  
01043  
01045 const Fl_Align FL_ALIGN_POSITION_MASK = 0x000f;  
01046  
01048 const Fl_Align FL_ALIGN_IMAGE_MASK   = 0x0320;  
01057 typedef int Fl_Font;  
01058  
01059 const Fl_Font FL_HELVETICA           = 0;  
01060 const Fl_Font FL_HELVETICA_BOLD      = 1;  
01061 const Fl_Font FL_HELVETICA_ITALIC    = 2;  
01062 const Fl_Font FL_HELVETICA_BOLD_ITALIC = 3;  
01063 const Fl_Font FL_COURIER             = 4;  
01064 const Fl_Font FL_COURIER_BOLD        = 5;  
01065 const Fl_Font FL_COURIER_ITALIC      = 6;  
01066 const Fl_Font FL_COURIER_BOLD_ITALIC = 7;  
01067 const Fl_Font FL_TIMES               = 8;  
01068 const Fl_Font FL_TIMES_BOLD          = 9;
```



```

01069 const Fl_Font FL_TIMES_ITALIC          = 10;
01070 const Fl_Font FL_TIMES_BOLD_ITALIC       = 11;
01071 const Fl_Font FL_SYMBOL                  = 12;
01072 const Fl_Font FL_SCREEN                   = 13;
01073 const Fl_Font FL_SCREEN_BOLD              = 14;
01074 const Fl_Font FL_ZAPF_DINGBATS            = 15;
01075
01076 const Fl_Font FL_FREE_FONT                = 16;
01077 const Fl_Font FL_BOLD                     = 1;
01078 const Fl_Font FL_ITALIC                    = 2;
01079 const Fl_Font FL_BOLD_ITALIC               = 3;
01080
01086 typedef int Fl_Fontsize;
01087
01088 extern FL_EXPORT Fl_Fontsize FL_NORMAL_SIZE;
01089
01114 typedef unsigned int Fl_Color;
01115
01116 // Standard colors. These are used as default colors in widgets and altered as necessary
01117 const Fl_Color FL_FOREGROUND_COLOR = 0;
01118 const Fl_Color FL_BACKGROUND2_COLOR = 7;
01119 const Fl_Color FL_INACTIVE_COLOR   = 8;
01120 const Fl_Color FL_SELECTION_COLOR   = 15;
01121
01122 // boxtypes generally limit themselves to these colors so
01123 // the whole ramp is not allocated:
01124
01125 const Fl_Color FL_GRAY0   = 32;           // 'A'
01126 const Fl_Color FL_DARK3   = 39;           // 'H'
01127 const Fl_Color FL_DARK2   = 45;           // 'N'
01128 const Fl_Color FL_DARK1   = 47;           // 'P'
01130 const Fl_Color FL_BACKGROUND_COLOR = 49;   // 'R' default background color
01131 const Fl_Color FL_LIGHT1   = 50;           // 'S'
01132 const Fl_Color FL_LIGHT2   = 52;           // 'U'
01133 const Fl_Color FL_LIGHT3   = 54;           // 'W'
01134
01135 // FLTK provides a 5x8x5 color cube that is used with colormap visuals
01136
01137 const Fl_Color FL_BLACK    = 56;
01138 const Fl_Color FL_RED      = 88;
01139 const Fl_Color FL_GREEN    = 63;
01140 const Fl_Color FL_YELLOW   = 95;
01141 const Fl_Color FL_BLUE     = 216;
01142 const Fl_Color FL_MAGENTA  = 248;
01143 const Fl_Color FL_CYAN     = 223;
01144 const Fl_Color FL_DARK_RED = 72;
01145
01146 const Fl_Color FL_DARK_GREEN = 60;
01147 const Fl_Color FL_DARK_YELLOW = 76;
01148 const Fl_Color FL_DARK_BLUE  = 136;
01149 const Fl_Color FL_DARK_MAGENTA = 152;
01150 const Fl_Color FL_DARK_CYAN  = 140;
01151
01152 const Fl_Color FL_WHITE      = 255;
01153
01156 #define FL_FREE_COLOR      (Fl_Color)16
01157 #define FL_NUM_FREE_COLOR  16
01158 #define FL_GRAY_RAMP       (Fl_Color)32
01159 #define FL_NUM_GRAY        24
01160 #define FL_GRAY             FL_BACKGROUND_COLOR
01161 #define FL_COLOR_CUBE      (Fl_Color)56
01162 #define FL_NUM_RED          5
01163 #define FL_NUM_GREEN        8
01164 #define FL_NUM_BLUE         5
01165
01166 FL_EXPORT Fl_Color fl_inactive(Fl_Color c);
01167
01190 typedef Fl_Color (Fl_Contrast_Function)(Fl_Color, Fl_Color, int, int);
01191
01192 FL_EXPORT void fl_contrast_function(Fl_Contrast_Function *f);
01193
01197 enum Fl_Contrast_Mode {
01198     FL_CONTRAST_NONE = 0,
01199     FL_CONTRAST_LEGACY,
01200     FL_CONTRAST_CIELAB,
01201     FL_CONTRAST_CUSTOM,
01202     FL_CONTRAST_LAST
01203 };
01204
01205 // The following functions are defined and documented in src/fl_contrast.cxx
01206
01207 FL_EXPORT void fl_contrast_level(int level);
01208 FL_EXPORT int fl_contrast_level();
01209 FL_EXPORT void fl_contrast_mode(int mode);
01210 FL_EXPORT int fl_contrast_mode();
01211
01212 FL_EXPORT Fl_Color fl_contrast(Fl_Color fg, Fl_Color bg, int context = 0, int size = 0);

```

```

01213
01214 FL_EXPORT double fl_lightness(Fl_Color color);
01215 FL_EXPORT double fl_luminance(Fl_Color color);
01216
01217 // Other color functions are defined and documented in src/fl_color.cxx
01218
01219 FL_EXPORT Fl_Color fl_color_average(Fl_Color c1, Fl_Color c2, float weight);
01220
01221 inline Fl_Color fl_lighter(Fl_Color c) { return fl_color_average(c, FL_WHITE, .67f); }
01222
01223 inline Fl_Color fl_darker(Fl_Color c) { return fl_color_average(c, FL_BLACK, .67f); }
01224
01225 inline Fl_Color fl_rgb_color(uchar r, uchar g, uchar b) {
01226     if (!r && !g && !b) return FL_BLACK;
01227     else return (Fl_Color)((((r < 8) | g) < 8) | b) < 8);
01228 }
01229
01230 inline Fl_Color fl_rgb_color(uchar g) {
01231     if (!g) return FL_BLACK;
01232     else return (Fl_Color)((((g < 8) | g) < 8) | g) < 8);
01233 }
01234
01235 inline Fl_Color fl_gray_ramp(int i) {return (Fl_Color)(i+FL_GRAY_RAMP);}
01236
01237 inline Fl_Color fl_color_cube(int r, int g, int b) {
01238     return (Fl_Color)((b*FL_NUM_RED + r) * FL_NUM_GREEN + g + FL_COLOR_CUBE);}
01239
01240 // group: Colors
01241
01242 /* FIXME: We should renumber these, but that will break the ABI */
01243 enum Fl_Cursor {
01244     FL_CURSOR_DEFAULT = 0, // U+2196
01245     FL_CURSOR_ARROW = 35, // U+2196
01246     FL_CURSOR_CROSS = 66, // U+FF0B
01247     FL_CURSOR_WAIT = 76, // U+231A, U+231B
01248     FL_CURSOR_INSERT = 77, // U+2336
01249     FL_CURSOR_HAND = 31, // U+261C
01250     FL_CURSOR_HELP = 47,
01251     FL_CURSOR_MOVE = 27, // U+2725, U+270B
01252
01253     /* Resize indicators */
01254     FL_CURSOR_NS = 78, // U+21D5
01255     FL_CURSOR_WE = 79, // U+21D4
01256     FL_CURSOR_NWSE = 80, // U+2921
01257     FL_CURSOR_NESW = 81, // U+2922
01258     FL_CURSOR_N = 70, // U+2912
01259     FL_CURSOR_NE = 69, // U+2197
01260     FL_CURSOR_E = 49, // U+21E5
01261     FL_CURSOR_SE = 8, // U+21F2
01262     FL_CURSOR_S = 9, // U+2913
01263     FL_CURSOR_SW = 7, // U+2199
01264     FL_CURSOR_W = 36, // U+21E4
01265     FL_CURSOR_NW = 68, // U+21F1
01266
01267     FL_CURSOR_NONE = 255
01268 };
01269 // group: Cursors
01270
01271 enum { // values for "when" passed to Fl::add_fd()
01272     FL_READ = 1,
01273     FL_WRITE = 4,
01274     FL_EXCEPT = 8
01275 };
01276
01277 enum Fl_Mode {
01278     FL_RGB = 0,
01279     FL_INDEX = 1,
01280     FL_SINGLE = 0,
01281     FL_DOUBLE = 2,
01282     FL_ACCUM = 4,
01283     FL_ALPHA = 8,
01284     FL_DEPTH = 16,
01285     FL_STENCIL = 32,
01286     FL_RGB8 = 64,
01287     FL_MULTISAMPLE = 128,
01288     FL_STEREO = 256,
01289     FL_FAKE_SINGLE = 512, // Fake single buffered windows using double-buffer
01290     FL_OPENGL3 = 1024
01291 };
01292
01293 // image alpha blending
01294
01295 #define FL_IMAGE_WITH_ALPHA 0x40000000
01296
01297 enum Fl_Damage {
01298     FL_DAMAGE_CHILD = 0x01,
01299     FL_DAMAGE_EXPOSE = 0x02,

```

```

01337     FL_DAMAGE_SCROLL    = 0x04,
01338     FL_DAMAGE_OVERLAY   = 0x08,
01339     FL_DAMAGE_USER1     = 0x10,
01340     FL_DAMAGE_USER2     = 0x20,
01341     FL_DAMAGE_ALL       = 0x80
01342 };
01343
01344 // FLTK 1.0.x compatibility definitions (FLTK_1_0_COMPAT) dropped in 1.4.0
01345
01358 enum Fl_Arrow_Type {
01359     FL_ARROW_SINGLE = 0x01,
01360     FL_ARROW_DOUBLE = 0x02,
01361     FL_ARROW_CHOICE = 0x03,
01362     FL_ARROW_RETURN = 0x04
01363 };
01364
01387 enum Fl_Orientation {
01388     FL_ORIENT_NONE = 0x00,
01389     FL_ORIENT_RIGHT = 0x00,
01390     FL_ORIENT_NE = 0x01,
01391     FL_ORIENT_UP = 0x02,
01392     FL_ORIENT_NW = 0x03,
01393     FL_ORIENT_LEFT = 0x04,
01394     FL_ORIENT_SW = 0x05,
01395     FL_ORIENT_DOWN = 0x06,
01396     FL_ORIENT_SE = 0x07
01397 };
01398
01399 #endif

```

12.3 filename.H File Reference

File names and URI utility functions.

```

#include "Fl_Export.H"
#include <FL/platform_types.h>
#include <string>

```

Macros

- `#define FL_PATH_MAX 2048`
all path buffers should use this length

Typedefs

- `typedef int Fl_File_Sort_F(struct dirent **, struct dirent **)`
File sorting function.

Functions

- `void fl_decode_uri (char *uri)`
Decodes a URL-encoded string.
- `int fl_filename_absolute (char *to, int tolen, const char *from)`
Makes a filename absolute from a relative filename to the current working directory.
- `int fl_filename_absolute (char *to, int tolen, const char *from, const char *cwd)`
*Concatenate the absolute path *base* with *from* to form the new absolute path in *to*.*
- `std::string fl_filename_absolute_str (const std::string &from)`
Makes a filename absolute from a filename relative to the current working directory.
- `std::string fl_filename_absolute_str (const std::string &from, const std::string &base)`
*Append the relative filename *from* to the absolute filename *base* to form the new absolute path.*
- `int fl_filename_expand (char *to, int tolen, const char *from)`
Expands a filename containing shell variables and tilde (~).
- `std::string fl_filename_expand_str (const std::string &from)`
Expands a filename containing shell variables and tilde (~).
- `const char * fl_filename_ext (const char *buf)`

- Gets the extension of a filename.*
- `std::string fl_filename_ext_str` (const std::string &filename)
Return a new string that contains the filename extension.
- `void fl_filename_free_list` (struct dirent ***l, int n)
Free the list of filenames that is generated by `fl_filename_list()`.
- `int fl_filename_isdir` (const char *name)
Determines if a file exists and is a directory from its filename.
- `int fl_filename_list` (const char *d, struct dirent ***l, `Fl_File_Sort_F` *s=`fl_numeric_sort`)
Portable and const-correct wrapper for the `scandir()` function.
- `int fl_filename_match` (const char *name, const char *pattern)
Checks if a string `s` matches a pattern `p`.
- `const char * fl_filename_name` (const char *filename)
Gets the file name from a path.
- `std::string fl_filename_name_str` (const std::string &filename)
Return a new string that contains the name part of the filename.
- `std::string fl_filename_path_str` (const std::string &filename)
Return a new string that contains the path part of the filename.
- `int fl_filename_relative` (char *to, int tolen, const char *from)
Makes a filename relative to the current working directory.
- `int fl_filename_relative` (char *to, int tolen, const char *from, const char *cwd)
Makes a filename relative to any other directory.
- `std::string fl_filename_relative_str` (const std::string &from)
Makes a filename relative to the current working directory.
- `std::string fl_filename_relative_str` (const std::string &from, const std::string &base)
Makes a filename relative to any directory.
- `char * fl_filename_setext` (char *to, int tolen, const char *ext)
Replaces the extension in `buf` of `max`.
- `std::string fl_filename_setext_str` (const std::string &filename, const std::string &new_extension)
Return a copy of the old filename with the new extension.
- `std::string fl_getcwd_str` ()
Cross-platform function to get the current working directory as a UTF-8 encoded value in an `std::string`.
- `int fl_open_uri` (const char *uri, char *msg, int msglen)
Opens the specified Uniform Resource Identifier (URI).

12.3.1 Detailed Description

File names and URI utility functions.

12.4 filename.H

[Go to the documentation of this file.](#)

```

00001 /*
00002  * Filename header file for the Fast Light Tool Kit (FLTK).
00003  *
00004  * Copyright 1998-2023 by Bill Spitzak and others.
00005  *
00006  * This library is free software. Distribution and use rights are outlined in
00007  * the file "COPYING" which should have been included with this file.  If this
00008  * file is missing or damaged, see the license at:
00009  *
00010  *     https://www.fltk.org/COPYING.php
00011  *
00012  * Please see the following page on how to report bugs and issues:
00013  *
00014  *     https://www.fltk.org/bugs.php
00015  */
00016
00017 /*
00018  * Note to devs:

```

```

00019  * Under Windows, we include filename.H from numericssort.c; this should probably change.
00020  * This implies that we need C-style comments and '#ifdef __cplusplus ... #endif'
00021  */
00022
00027 #ifndef FL_FILENAME_H
00028 #   define FL_FILENAME_H
00029
00030 #include "Fl_Export.H"
00031 #include <FL/platform_types.h>
00032
00033 #ifdef __cplusplus
00034
00035 // The following include is not (yet) used in FLTK 1.4
00036 // In FLTK 1.5 or 4.0 using std::string would be default.
00037 // #include <string>
00038
00039 #endif /* __cplusplus */
00040
00045 #   define FL_PATH_MAX 2048
00061 FL_EXPORT const char *fl_filename_name(const char * filename);
00062 FL_EXPORT const char *fl_filename_ext(const char *buf);
00063 FL_EXPORT char *fl_filename_settext(char *to, int tolen, const char *ext);
00064 FL_EXPORT int fl_filename_expand(char *to, int tolen, const char *from);
00065 FL_EXPORT int fl_filename_absolute(char *to, int tolen, const char *from);
00066 FL_EXPORT int fl_filename_relative(char *to, int tolen, const char *from);
00067 FL_EXPORT int fl_filename_match(const char *name, const char *pattern);
00068 FL_EXPORT int fl_filename_isdir(const char *name);
00069
00070 #   if defined(__cplusplus)
00071
00072 FL_EXPORT int fl_filename_absolute(char *to, int tolen, const char *from, const char *cwd);
00073 FL_EXPORT int fl_filename_relative(char *to, int tolen, const char *from, const char *cwd);
00074
00075 #include <string>
00076
00077 FL_EXPORT std::string fl_filename_name_str(const std::string &filename);
00078 FL_EXPORT std::string fl_filename_path_str(const std::string &filename);
00079 FL_EXPORT std::string fl_filename_ext_str(const std::string &filename);
00080 FL_EXPORT std::string fl_filename_settext_str(const std::string &filename, const std::string
&new_extension);
00081 FL_EXPORT std::string fl_filename_expand_str(const std::string &from);
00082 FL_EXPORT std::string fl_filename_absolute_str(const std::string &from);
00083 FL_EXPORT std::string fl_filename_absolute_str(const std::string &from, const std::string &base);
00084 FL_EXPORT std::string fl_filename_relative_str(const std::string &from);
00085 FL_EXPORT std::string fl_filename_relative_str(const std::string &from, const std::string &base);
00086 FL_EXPORT std::string fl_getcwd_str();
00087
00088 #   endif /* defined(__cplusplus) */
00089
00090 #   if defined(__cplusplus) && !defined(FL_DOXYGEN)
00091 /*
00092  * Under Windows, we include filename.H from numericssort.c; this should probably change...
00093  */
00094
00095 inline char *fl_filename_settext(char *to, const char *ext) { return fl_filename_settext(to,
FL_PATH_MAX, ext); }
00096 inline int fl_filename_expand(char *to, const char *from) { return fl_filename_expand(to, FL_PATH_MAX,
from); }
00097 inline int fl_filename_absolute(char *to, const char *from) { return fl_filename_absolute(to,
FL_PATH_MAX, from); }
00098 inline int fl_filename_relative(char *to, const char *from) { return fl_filename_relative(to,
FL_PATH_MAX, from); }
00099 #   endif /* __cplusplus */
00100
00101 #   if defined(__cplusplus)
00102 extern "C" {
00103 #   endif /* __cplusplus */
00104
00105 #   if !defined(FL_DOXYGEN)
00106 FL_EXPORT int fl_alphasort(struct dirent **, struct dirent **);
00107 FL_EXPORT int fl_casealphasort(struct dirent **, struct dirent **);
00108 FL_EXPORT int fl_casenumERICsort(struct dirent **, struct dirent **);
00109 FL_EXPORT int fl_numericssort(struct dirent **, struct dirent **);
00110 #   endif
00111
00112 typedef int (Fl_File_Sort_F)(struct dirent **, struct dirent **);
00114 #   if defined(__cplusplus)
00115 }
00116
00117 /*
00118  * Portable "scandir" function.  Ugly but necessary...
00119  */
00120
00121 FL_EXPORT int fl_filename_list(const char *d, struct dirent ***l,
Fl_File_Sort_F *s = fl_numericssort);
00123 FL_EXPORT void fl_filename_free_list(struct dirent ***l, int n);
00124

```

```

00125 /*
00126  * Generic function to open a Uniform Resource Identifier (URI) using a
00127  * system-defined program (added in FLTK 1.1.8)
00128  */
00129
00130 FL_EXPORT int    fl_open_uri(const char *uri, char *msg = (char *)0,
00131                             int msglen = 0);
00132
00133 FL_EXPORT void fl_decode_uri(char *uri);
00134
00135 #  endif /* __cplusplus */
00136
00137 /*
00138  * Note: FLTK 1.0.x compatibility definitions (FLTK_1_0_COMPAT) dropped in 1.4.0
00139  */
00140
00141 #endif /* FL_FILENAME_H */
00142

```

12.5 FL.H File Reference

Fl static class.

```

#include <FL/fl_config.h>
#include <FL/Fl_Export.H>
#include <FL/platform_types.h>
#include <FL/fl_casts.H>
#include <FL/Fl_Cairo.H>
#include "fl_utf8.h"
#include "Enumerations.H"
#include <string.h>

```

Classes

- class [Fl](#)
The [Fl](#) is the FLTK global (static) class containing state information and global methods for the current application.
- class [Fl_Widget_Tracker](#)
This class should be used to control safe widget deletion.

Macros

- #define **FL_Object** [Fl_Widget](#)
for back compatibility - use [Fl_Widget](#)!

Typedefs

- typedef void(* **FL_Abort_Handler**) (const char *format,...)
Signature of *set_abort* functions passed as parameters.
- typedef int(* **FL_Args_Handler**) (int argc, char **argv, int &i)
Signature of *args* functions passed as parameters.
- typedef void(* **FL_Atclose_Handler**) ([Fl_Window](#) *window, void *data)
Signature of *set_atclose* functions passed as parameters.
- typedef void(* **FL_Awake_Handler**) (void *data)
Signature of some wakeup callback functions passed as parameters.
- typedef void **FL_Box_Draw_F**(int x, int y, int w, int h, [Fl_Color](#) color)
Signature of some box drawing functions passed as parameters.
- typedef void **FL_Box_Draw_Focus_F**([Fl_Boxtype](#) bt, int x, int y, int w, int h, [Fl_Color](#) fg, [Fl_Color](#) bg)
Signature of box focus frame drawing functions.
- typedef void(* **FL_Clipboard_Notify_Handler**) (int source, void *data)
Signature of *add_clipboard_notify* functions passed as parameters.
- typedef int(* [Fl_Event_Dispatch](#)) (int event, [Fl_Window](#) *w)

- Signature of event_dispatch functions passed as parameters.*
- typedef int(* **Fl_Event_Handler**) (int event)
- Signature of add_handler functions passed as parameters.*
- typedef void(* **Fl_FD_Handler**) ([FL_SOCKET](#) fd, void *data)
- Signature of add_fd functions passed as parameters.*
- typedef void(* **Fl_Idle_Handler**) (void *data)
- Signature of add_idle callback functions passed as parameters.*
- typedef void **Fl_Label_Draw_F**(const [Fl_Label](#) *label, int x, int y, int w, int h, [Fl_Align](#) align)
- Signature of some label drawing functions passed as parameters.*
- typedef void **Fl_Label_Measure_F**(const [Fl_Label](#) *label, int &width, int &height)
- Signature of some label measurement functions passed as parameters.*
- typedef void(* **Fl_Old_Idle_Handler**) ()
- Signature of set_idle callback functions passed as parameters.*
- typedef int(* **Fl_System_Handler**) (void *event, void *data)
- Signature of add_system_handler functions passed as parameters.*
- typedef void(* [Fl_Timeout_Handler](#)) (void *data)
- Signature of timeout callback functions passed as parameters.*

Variables

- const char * **fl_local_alt**
string pointer used in shortcuts, you can change it to another language
- const char * **fl_local_ctrl**
string pointer used in shortcuts, you can change it to another language
- const char * **fl_local_meta**
string pointer used in shortcuts, you can change it to another language
- const char * **fl_local_shift**
string pointer used in shortcuts, you can change it to another language

12.5.1 Detailed Description

[Fl](#) static class.

12.6 Fl.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Main header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00021 #ifndef Fl_H
00022 #   define Fl_H
00023
00024 #include <FL/fl_config.h> // build configuration
00025 #include <FL/Fl_Export.H>
00026 #include <FL/platform_types.h> // for FL_SOCKET
00027 #include <FL/fl_casts.H>      // experimental
00028
00029 #ifdef FLTK_HAVE_CAIRO
00030 #   include <FL/Fl_Cairo.H>
00031 #endif

```

```

00032
00033 # include "fl_utf8.h"
00034 # include "Enumerations.H"
00035 # ifndef Fl_Object
00036 #     define Fl_Object Fl_Widget
00037 # endif
00038
00039 # ifdef check
00040 #     undef check
00041 # endif
00042
00043 # ifdef BSD
00044 #     undef BSD
00045 # endif
00046
00047 #include <string.h> // FIXME: Fl::is_scheme(): strcmp needs string.h
00048
00049 class Fl_Widget;
00050 class Fl_Window;
00051 class Fl_Image;
00052 struct Fl_Label;
00053 class Fl_Screen_Driver;
00054 class Fl_System_Driver;
00055
00056 // Pointers you can use to change FLTK to another language.
00057 // Note: Similar pointers are defined in FL/fl_ask.H and src/fl_ask.cxx
00058
00059 extern FL_EXPORT const char* fl_local_alt;
00060 extern FL_EXPORT const char* fl_local_ctrl;
00061 extern FL_EXPORT const char* fl_local_meta;
00062 extern FL_EXPORT const char* fl_local_shift;
00063
00063 typedef void (Fl_Label_Draw_F)(const Fl_Label *label, int x, int y, int w, int h, Fl_Align align);
00064
00066 typedef void (Fl_Label_Measure_F)(const Fl_Label *label, int &width, int &height);
00067
00068 typedef void (Fl_Box_Draw_F)(int x, int y, int w, int h, Fl_Color color);
00069
00092 typedef void (Fl_Box_Draw_Focus_F)(Fl_Boxtype bt, int x, int y, int w, int h, Fl_Color fg, Fl_Color
bg);
00093
00097 typedef void (*Fl_Timeout_Handler)(void *data);
00098
00100 typedef void (*Fl_Awake_Handler)(void *data);
00101
00103 typedef void (*Fl_Idle_Handler)(void *data);
00104
00106 typedef void (*Fl_Old_Idle_Handler)();
00107
00109 typedef void (*Fl_FD_Handler)(FL_SOCKET fd, void *data);
00110
00112 typedef int (*Fl_Event_Handler)(int event);
00113
00115 typedef int (*Fl_System_Handler)(void *event, void *data);
00116
00118 typedef void (*Fl_Abort_Handler)(const char *format,...);
00119
00121 typedef void (*Fl_Atclose_Handler)(Fl_Window *window, void *data);
00122
00124 typedef int (*Fl_Args_Handler)(int argc, char **argv, int &i);
00125
00128 typedef int (*Fl_Event_Dispatch)(int event, Fl_Window *w);
00129
00131 typedef void (*Fl_Clipboard_Notify_Handler)(int source, void *data);
00132
00133 /* group callback_functions */
00134
00135
00140 class FL_EXPORT Fl {
00141     friend class Fl_System_Driver;
00142     Fl() {} // no constructor!
00143
00144 private:
00145
00146     static int use_high_res_GL_;
00147     static int draw_GL_text_with_textures_;
00148     static int box_shadow_width_;
00149     static int box_border_radius_max_;
00150     static int selection_to_clipboard_;
00151
00152 public:
00153
00154     static Fl_Screen_Driver *screen_driver();
00155     static Fl_System_Driver *system_driver();
00156 #ifdef __APPLE__ // deprecated in 1.4 - only for compatibility with 1.3
00157     static void reset_marked_text();
00158     static void insertion_point_location(int x, int y, int height);

```



```

00159 #endif
00160
00161
00165 static int box_shadow_width() { return box_shadow_width_; }
00170 static void box_shadow_width(int W) { box_shadow_width_ = W < 1 ? 1 : W; }
00171
00175 static int box_border_radius_max() { return box_border_radius_max_; }
00188 static void box_border_radius_max(int R) { box_border_radius_max_ = R < 5 ? 5 : R; }
00189
00190 public: // should be private!
00191
00192 #ifndef FL_DOXYGEN
00193 static int e_number;
00194 static int e_x;
00195 static int e_y;
00196 static int e_x_root;
00197 static int e_y_root;
00198 static int e_dx;
00199 static int e_dy;
00200 static int e_state;
00201 static int e_clicks;
00202 static int e_is_click;
00203 static int e_keysym;
00204 static char* e_text;
00205 static int e_length;
00206 static void *e_clipboard_data;
00207 static const char *e_clipboard_type;
00208 static Fl_Event_Dispatch e_dispatch;
00209 static Fl_Callback_Reason callback_reason_;
00210 static Fl_Widget* belowmouse_;
00211 static Fl_Widget* pushed_;
00212 static Fl_Widget* focus_;
00213 static int damage_;
00214 static Fl_Widget* selection_owner_;
00215 static Fl_Window* modal_;
00216 static Fl_Window* grab_;
00217 static int compose_state; // used for dead keys (Windows) or marked text (MacOS)
00218 static void call_screen_init(); // recompute screen number and dimensions
00219 #endif // FL_DOXYGEN
00220
00221
00225 static void damage(int d) {damage_ = d;}
00226
00227 public:
00234 typedef enum {
00244     OPTION_ARROW_FOCUS = 0,
00245     // When switched on, FLTK will use the file chooser dialog that comes
00246     // with your operating system whenever possible. When switched off, FLTK
00247     // will present its own file chooser.
00248     // \todo implement me
00249     // OPTION_NATIVE_FILECHOOSER,
00250     // When Filechooser Preview is enabled, the FLTK or native file chooser
00251     // will show a preview of a selected file (if possible) before the user
00252     // decides to choose the file.
00253     // \todo implement me
00254     //OPTION_FILECHOOSER_PREVIEW,
00259     OPTION_VISIBLE_FOCUS,
00263     OPTION_DND_TEXT,
00267     OPTION_SHOW_TOOLTIPS,
00271     OPTION_FNFC_USES_GTK,
00274     OPTION_FNFC_USES_ZENITY,
00278     OPTION_FNFC_USES_KDIALOG,
00282     OPTION_PRINTER_USES_GTK,
00286     OPTION_SHOW_SCALING,
00292     OPTION_SIMPLE_ZOOM_SHORTCUT,
00293     // don't change this, leave it always as the last element
00295     OPTION_LAST
00296 } Fl_Option;
00297
00298 private:
00299 static unsigned char options_[OPTION_LAST];
00300 static unsigned char options_read_;
00301 static int program_should_quit_; // non-zero means the program was asked to cleanly terminate
00302
00303 public:
00304 /*
00305  * Return a global setting for all FLTK applications, possibly overridden
00306  * by a setting specifically for this application.
00307  */
00308 static bool option(Fl_Option opt);
00309
00310 /*
00311  * Override an option while the application is running.
00312  */
00313 static void option(Fl_Option opt, bool val);
00314
00322 static void (*idle)();

```

```

00323
00324 #ifndef FL_DOXYGEN
00325 private:
00326     static Fl_Awake_Handler *awake_ring_;
00327     static void **awake_data_;
00328     static int awake_ring_size_;
00329     static int awake_ring_head_;
00330     static int awake_ring_tail_;
00331 public:
00332     static const char* scheme_;
00333     static Fl_Image* scheme_bg_;
00334
00335     static int e_original_keysym; // late addition
00336     static int scrollbar_size_;
00337     static int menu_linespacing_; // STR #2927
00338 #endif
00339
00340
00341     static int add_aware_handler(Fl_Awake_Handler, void*);
00342     static int get_aware_handler(Fl_Awake_Handler&, void*&);
00343
00344 public:
00345
00346     // API version number
00347     static double version();
00348     static int api_version();
00349
00350     // ABI version number
00351     static int abi_version();
00352
00375     static inline int abi_check(const int val = FL_ABI_VERSION) {
00376         return val == abi_version();
00377     }
00378
00379     // argument parsers:
00380     static int arg(int argc, char **argv, int& i);
00381     static int args(int argc, char **argv, int& i, Fl_Args_Handler cb = 0);
00382     static void args(int argc, char **argv);
00387     static const char* const help;
00388
00389     // things called by initialization:
00390     static void display(const char*);
00391     static int visual(int);
00401     static int gl_visual(int, int *alist=0); // platform dependent
00402     static void own_colormap();
00403     static void get_system_colors();
00404     static void foreground(uchar, uchar, uchar);
00405     static void background(uchar, uchar, uchar);
00406     static void background2(uchar, uchar, uchar);
00407
00408     // schemes:
00409     static int scheme(const char *name);
00411     static const char* scheme() {return scheme_;}
00412
00440     static int is_scheme(const char *name) {
00441         return (scheme_ && name && !strcmp(name,scheme_));
00442     }
00443
00444     static int reload_scheme(); // defined in 'src/Fl_get_system_colors.cxx'
00445
00446     static int scrollbar_size();
00447     static void scrollbar_size(int W);
00448     static int menu_linespacing();
00449     static void menu_linespacing(int H);
00450
00451     // execution:
00452     static int wait();
00453     static double wait(double time);
00454     static int check();
00455     static int ready();
00456     static int run();
00463     static int program_should_quit() {return program_should_quit_;}
00469     static void program_should_quit(int should_i) { program_should_quit_ = should_i; }
00470     static void hide_all_windows();
00471
00472     static Fl_Widget* readqueue();
00473
00474     //
00475     // cross-platform timer support
00476     //
00477
00478     static void add_timeout(double t, Fl_Timeout_Handler cb, void *data = 0);
00479     static void repeat_timeout(double t, Fl_Timeout_Handler cb, void *data = 0);
00480     static int has_timeout(Fl_Timeout_Handler cb, void *data = 0);
00481     static void remove_timeout(Fl_Timeout_Handler cb, void *data = 0);
00482     static int remove_next_timeout(Fl_Timeout_Handler cb, void *data = 0, void **data_return = 0);
00483

```

```

00484 static void add_check(Fl_Timeout_Handler, void* = 0);
00485 static int has_check(Fl_Timeout_Handler, void* = 0);
00486 static void remove_check(Fl_Timeout_Handler, void* = 0);
00487
00488 static Fl_Timestamp now(double offset = 0);
00489 static double seconds_since(Fl_Timestamp& then);
00490 static double seconds_between(Fl_Timestamp& back, Fl_Timestamp& further_back);
00491 static long ticks_since(Fl_Timestamp& then);
00492 static long ticks_between(Fl_Timestamp& back, Fl_Timestamp& further_back);
00493
00494 // private
00495 static void run_idle();
00496 static void run_checks();
00497 static void add_fd(int fd, int when, Fl_FD_Handler cb, void* = 0); // platform dependent
00498 static void add_fd(int fd, Fl_FD_Handler cb, void* = 0); // platform dependent
00500 static void remove_fd(int, int when); // platform dependent
00502 static void remove_fd(int); // platform dependent
00503
00504 static void add_idle(Fl_Idle_Handler cb, void* data = 0);
00505 static int has_idle(Fl_Idle_Handler cb, void* data = 0);
00506 static void remove_idle(Fl_Idle_Handler cb, void* data = 0);
00508 static int damage() {return damage_;}
00509 static void redraw();
00510 static void flush();
00531 static void (*warning)(const char*, ...);
00546 static void (*error)(const char*, ...);
00563 static void (*fatal)(const char*, ...);
00569 static Fl_Window* first_window();
00570 static void first_window(Fl_Window*);
00571 static Fl_Window* next_window(const Fl_Window*);
00572
00582 static Fl_Window* modal() {return modal_;}
00588 static Fl_Window* grab() {return grab_;}
00613 static void grab(Fl_Window*); // platform dependent
00620 // event information:
00626 static int event() {return e_number;}
00631 static int event_x() {return e_x;}
00636 static int event_y() {return e_y;}
00643 static int event_x_root() {return e_x_root;}
00650 static int event_y_root() {return e_y_root;}
00655 static int event_dx() {return e_dx;}
00660 static int event_dy() {return e_dy;}
00669 static void get_mouse(int &,int &);
00678 static int event_clicks() {return e_clicks;}
00686 static void event_clicks(int i) {e_clicks = i;}
00694 static int event_is_click() {return e_is_click;}
00701 static void event_is_click(int i) {e_is_click = i;}
00713 static int event_button() {return e_keysym - FL_Button;}
00754 static int event_state() {return e_state;}
00755
00761 static int event_state(int mask) {return e_state&mask;}
00773 static int event_key() {return e_keysym;}
00782 static int event_original_key() {return e_original_keysym;}
00822 static int event_key(int key);
00828 static int get_key(int key); // platform dependent
00843 static const char* event_text() {return e_text;}
00850 static int event_length() {return e_length;}
00851
00855 static void *event_clipboard() {return e_clipboard_data;}
00859 static const char *event_clipboard_type() {return e_clipboard_type;}
00860
00861
00862 static int compose(int &del);
00863 static void compose_reset();
00864 static int event_inside(int,int,int,int);
00865 static int event_inside(const Fl_Widget*);
00866 static int test_shortcut(Fl_Shortcut);
00867
00868 static void enable_im();
00869 static void disable_im();
00870
00871 // event destinations:
00872 static int handle(int, Fl_Window*);
00873 static int handle_(int, Fl_Window*);
00876 static Fl_Widget* belowmouse() {return belowmouse_;}
00877 static void belowmouse(Fl_Widget*);
00880 static Fl_Widget* pushed() {return pushed_;}
00881 static void pushed(Fl_Widget*);
00883 static Fl_Widget* focus() {return focus_;}
00884 static void focus(Fl_Widget*);
00885 static void add_handler(Fl_Event_Handler ha);
00886 static void add_handler(Fl_Event_Handler ha, Fl_Event_Handler before);
00887 static Fl_Event_Handler last_handler();
00888 static void remove_handler(Fl_Event_Handler h);
00889 static void add_system_handler(Fl_System_Handler h, void *data);
00890 static void remove_system_handler(Fl_System_Handler h);
00891 static void event_dispatch(Fl_Event_Dispatch d);

```

```

00892 static Fl_Event_Dispatch event_dispatch();
00893 static Fl_Callback_Reason callback_reason();
00939 static void copy(const char *stuff, int len, int destination = 0,
00940                const char *type = Fl::clipboard_plain_text);
00941
00969 static void selection_to_clipboard(int mode) {
00970     selection_to_clipboard_ = mode ? 1 : 0;
00971 }
00972
00978 static int selection_to_clipboard() { return selection_to_clipboard_; }
00979
01022 static void paste(Fl_Widget &receiver, int source, const char *type = Fl::clipboard_plain_text);
01023
01045 static void add_clipboard_notify(Fl_Clipboard_Notify_Handler h, void *data = 0);
01050 static void remove_clipboard_notify(Fl_Clipboard_Notify_Handler h);
01059 static int clipboard_contains(const char *type);
01062 static char const * const clipboard_plain_text;
01065 static char const * const clipboard_image;
01066
01076 static int dnd(); // platform dependent
01077
01078 // These are for back-compatibility only:
01081 static Fl_Widget* selection_owner() {return selection_owner_;}
01082 static void selection_owner(Fl_Widget*);
01083 static void selection(Fl_Widget &owner, const char*, int len);
01084 static void paste(Fl_Widget &receiver);
01101 static int x(); // via screen driver
01102 static int y(); // via screen driver
01103 static int w(); // via screen driver
01104 static int h(); // via screen driver
01105
01106 // multi-head support:
01107 static int screen_count(); // via screen driver
01108 static void screen_xywh(int &X, int &Y, int &W, int &H); // via screen driver
01109 static void screen_xywh(int &X, int &Y, int &W, int &H, int mx, int my); // via screen driver
01110 static void screen_xywh(int &X, int &Y, int &W, int &H, int n); // via screen driver
01111 static void screen_xywh(int &X, int &Y, int &W, int &H, int mx, int my, int mw, int mh); // via
screen driver
01112 static int screen_num(int x, int y); // via screen driver
01113 static int screen_num(int x, int y, int w, int h); // via screen driver
01114 static void screen_dpi(float &h, float &v, int n=0); // via screen driver
01115 static void screen_work_area(int &X, int &Y, int &W, int &H, int mx, int my); // via screen driver
01116 static void screen_work_area(int &X, int &Y, int &W, int &H, int n); // via screen driver
01117 static void screen_work_area(int &X, int &Y, int &W, int &H); // via screen driver
01118 static float screen_scale(int n); // via screen driver
01119 static void screen_scale(int n, float factor); // via screen driver
01120 static int screen_scaling_supported();
01121 static void keyboard_screen_scaling(int value);
01122
01131 // color map:
01132 static void set_color(Fl_Color, uchar, uchar, uchar);
01133 static void set_color(Fl_Color, uchar, uchar, uchar, uchar);
01138 static void set_color(Fl_Color i, unsigned c); // platform dependent
01139 static unsigned get_color(Fl_Color i);
01140 static void get_color(Fl_Color i, uchar &red, uchar &green, uchar &blue);
01141 static void get_color(Fl_Color i, uchar &red, uchar &green, uchar &blue, uchar &alpha);
01147 static void free_color(Fl_Color i, int overlay = 0); // platform dependent
01148
01149 // fonts:
01150 static const char* get_font(Fl_Font);
01163 static const char* get_font_name(Fl_Font, int* attributes = 0);
01175 static int get_font_sizes(Fl_Font, int*& sizes);
01176 static void set_font(Fl_Font, const char*);
01177 static void set_font(Fl_Font, Fl_Font);
01205 static Fl_Font set_fonts(const char* = 0); // platform dependent
01206
01213 // <Hack to re-order the 'Drawing functions' group>
01216 // labeltypes:
01217 static void set_labeltype(Fl_Labeltype, Fl_Label_Draw_F*, Fl_Label_Measure_F*);
01219 static void set_labeltype(Fl_Labeltype, Fl_Labeltype from); // is it defined ?
01220
01221 // boxtypes:
01222 static Fl_Box_Draw_F *get_boxtype(Fl_Boxtype);
01223 static void set_boxtype(Fl_Boxtype, Fl_Box_Draw_F*,
01224                        uchar, uchar, uchar, uchar,
01225                        Fl_Box_Draw_Focus_F* =NULL);
01226 static void set_boxtype(Fl_Boxtype, Fl_Boxtype from);
01227 static int box_dx(Fl_Boxtype);
01228 static int box_dy(Fl_Boxtype);
01229 static int box_dw(Fl_Boxtype);
01230 static int box_dh(Fl_Boxtype);
01231
01232 static int draw_box_active();
01233 static Fl_Color box_color(Fl_Color);
01234 static void set_box_color(Fl_Color);
01235
01236 // back compatibility:

```

```

01240 static void set_abort(Fl_Abort_Handler f) {fatal = f;}
01241 static void (*atclose)(Fl_Window*,void*);
01242 static void default_atclose(Fl_Window*,void*);
01246 static void set_atclose(Fl_Atclose_Handler f) {atclose = f;}
01252 static int event_shift() {return e_state&FL_SHIFT;}
01254 static int event_ctrl() {return e_state&FL_CTRL;}
01256 static int event_command() {return e_state&FL_COMMAND;}
01258 static int event_alt() {return e_state&FL_ALT;}
01267 static int event_buttons() {return e_state & FL_BUTTONS;}
01272 static int event_button1() {return e_state & FL_BUTTON1;}
01277 static int event_button2() {return e_state & FL_BUTTON2;}
01282 static int event_button3() {return e_state & FL_BUTTON3;}
01287 static int event_button4() {return e_state & FL_BUTTON4;}
01292 static int event_button5() {return e_state & FL_BUTTON5;}
01300 static void set_idle(Fl_Old_Idle_Handler cb) {idle = cb;}
01302 static void grab(Fl_Window& win) {grab(&win);}
01306 static void release() {grab(0);}
01307
01308 // Visible focus methods...
01314 static void visible_focus(int v) { option(OPTION_VISIBLE_FOCUS, (v!=0)); }
01320 static int visible_focus() { return option(OPTION_VISIBLE_FOCUS); }
01321
01322 // Drag-n-drop text operation methods...
01329 static void dnd_text_ops(int v) { option(OPTION_DND_TEXT, (v!=0)); }
01336 static int dnd_text_ops() { return option(OPTION_DND_TEXT); }
01341 // Multithreading support:
01342 static int lock();
01343 static void unlock();
01344 static void awake(void* message = 0);
01346 static int awake(Fl_Awake_Handler cb, void* message = 0);
01353 static void* thread_message(); // platform dependent
01385 // Widget deletion:
01386 static void delete_widget(Fl_Widget *w);
01387 static void do_widget_deletion();
01388 static void watch_widget_pointer(Fl_Widget *w);
01389 static void release_widget_pointer(Fl_Widget *w);
01390 static void clear_widget_pointer(Fl_Widget const *w);
01397 static void use_high_res_GL(int val) { use_high_res_GL_ = val; }
01403 static int use_high_res_GL() { return use_high_res_GL_; }
01404
01416 static void draw_GL_text_with_textures(int val) { draw_GL_text_with_textures_ = val; }
01417
01423 static int draw_GL_text_with_textures() { return draw_GL_text_with_textures_; }
01424
01425 static int system(const char *command);
01426
01427 // Convert Windows commandline arguments to UTF-8 (documented in src/Fl.cxx)
01428 static int args_to_utf8(int argc, char ** &argv);
01429
01430 #ifndef FLTK_HAVE_CAIRO
01434 public:
01435 // Cairo support API
01436
01437 static cairo_t *cairo_make_current(Fl_Window *w);
01438
01454 static void cairo_autolink_context(bool alink) {
01455     cairo_state_.autolink(alink);
01456 }
01457
01465 static bool cairo_autolink_context() {
01466     return cairo_state_.autolink();
01467 }
01468
01470 static cairo_t *cairo_cc() {
01471     return cairo_state_.cc();
01472 }
01473
01478 static void cairo_cc(cairo_t *c, bool own=false) {
01479     cairo_state_.cc(c, own);
01480 }
01481
01515 static void cairo_flush(cairo_t *c) {
01516     // flush Cairo drawings: necessary at least for Windows
01517     cairo_surface_t *s = cairo_get_target(c);
01518     cairo_surface_flush(s);
01519 }
01520
01521 private:
01522 static cairo_t *cairo_make_current(void *gc);
01523 static cairo_t *cairo_make_current(void *gc, int W, int H);
01524 static Fl_Cairo_State cairo_state_;
01525
01526 public:
01529 #endif // FLTK_HAVE_CAIRO
01530
01531 };
01532

```

```

01575 class FL_EXPORT Fl_Widget_Tracker {
01576
01577     Fl_Widget* wp_;
01578
01579 public:
01580
01581     Fl_Widget_Tracker(Fl_Widget *wi);
01582     ~Fl_Widget_Tracker();
01583
01584     Fl_Widget *widget() {return wp_;}
01585
01586     int deleted() {return wp_ == 0;}
01587
01588     int exists() {return wp_ != 0;}
01589 };
01590
01591 #endif // !Fl_H

```

12.7 Fl_Adjuster.H

```

00001 //
00002 // Adjuster widget header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Adjuster widget . */
00019
00020 // 3-button "slider", made for Nuke
00021
00022 #ifndef Fl_Adjuster_H
00023 #define Fl_Adjuster_H
00024
00025 #ifndef Fl_Valuator_H
00026 #include "Fl_Valuator.H"
00027 #endif
00028
00029 class FL_EXPORT Fl_Adjuster : public Fl_Valuator {
00030
00031     int drag;
00032     int ix;
00033     int soft_;
00034
00035 protected:
00036     void draw() FL_OVERRIDE;
00037     int handle(int) FL_OVERRIDE;
00038     void value_damage() FL_OVERRIDE;
00039
00040 public:
00041     Fl_Adjuster(int X,int Y,int W,int H,const char *l=0);
00042     void soft(int s) {soft_ = s;}
00043     int soft() const {return soft_;}
00044 };
00045
00046 #endif

```

12.8 Fl_Anim_GIF_Image.H

```

00001 //
00002 // Fl_Anim_GIF_Image class header for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2016-2023 by Christian Grabner <wcout@gmx.net>.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //

```

```

00016
00017 #ifndef Fl_Anim_Gif_Image_H
00018 #define Fl_Anim_Gif_Image_H
00019
00020 // forward declarations
00021 class Fl_Image;
00022 class Fl_Widget;
00023
00024 #include <FL/Fl_GIF_Image.H>
00025
00026 // Load and display animated GIF images
00027 class FL_EXPORT Fl_Anim_GIF_Image : public Fl_GIF_Image {
00028
00029     class FrameInfo; // internal helper class
00030
00031 public:
00032
00033     enum Flags {
00034         DONT_START = 1,
00035         DONT_RESIZE_CANVAS = 2,
00036         DONT_SET_AS_IMAGE = 4,
00037         OPTIMIZE_MEMORY = 8,
00038         LOG_FLAG = 64,
00039         DEBUG_FLAG = 128
00040     };
00041
00042     // -- constructors and destructor
00043     Fl_Anim_GIF_Image(const char *filename, Fl_Widget *canvas = 0, unsigned short flags = 0);
00044     Fl_Anim_GIF_Image(const char *imagename, const unsigned char *data,
00045                     const size_t length, Fl_Widget *canvas = 0,
00046                     unsigned short flags = 0);
00047     Fl_Anim_GIF_Image();
00048     ~Fl_Anim_GIF_Image() FL_OVERRIDE;
00049
00050     // -- file handling
00051     bool load(const char *name, const unsigned char *imgdata=NULL, size_t imglength=0);
00052     bool valid() const;
00053
00054     // -- getters and setters
00055     void frame_uncache(bool uncache);
00056     bool frame_uncache() const;
00057     double delay(int frame_) const;
00058     void delay(int frame, double delay);
00059     void canvas(Fl_Widget *canvas, unsigned short flags = 0);
00060     Fl_Widget *canvas() const;
00061     int canvas_w() const;
00062     int canvas_h() const;
00063     bool is_animated() const;
00064     const char *name() const;
00065     void speed(double speed);
00066     double speed() const;
00067
00068     // -- animation
00069     int frames() const;
00070     void frame(int frame);
00071     int frame() const;
00072     Fl_Image *image() const;
00073     Fl_Image *image(int frame) const;
00074     bool start();
00075     bool stop();
00076     bool next();
00077
00078     bool playing() const { return valid() && Fl::has_timeout(cb_animate, (void *)this); }
00079
00080     // -- image data
00081     Fl_Anim_GIF_Image& resize(int w, int h);
00082     Fl_Anim_GIF_Image& resize(double scale);
00083     int frame_x(int frame) const;
00084     int frame_y(int frame) const;
00085     int frame_w(int frame) const;
00086     int frame_h(int frame) const;
00087
00088     // -- overridden methods
00089     void color_average(Fl_Color c, float i) FL_OVERRIDE;
00090     Fl_Image *copy(int W, int H) const FL_OVERRIDE;
00091     Fl_Image *copy() const { return Fl_Pixmap::copy(); }
00092     void desaturate() FL_OVERRIDE;
00093     void draw(int x, int y, int w, int h, int cx = 0, int cy = 0) FL_OVERRIDE;
00094     void uncache() FL_OVERRIDE;
00095
00096     // -- debugging and logging
00097     int debug() const;
00098
00099     // -- static methods
00100     static int frame_count(const char *name, const unsigned char *imgdata = NULL, size_t imglength = 0);
00101     static bool loop;

```

```

00152
00159     static double min_delay;
00160
00161 protected:
00162
00163     bool next_frame();
00164     void clear_frames();
00165     void set_frame(int frame);
00166
00167     static void cb_animate(void *d);
00168     void scale_frame();
00169     void set_frame();
00170     void on_frame_data(Fl_GIF_Image::GIF_FRAME &f) FL_OVERRIDE;
00171     void on_extension_data(Fl_GIF_Image::GIF_FRAME &f) FL_OVERRIDE;
00172
00173 private:
00174
00175     char *name_;
00176     unsigned short flags_;
00177     Fl_Widget *canvas_;
00178     bool uncache_;
00179     bool valid_;
00180     int frame_; // current frame
00181     double speed_;
00182     FrameInfo *fi_;
00183 };
00184
00185 #endif // Fl_Anim_Gif_Image_H

```

12.9 fl_ask.H File Reference

API for common dialogs.

```
#include <FL/Enumerations.H>
```

```
#include <FL/fl_attr.h>
```

Enumerations

- enum [Fl_Beep](#) {
[FL_BEEP_DEFAULT](#) = 0 , [FL_BEEP_MESSAGE](#) , [FL_BEEP_ERROR](#) , [FL_BEEP_QUESTION](#) ,
[FL_BEEP_PASSWORD](#) , [FL_BEEP_NOTIFICATION](#) }

Defines the different system beeps available.

Functions

- void [fl_alert](#) (const char *,...) [__fl_attr\(\(__format__\(__printf__](#)
- void [fl_ask](#) (const char *,...) [__fl_attr\(\(__format__\(__printf__](#)
- void [fl_beep](#) (int type=[FL_BEEP_DEFAULT](#))
Emits a system beep.
- int [fl_choice](#) (const char *q, const char *b0, const char *b1, const char *b2,...) [__fl_attr\(\(__format__\(__↵
printf__](#)
- int const char const char int [fl_choice_n](#) (const char *q, const char *b0, const char *b1, const char *b2,...)
[__fl_attr\(\(__format__\(__printf__](#)
- int const char * [fl_input](#) (const char *label, const char *deflt=0,...) [__fl_attr\(\(__format__\(__printf__](#)
- int const char const char int const char * [fl_input](#) (int maxchar, const char *label, const char *deflt=0,...)
[__fl_attr\(\(__format__\(__printf__](#)
- void [fl_message](#) (const char *,...) [__fl_attr\(\(__format__\(__printf__](#)
- void [fl_message_font](#) ([Fl_Font](#) f, [Fl_Fontsize](#) s)
- void [fl_message_hotspot](#) (int enable)
*Sets whether or not to move the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#),
[fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#) to follow the mouse pointer.*
- int [fl_message_hotspot](#) (void)
*Gets whether or not to move the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#),
[fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#) to follow the mouse pointer.*
- int const char const char int const char const char [Fl_Widget](#) * [fl_message_icon](#) ()

- Gets the [Fl_Box](#) icon container of the current default dialog used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).*
- void [fl_message_icon_label](#) (const char *str)
Sets the icon label of the dialog window used in many common dialogs.
 - void [fl_message_position](#) (const int x, const int y, const int center=0)
Sets the preferred position for the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).
 - void [fl_message_position](#) (Fl_Widget &widget)
 - void [fl_message_position](#) (Fl_Widget *widget)
Sets the preferred position for the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).
 - int [fl_message_position](#) (int *x=0, int *y=0)
Gets the preferred position for the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).
 - void [fl_message_title](#) (const char *title)
Sets the title of the dialog window used in many common dialogs.
 - void [fl_message_title_default](#) (const char *title)
Sets the default title of the dialog window used in many common dialogs.
 - int const char const char * [fl_password](#) (const char *label, const char *deflt=0,...) [__fl_attr](#)(([__format__](#) ([__printf__](#)
 - int const char const char int const char const char * [fl_password](#) (int maxchar, const char *label, const char *deflt=0,...) [__fl_attr](#)(([__format__](#) ([__printf__](#)

Variables

- void void int [__deprecated__](#)
- const char * [fl_cancel](#)
string pointer used in common dialogs, you can change it to another language
- const char * [fl_close](#)
string pointer used in common dialogs, you can change it to another language
- [Fl_Font](#) [fl_message_font__](#)
- [Fl_Fonsize](#) [fl_message_size__](#)
- const char * [fl_no](#)
string pointer used in common dialogs, you can change it to another language
- const char * [fl_ok](#)
string pointer used in common dialogs, you can change it to another language
- const char * [fl_yes](#)
string pointer used in common dialogs, you can change it to another language

12.9.1 Detailed Description

API for common dialogs.

12.9.2 Enumeration Type Documentation

12.9.2.1 Fl_Beep

enum [Fl_Beep](#)

Defines the different system beeps available.

Some systems may play different sounds or use different sound volume depending on the [Fl_Beep](#) value. The implementation is platform dependent.

See also

[fl_beep\(int\)](#)

Enumerator

FL_BEEP_DEFAULT	Default beep.
FL_BEEP_MESSAGE	Message beep.
FL_BEEP_ERROR	Error beep.
FL_BEEP_QUESTION	Question beep.
FL_BEEP_PASSWORD	Password beep.
FL_BEEP_NOTIFICATION	Notification beep.

12.9.3 Function Documentation

12.9.3.1 fl_message_position()

```
void fl_message_position (
    Fl_Widget & widget) [inline]
```

See also

[fl_message_position\(Fl_Widget *widget\).](#)

12.10 fl_ask.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Standard dialog header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00021 #ifndef _FL_fl_ask_H_
00022 #define _FL_fl_ask_H_
00023
00024 #include <FL/Enumerations.H>
00025 #include <FL/fl_attr.h>
00026
00027 #if (FLTK_USE_STD)
00028 #include <string>
00029 #endif
00030
00031 class Fl_Widget;
00032
00040 enum Fl_Beep {
00041     FL_BEEP_DEFAULT = 0,
00042     FL_BEEP_MESSAGE,
00043     FL_BEEP_ERROR,
00044     FL_BEEP_QUESTION,
00045     FL_BEEP_PASSWORD,
00046     FL_BEEP_NOTIFICATION
00047 };
00048
00049 FL_EXPORT void fl_beep(int type = FL_BEEP_DEFAULT);
00050 FL_EXPORT void fl_message(const char *, ...) __fl_attr((__format__(__printf__, 1, 2)));
00051 FL_EXPORT void fl_alert(const char *, ...) __fl_attr((__format__(__printf__, 1, 2)));
00052
00053 // fl_ask() is deprecated since it uses "Yes" and "No" for the buttons,
00054 // which does not conform to the current FLTK Human Interface Guidelines.
00055 // Use fl_choice() or fl_choice_n() with the appropriate verbs instead.
00056
00057 FL_EXPORT int fl_ask(const char *, ...) __fl_attr((__format__(__printf__, 1, 2), __deprecated__));
00058
00059 FL_EXPORT int fl_choice(const char *q, const char *b0, const char *b1, const char *b2, ...)
00060     __fl_attr((__format__(__printf__, 1, 5)));
00061 FL_EXPORT const char *fl_input(const char *label, const char *deflt = 0, ...)
00062     __fl_attr((__format__(__printf__, 1, 3)));
```

```

00062 FL_EXPORT const char *fl_password(const char *label, const char *deflt = 0, ...)
00063     __fl_attr((__format__ (__printf__, 1, 3)));
00064
00065 // since FLTK 1.3.8:
00066 // - fl_choice_n()      with extended return value (-2, -1, 0, 1, 2)
00067
00068 FL_EXPORT int fl_choice_n(const char *q, const char *b0, const char *b1, const char *b2, ...)
00069     __fl_attr((__format__ (__printf__, 1, 5)));
00070
00071 // since FLTK 1.4.0: with 'maxchar' to limit input size
00072
00073 FL_EXPORT const char *fl_input(int maxchar, const char *label, const char *deflt = 0, ...)
00074     __fl_attr((__format__ (__printf__, 2, 4)));
00075
00076 FL_EXPORT const char *fl_password(int maxchar, const char *label, const char *deflt = 0, ...)
00077     __fl_attr((__format__ (__printf__, 2, 4)));
00078
00079 // since FLTK 1.4.0 -- only with option FLTK_USE_STD
00080
00081 // - fl_input_str()      with limited input size, returns std::string
00082 // - fl_password_str() with limited input size, returns std::string
00083
00084 #if (FLTK_USE_STD)
00085
00086 FL_EXPORT std::string fl_input_str(int maxchar, const char *label, const char *deflt = 0, ...)
00087     __fl_attr((__format__ (__printf__, 2, 4)));
00088
00089 FL_EXPORT std::string fl_input_str(int &ret, int maxchar, const char *label, const char *deflt = 0,
00090     ...)
00091     __fl_attr((__format__ (__printf__, 3, 5)));
00092
00093 FL_EXPORT std::string fl_password_str(int maxchar, const char *label, const char *deflt = 0, ...)
00094     __fl_attr((__format__ (__printf__, 2, 4)));
00095
00096 FL_EXPORT std::string fl_password_str(int &ret, int maxchar, const char *label, const char *deflt = 0,
00097     ...)
00098     __fl_attr((__format__ (__printf__, 3, 5)));
00099
00100 #endif
00101
00102 FL_EXPORT Fl_Widget *fl_message_icon();
00103 extern FL_EXPORT Fl_Font fl_message_font_;
00104 extern FL_EXPORT Fl_Fontsize fl_message_size_;
00105 inline void fl_message_font(Fl_Font f, Fl_Fontsize s) {
00106     fl_message_font_ = f;
00107     fl_message_size_ = s;
00108 }
00109
00110 FL_EXPORT void fl_message_hotspot(int enable);
00111 FL_EXPORT int fl_message_hotspot(void);
00112
00113 // since FLTK 1.4.0: fl_message_position()
00114
00115 FL_EXPORT void fl_message_position(const int x, const int y, const int center = 0);
00116 FL_EXPORT void fl_message_position(Fl_Widget *widget);
00117 FL_EXPORT int fl_message_position(int *x = 0, int *y = 0);
00118
00119 inline void fl_message_position(Fl_Widget &widget) {
00120     fl_message_position(&widget);
00121 }
00122
00123 FL_EXPORT void fl_message_title(const char *title);
00124 FL_EXPORT void fl_message_title_default(const char *title);
00125
00126 FL_EXPORT void fl_message_icon_label(const char *str);
00127
00128 // pointers you can use to change FLTK to another language:
00129 extern FL_EXPORT const char *fl_no;
00130 extern FL_EXPORT const char *fl_yes;
00131 extern FL_EXPORT const char *fl_ok;
00132 extern FL_EXPORT const char *fl_cancel;
00133 extern FL_EXPORT const char *fl_close;
00134
00135 #endif // !_FL_fl_ask_H_

```

12.11 fl_attr.h File Reference

This file defines compiler-specific macros.

Macros

- #define `__fl_attr(x)`

This section lists macros for Doxygen documentation only.

- `#define FL_DEPRECATED(msg, func)`

Enclosing a function or method in FL_DEPRECATED marks it as no longer recommended.

- `#define FL_OVERRIDE override`

This macro makes it safe to use the C++11 keyword `override` with older compilers.

12.11.1 Detailed Description

This file defines compiler-specific macros.

12.11.2 Macro Definition Documentation

12.11.2.1 `__fl_attr`

```
#define __fl_attr(
    x)
```

This section lists macros for Doxygen documentation only.

The next section will define the actual macros based on the compiler used and based on the capabilities of the version of that compiler. To be used in prototypes with a variable list of arguments. This macro helps detection of mismatches between format string and argument list at compilation time.

Usage example: [FL/fl_ask.H](#)

12.11.2.2 FL_DEPRECATED

```
#define FL_DEPRECATED(
    msg,
    func)
```

Value:

```
/*##### \deprecated msg */ \
func
```

Enclosing a function or method in FL_DEPRECATED marks it as no longer recommended.

This macro syntax can not be used if the return type contains a comma, which is not the case in FLTK.

```
FL_DEPRECATED("Outdated, don't use", int position()) { return position; }
```

12.11.2.3 FL_OVERRIDE

```
#define FL_OVERRIDE override
```

This macro makes it safe to use the C++11 keyword `override` with older compilers.

Since FLTK 1.5.0 a C++11 capable compiler is required, hence we can safely define FL_OVERRIDE unconditionally as 'override'.

This macro is kept for backwards compatibility with old (1.4.x) user code that uses the FL_OVERRIDE macro.

We recommend to use 'override' explicitly if C++11 or higher is available.

12.12 fl_attr.h

[Go to the documentation of this file.](#)

```
00001 /*
00002  * Function attribute declarations for the Fast Light Tool Kit (FLTK).
00003  *
00004  * Copyright 1998-2025 by Bill Spitzak and others.
00005  *
00006  * This library is free software. Distribution and use rights are outlined in
00007  * the file "COPYING" which should have been included with this file. If this
00008  * file is missing or damaged, see the license at:
00009  *
00010  *     https://www.fltk.org/COPYING.php
00011  *
00012  * Please see the following page on how to report bugs and issues:
00013  *
00014  *     https://www.fltk.org/bugs.php
00015  */
00016
00022 #ifndef _FL_fl_attr_h_
00023 #define _FL_fl_attr_h_
00024
```

```

00025
00038 #define FL_OVERRIDE override
00039
00040
00046 #ifndef FL_DOXYGEN
00047
00048
00055 #define __fl_attr(x)
00056
00057
00067 #define FL_DEPRECATED(msg, func) \
00068     /##### \deprecated msg #### \
00069     func
00070
00071
00072 #else /* FL_DOXYGEN */
00073
00074 // If FL_NO_DEPRECATED is defined FLTK 1.4 can compile 1.3.x code without
00075 // issuing several "deprecated" warnings (1.3 "compatibility" mode).
00076 // FL_DEPRECATED will be defined as a no-op.
00077
00078 // If FL_NO_DEPRECATED is not defined (default) FLTK 1.4 will issue several
00079 // "deprecated" warnings depending on the compiler in use: FL_DEPRECATED
00080 // will be defined according to the capabilities of the compiler (below).
00081 // The definition below this comment must match the one at the end of this file.
00082
00083 #if defined(FL_NO_DEPRECATED)
00084 #define FL_DEPRECATED(msg, func) func
00085 #endif
00086
00087 #ifdef __cplusplus
00088
00089 /*
00090  Declare macros specific to Visual Studio.
00091
00092  Visual Studio defines __cplusplus = '199711L' in all its versions which is
00093  not helpful for us here. For VS version number encoding see:
00094  https://learn.microsoft.com/en-us/cpp/preprocessor/predefined-macros
00095
00096  This document specifies that the macro _MSVC_LANG is defined since
00097  "Visual Studio 2015 Update 3" as 201402L (default) and undefined in
00098  earlier versions. It can be used to determine the C++ standard as
00099  specified by the /std:c++ compiler option:
00100
00101  - /std:c++14      201402L  (also if /std:c++ is not used)
00102  - /std:c++17      201703L
00103  - /std:c++20      202002L
00104  - /std:c++latest  a "higher, unspecified value" (docs of VS 2022)
00105
00106  As of this writing (02/2023) _MSVC_LANG is not yet used in this file
00107  but it is documented for future use.
00108 */
00109
00110 #if defined(_MSC_VER)
00111
00112 #if (_MSC_VER >= 1900) // Visual Studio 2015 (14.0)
00113 #endif // Visual Studio 2015 (14.0)
00114
00115 #if (_MSC_VER >= 1400) // Visual Studio 2005 (8.0)
00116 #ifndef FL_DEPRECATED
00117 #define FL_DEPRECATED(msg, func) __declspec(deprecated(msg)) func
00118 #endif
00119 #endif // Visual Studio 2005 (8.0)
00120
00121 #if (_MSC_VER >= 1310) // Visual Studio .NET 2003 (7.1)
00122 #ifndef FL_DEPRECATED
00123 #define FL_DEPRECATED(msg, func) __declspec(deprecated) func
00124 #endif
00125 #endif // Visual Studio .NET 2003 (7.1)
00126
00127 #endif // Visual Studio
00128
00129
00130 /*
00131  Declare macros specific to the C++ standard used.
00132
00133  Macros may have been declared already in previous sections.
00134 */
00135 #if (__cplusplus >= 202002L) // C++20
00136 #endif // C++20
00137
00138 #if (__cplusplus >= 201703L) // C++17
00139 #endif // C++17
00140
00141 #if (__cplusplus >= 201402L) // C++14
00142 #ifndef FL_DEPRECATED
00143 #define FL_DEPRECATED(msg, func) [[deprecated(msg)]] func

```

```

00144 #endif
00145 #endif // C++14
00146
00147 #if (__cplusplus >= 201103L) // C++11
00148 #endif // C++11
00149
00150 #if (__cplusplus >= 199711L) // C++89
00151 #endif // C++89
00152
00153 #endif // __cplusplus
00154
00155 /*
00156  Declare macros specific to clang
00157
00158  Macros may have been declared already in previous sections.
00159  */
00160 #if defined(__clang__)
00161
00162 #define FL_CLANG_VERSION (__clang_major__ * 10000 + __clang_minor__ * 100 + __clang_patchlevel__)
00163
00164 // -- nothing yet --
00165
00166 #endif /* __clang__ */
00167
00168 /*
00169  Declare macros specific to gcc.
00170
00171  Macros may have been declared already in previous sections.
00172  */
00173 #if defined(__GNUC__)
00174
00175 #define FL_GCC_VERSION (__GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__)
00176
00177 #ifndef __fl_attr
00178 #define __fl_attr(x) __attribute__((x))
00179 #endif
00180
00181 #if FL_GCC_VERSION > 40500 /* gcc 4.5.0 */
00182 #ifndef FL_DEPRECATED
00183 #define FL_DEPRECATED(msg, func) func __attribute__((deprecated(msg)))
00184 #endif
00185 #endif /* gcc 4.5.0 */
00186
00187 #if FL_GCC_VERSION >= 30400 /* gcc 3.4.0 */
00188 #ifndef FL_DEPRECATED
00189 #define FL_DEPRECATED(msg, func) func __attribute__((deprecated))
00190 #endif
00191 #endif /* gcc 3.4.0 */
00192
00193 #endif /* __GNUC__ */
00194
00195 /*
00196  If a macro was not defined in any of the sections above, set it to no-op here.
00197  */
00198 #ifndef __fl_attr
00199 #define __fl_attr(x)
00200 #endif
00201
00202 #ifndef FL_DEPRECATED
00203 #define FL_DEPRECATED(msg, func) func
00204 #endif
00205
00206 #endif /* !FL_fl_attr_h_ */

```

12.13 Fl_Bitmap.H

```

00001 //
00002 // Bitmap header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2017 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:

```

```

00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Bitmap widget . */
00019
00020 #ifndef Fl_Bitmap_H
00021 #define Fl_Bitmap_H
00022 #include "Fl_Image.H"
00023 #include "Fl_Widget.H" // for fl_uintptr_t
00024
00025 class Fl_Widget;
00026 struct Fl_Menu_Item;
00027
00032 class FL_EXPORT Fl_Bitmap : public Fl_Image {
00033     friend class Fl_Graphics_Driver;
00034 public:
00036     const uchar *array;
00038     int alloc_array;
00039
00040 private:
00042     fl_uintptr_t id_;
00043     int cache_w_, cache_h_; // size of bitmap when cached
00044
00045 public:
00048     Fl_Bitmap(const uchar *bits, int W, int H) :
00049         Fl_Image(W,H,0), array(bits), alloc_array(0), id_(0), cache_w_(0),cache_h_(0) {data((const char
00052     Fl_Bitmap(const char *bits, int W, int H) :
00053         Fl_Image(W,H,0), array((const uchar *)bits), alloc_array(0), id_(0), cache_w_(0),cache_h_(0)
00054     {data((const char **)&array, 1);}
00055     Fl_Bitmap(const uchar *bits, int bits_length, int W, int H);
00056     Fl_Bitmap(const char *bits, int bits_length, int W, int H);
00057     virtual ~Fl_Bitmap();
00058     Fl_Image *copy(int W, int H) const FL_OVERRIDE;
00059     Fl_Image *copy() const { return Fl_Image::copy(); }
00060     void draw(int X, int Y, int W, int H, int cx=0, int cy=0) FL_OVERRIDE;
00061     void draw(int X, int Y) {draw(X, Y, w(), h(), 0, 0);}
00062     void label(Fl_Widget*w) FL_OVERRIDE;
00063     void label(Fl_Menu_Item*m) FL_OVERRIDE;
00064     void uncache() FL_OVERRIDE;
00065     int cache_w() {return cache_w_;}
00066     int cache_h() {return cache_h_;}
00067 };
00068 #endif

```

12.14 Fl_BMP_Image.H

```

00001 //
00002 // BMP image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_BMP_Image widget . */
00019
00020 #ifndef Fl_BMP_Image_H
00021 #define Fl_BMP_Image_H
00022 #include "Fl_Image.H"
00023
00028 class FL_EXPORT Fl_BMP_Image : public Fl_RGB_Image {
00029
00030 public:
00031
00032     Fl_BMP_Image(const char* filename);
00033     Fl_BMP_Image(const char* imagename, const unsigned char *data, const long length = -1);
00034
00035 protected:
00036
00037     void load_bmp_(class Fl_Image_Reader &rdr, int ico_height = 0, int ico_width = 0);
00038
00039 };

```

```
00040
00041 #endif
```

12.15 Fl_Box.H File Reference

[Fl_Box](#) widget.

```
#include "Fl_Widget.H"
```

Classes

- class [Fl_Box](#)

This widget simply draws its box, and possibly its label.

12.15.1 Detailed Description

[Fl_Box](#) widget.

12.16 Fl_Box.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Box header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2025 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00021 #ifndef Fl_Box_H
00022 #define Fl_Box_H
00023
00024 #ifndef Fl_Widget_H
00025 #include "Fl_Widget.H"
00026 #endif
00027
00034 class FL_EXPORT Fl_Box : public Fl_Widget {
00035 protected:
00036     void draw() FL_OVERRIDE;
00037 public:
00054     Fl_Box(int X, int Y, int W, int H, const char *L = 0);
00055
00073     Fl_Box(Fl_Boxtype B, int X, int Y, int W, int H, const char *L);
00074
00075     int handle(int) FL_OVERRIDE;
00076 };
00077
00078 #endif
```

12.17 Fl_Browser.H

```
00001 //
00002 // Browser header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
```



```

00015 //
00016
00017 /* \file
00018     Fl_Browser widget . */
00019
00020 // Forms-compatible browser.  Probably useful for other
00021 // lists of textual data.  Notice that the line numbers
00022 // start from 1, and 0 means "no line".
00023
00024 #ifndef Fl_Browser_H
00025 #define Fl_Browser_H
00026
00027 #include "Fl_Browser_.H"
00028 #include "Fl_Image.H"
00029
00030 struct FL_BLINE;
00031
00086 class FL_EXPORT Fl_Browser : public Fl_Browser_ {
00087
00088     FL_BLINE *first;           // the array of lines
00089     FL_BLINE *last;
00090     FL_BLINE *cache;
00091     int cacheline;            // line number of cache
00092     int lines;                // Number of lines
00093     int full_height_;
00094     const int* column_widths_;
00095     char format_char_;        // alternative to @-sign
00096     char column_char_;        // alternative to tab
00097
00098 protected:
00099
00100     // required routines for Fl_Browser_ subclass:
00101     void* item_first() const FL_OVERRIDE;
00102     void* item_next(void* item) const FL_OVERRIDE;
00103     void* item_prev(void* item) const FL_OVERRIDE;
00104     void* item_last() const FL_OVERRIDE;
00105     int item_selected(void* item) const FL_OVERRIDE;
00106     void item_select(void* item, int val) FL_OVERRIDE;
00107     int item_height(void* item) const FL_OVERRIDE;
00108     int item_width(void* item) const FL_OVERRIDE;
00109     void item_draw(void* item, int X, int Y, int W, int H) const FL_OVERRIDE;
00110     int full_height() const FL_OVERRIDE;
00111     int incr_height() const FL_OVERRIDE;
00112     const char* item_text(void* item) const FL_OVERRIDE;
00118     void item_swap(void *a, void *b) FL_OVERRIDE { swap((FL_BLINE*)a, (FL_BLINE*)b); }
00124     void* item_at(int line) const FL_OVERRIDE { return (void*)find_line(line); }
00125
00126     FL_BLINE* find_line(int line) const ;
00127     FL_BLINE* _remove(int line) ;
00128     void insert(int line, FL_BLINE* item);
00129     int lineno(void* item) const ;
00130     void swap(FL_BLINE *a, FL_BLINE *b);
00131
00132 public:
00133
00134     void remove(int line);
00135     void add(const char* newtext, void* d = 0);
00136     void insert(int line, const char* newtext, void* d = 0);
00137     void move(int to, int from);
00138     int load(const char* filename);
00139     void swap(int a, int b);
00140     void clear();
00141
00147     int size() const { return lines; }
00148     void size(int W, int H) { Fl_Widget::size(W, H); }
00149
00153     Fl_Fontsize textsize() const { return Fl_Browser_::textsize(); }
00154
00155     /*
00156     Sets the default text size for the lines in the browser to newSize.
00157     Defined and documented in Fl_Browser.cxx
00158     */
00159     void textsize(Fl_Fontsize newSize);
00160
00161     int topline() const ;
00163     enum Fl_Line_Position { TOP, BOTTOM, MIDDLE };
00164     void lineposition(int line, Fl_Line_Position pos);
00171     void topline(int line) { lineposition(line, TOP); }
00178     void bottomline(int line) { lineposition(line, BOTTOM); }
00185     void middleline(int line) { lineposition(line, MIDDLE); }
00186
00187     int select(int line, int val=1);
00188     int selected(int line) const ;
00189     void show(int line);
00191     void show() FL_OVERRIDE { Fl_Widget::show(); }
00192     void hide(int line);
00194     void hide() FL_OVERRIDE { Fl_Widget::hide(); }

```

```

00195 int visible(int line) const ;
00196
00197 int value() const ;
00203 void value(int line) { select(line); }
00204 const char* text(int line) const ;
00205 void text(int line, const char* newtext);
00206 void* data(int line) const ;
00207 void data(int line, void* d);
00208
00209 Fl_Browser(int X, int Y, int W, int H, const char *L = 0);
00213 ~Fl_Browser() { clear(); }
00214
00246 char format_char() const { return format_char_; }
00252 void format_char(char c) { format_char_ = c; }
00258 char column_char() const { return column_char_; }
00265 void column_char(char c) { column_char_ = c; }
00289 const int* column_widths() const { return column_widths_; }
00294 void column_widths(const int* arr) { column_widths_ = arr; }
00295
00305 int displayed(int line) const { return Fl_Browser_::displayed(find_line(line)); }
00306
00314 void make_visible(int line) {
00315     if (line < 1) Fl_Browser_::display(find_line(1));
00316     else if (line > lines) Fl_Browser_::display(find_line(lines));
00317     else Fl_Browser_::display(find_line(line));
00318 }
00319
00320 // icon support
00321 void icon(int line, Fl_Image* icon);
00322 Fl_Image* icon(int line) const;
00323 void remove_icon(int line);
00324
00326 void replace(int a, const char* b) { text(a, b); }
00327 void display(int line, int val=1);
00328 };
00329
00330 #endif

```

12.18 Fl_Browser_.H

```

00001 //
00002 // Common browser header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2016 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Browser_ widget . */
00019
00020 // Yes, I know this should be a template...
00021
00022 #ifndef Fl_Browser__H
00023 #define Fl_Browser__H
00024
00025 #ifndef Fl_Group_H
00026 #include "Fl_Group.H"
00027 #endif
00028 #include "Fl_Scrollbar.H"
00029 #include <FL/Fl.H> // Fl::scrollbar_size()
00030
00031 #define FL_NORMAL_BROWSER      0
00032 #define FL_SELECT_BROWSER     1
00033 #define FL_HOLD_BROWSER       2
00034 #define FL_MULTIPLE_BROWSER   3
00035 #define FL_SORT_ASCENDING     0
00036 #define FL_SORT_DESCENDING    1
00037 #define FL_SORT_CASEINSENSITIVE 0x2
00038 #define FL_EXPORT Fl_Browser_ : public Fl_Group {
00039     int position_; // where user wants it scrolled to
00040     int real_position_; // the current vertical scrolling position
00041     int hposition_; // where user wants it panned to
00042     int real_hposition_; // the current horizontal scrolling position
00043     int offset_; // how far down top_ item the real_position is
00044     int max_width; // widest object seen so far
00045     uchar has_scrollbar_; // which scrollbars are enabled

```

```

00091 Fl_Font textfont_;
00092 Fl_Fonsize textsize_;
00093 Fl_Color textcolor_;
00094 void* top_; // which item scrolling position is in
00095 void* selection_; // which is selected (except for FL_MULTI_BROWSER)
00096 void *redraw1,*redraw2; // minimal update pointers
00097 void* max_width_item; // which item has max_width_
00098 int scrollbar_size_; // size of scrollbar trough
00099 int linespacing_;
00100
00101 void update_top();
00102
00103 protected:
00104
00105 // All of the following must be supplied by the subclass:
00111 virtual void *item_first() const = 0;
00112 virtual void *item_next(void *item) const = 0;
00123 virtual void *item_prev(void *item) const = 0;
00129 virtual void *item_last() const { return 0L; }
00138 virtual int item_height(void *item) const = 0;
00146 virtual int item_width(void *item) const = 0;
00147 virtual int item_quick_height(void *item) const ;
00152 virtual void item_draw(void *item,int X,int Y,int W,int H) const = 0;
00158 virtual const char *item_text(void *item) const { (void)item; return 0L; }
00164 virtual void item_swap(void *a,void *b) { (void)a; (void)b; }
00171 virtual void *item_at(int index) const { (void)index; return 0L; }
00172 // you don't have to provide these but it may help speed it up:
00173 virtual int full_width() const ; // current width of all items
00174 virtual int full_height() const ; // current height of all items
00175 virtual int incr_height() const ; // average height of an item
00176 // These only need to be done by subclass if you want a multi-browser:
00177 virtual void item_select(void *item,int val=1);
00178 virtual int item_selected(void *item) const ;
00179
00180 // things the subclass may want to call:
00184 void *top() const { return top_; }
00192 void *selection() const { return selection_; }
00193 void new_list(); // completely clobber all data, as though list replaced
00194 void deleting(void *item); // get rid of any pointers to item
00195 void replacing(void *a,void *b); // change a pointers to b
00196 void swapping(void *a,void *b); // exchange pointers a and b
00197 void inserting(void *a,void *b); // insert b near a
00198 int displayed(void *item) const ; // true if this item is visible
00199 void redraw_line(void *item); // minimal update, no change in size
00204 void redraw_lines() { damage(FL_DAMAGE_SCROLL); } // redraw all of them
00205 void bbox(int &X,int &Y,int &W,int &H) const;
00206 int leftedge() const; // x position after scrollbar & border
00207 void *find_item(int ypos); // item under mouse
00208
00209 void draw() FL_OVERRIDE;
00210 Fl_Browser_(int X,int Y,int W,int H,const char *L=0);
00211
00212 public:
00213
00229 Fl_Scrollbar scrollbar;
00233 Fl_Scrollbar hscrollbar;
00234
00235 int handle(int event) FL_OVERRIDE;
00236 void resize(int X,int Y,int W,int H) FL_OVERRIDE;
00237
00238 int select(void *item,int val=1,int docallbacks=0);
00239 int select_only(void *item,int docallbacks=0);
00240 int deselect(int docallbacks=0);
00248 int vposition() const { return position_; }
00249 FL_DEPRECATED("since 1.4.0 - use vposition() instead",
00250 int position() const) { return vposition(); }
00251
00252 void vposition(int pos); // scroll to here
00253 FL_DEPRECATED("since 1.4.0 - use vposition(pos) instead",
00254 void position(int pos)) { return vposition(pos); }
00255 void position(int x, int y) { Fl_Group::position(x, y); }
00256
00264 int hposition() const { return hposition_; }
00265 void hposition(int); // pan to here
00266 void display(void *item); // scroll so this item is shown
00267
00277 enum { // values for has_scrollbar()
00278 HORIZONTAL = 1,
00279 VERTICAL = 2,
00280 BOTH = 3,
00281 ALWAYS_ON = 4,
00282 HORIZONTAL_ALWAYS = 5,
00283 VERTICAL_ALWAYS = 6,
00284 BOTH_ALWAYS = 7
00285 };
00289 uchar has_scrollbar() const { return has_scrollbar_; }
00312 void has_scrollbar(uchar mode) { has_scrollbar_ = mode; }

```

```

00313
00318 Fl_Font textfont() const { return textfont_; }
00322 void textfont(Fl_Font font) { textfont_ = font; }
00323
00327 Fl_Fontsize textsize() const { return textsize_; }
00331 void textsize(Fl_Fontsize newSize) { textsize_ = newSize; }
00332
00336 Fl_Color textcolor() const { return textcolor_; }
00340 void textcolor(Fl_Color col) { textcolor_ = col; }
00341
00351 int scrollbar_size() const {
00352     return(scrollbar_size_);
00353 }
00373 void scrollbar_size(int newSize) {
00374     scrollbar_size_ = newSize;
00375 }
00381 int scrollbar_width() const {
00382     return(Fl::scrollbar_size());
00383 }
00389 void scrollbar_width(int width) {
00390     Fl::scrollbar_size(width);
00391     scrollbar_size_ = 0;
00392 }
00397 void scrollbar_right() { scrollbar.align(FL_ALIGN_RIGHT); }
04002 void scrollbar_left() { scrollbar.align(FL_ALIGN_LEFT); }
04003 void sort(int flags=0);
04004
04009 void linespacing(int pixels) { linespacing_ = pixels; }
04010
04014 int linespacing() const { return linespacing_; }
04015 };
04016
04017 #endif

```

12.19 Fl_Button.H

```

00001 //
00002 // Button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2014 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Button widget . */
00019
00020 #ifndef Fl_Button_H
00021 #define Fl_Button_H
00022
00023 #ifndef Fl_Widget_H
00024 #include "Fl_Widget.H"
00025 #endif
00026
00027 // values for type()
00028 #define FL_NORMAL_BUTTON      0
00030 #define FL_TOGGLE_BUTTON      1
00031 #define FL_RADIO_BUTTON       (FL_RESERVED_TYPE+2)
00034 #define FL_HIDDEN_BUTTON      3
00035
00036 extern FL_EXPORT Fl_Shortcut fl_old_shortcut(const char*);
00037
00038 class Fl_Widget_Tracker;
00039
00076 class FL_EXPORT Fl_Button : public Fl_Widget {
00077
00078     int shortcut_;
00079     char value_;
00080     char oldval;
00081     uchar down_box_;
00082     uchar compact_;
00083
00084 protected:
00085
00086     static Fl_Widget_Tracker *key_release_tracker;
00087     static void key_release_timeout(void*);
00088     void simulate_key_action();

```

```

00089
00090     void draw() FL_OVERRIDE;
00091
00092 public:
00093
00094     int handle(int) FL_OVERRIDE;
00095
00096     Fl_Button(int X, int Y, int W, int H, const char *L = 0);
00097
00098     int value(int v);
00099
00103     char value() const {return value_;}
00104
00109     int set() {return value(1);}
00110
00115     int clear() {return value(0);}
00116
00117     void setonly(); // this should only be called on FL_RADIO_BUTTONs
00118
00123     int shortcut() const {return shortcut_;}
00124
00144     void shortcut(int s) {shortcut_ = s;}
00145
00150     Fl_Boxtype down_box() const {return (Fl_Boxtype)down_box_;}
00151
00161     void down_box(Fl_Boxtype b) {down_box_ = b;}
00162
00164     void shortcut(const char *s) {shortcut(fl_old_shortcut(s));}
00165
00167     Fl_Color down_color() const {return selection_color();}
00168
00170     void down_color(unsigned c) {selection_color(c);}
00171
00172     // handle flag for compact buttons, documentation in source code
00173     void compact(uchar v);
00174
00178     uchar compact() { return compact_; }
00179 };
00180
00181 #endif

```

12.20 Fl_Cairo.H File Reference

Cairo is currently supported for the following platforms: Windows, macOS, Unix/Linux (X11 + Wayland).

```

#include <FL/Fl.H>
#include <cairo.h>

```

Classes

- class [Fl_Cairo_State](#)

Contains all the necessary info on the current cairo context.

12.20.1 Detailed Description

Cairo is currently supported for the following platforms: Windows, macOS, Unix/Linux (X11 + Wayland).

Note

In FLTK 1.3.x this header file ([Fl_Cairo.H](#)) included the platform specific Cairo headers. This is no longer true since 1.4.0.

This header file is platform agnostic. If you need platform specific Cairo headers you need to `#include` them in your source file.

To use FLTK's builtin Cairo support you need to `#include <FL/Fl.H>` **before** you include any other FLTK header which is officially required anyway. Since FLTK 1.4.0 the preprocessor constants `FLTK_HAVE_CAIRO` and/or `FLTK_HAVE_CAIROEXT` are defined in `<FL/Fl.H>` by including `<FL/fl_config.h>`.

12.21 Fl_Cairo.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Main Cairo support header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00033 #ifndef FL_CAIRO_H
00034 #define FL_CAIRO_H
00035
00036 #include <FL/Fl.H>
00037
00038 # ifdef FLTK_HAVE_CAIRO
00039 #   include <cairo.h>
00041
00056 class FL_EXPORT Fl_Cairo_State {
00057 public:
00058     Fl_Cairo_State()
00059         : cc_(0)
00060         , own_cc_(false)
00061         , autolink_(false)
00062         , window_(0)
00063         , gc_(0) {}
00064
00065     // access attributes
00066     cairo_t *cc() const { return cc_; }
00067     bool autolink() const { return autolink_; }
00076     void cc(cairo_t *c, bool own = true) {
00077         if (cc_ && own_cc_)
00078             cairo_destroy(cc_);
00079         cc_ = c;
00080         if (!cc_)
00081             window_ = 0;
00082         own_cc_ = own;
00083     }
00084     void autolink(bool b);
00085     void window(void *w) { window_ = w; }
00086     void *window() const { return window_; }
00087     void gc(void *c) { gc_ = c; }
00088     void *gc() const { return gc_; }
00089
00090 private:
00091     cairo_t *cc_;           // contains the unique autoupdated cairo context
00092     bool own_cc_;          // indicates whether we must delete the cc, useful for internal cleanup
00093     bool autolink_;        // false by default, prevents the automatic cairo mapping on fltk windows
00094                             // for custom cairo implementations.
00095     void *window_, *gc_;   // for keeping track internally of last win+gc treated
00096 };
00097
00100 #endif // FLTK_HAVE_CAIRO
00101 #endif // FL_CAIRO_H

```

12.22 Fl_Cairo_Window.H File Reference

[Fl_Cairo_Window](#), an FLTK window incorporating a Cairo draw callback.

```

#include <FL/fl_config.h>
#include <FL/Fl.H>
#include <FL/Fl_Double_Window.H>

```

Classes

- class [Fl_Cairo_Window](#)
This defines an FLTK window with Cairo support.

12.22.1 Detailed Description

[Fl_Cairo_Window](#), an FLTK window incorporating a Cairo draw callback.

12.23 Fl_Cairo_Window.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Fl_Cairo_Window header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00021 #ifndef FL_CAIRO_WINDOW_H
00022 # define FL_CAIRO_WINDOW_H
00023
00024 #include <FL/fl_config.h>
00025
00026 # ifdef FLTK_HAVE_CAIRO
00027
00028 // Cairo is currently supported for the following platforms:
00029 // Win32, Apple Quartz, X11, Wayland
00030
00031 # include <FL/Fl.H>
00032 # include <FL/Fl_Double_Window.H>
00033
00089 class FL_EXPORT Fl_Cairo_Window : public Fl_Double_Window {
00090
00091 public:
00092     Fl_Cairo_Window(int W, int H, const char *L = 0)
00093         : Fl_Double_Window(W, H, L), draw_cb_(0) {}
00094     Fl_Cairo_Window(int X, int Y, int W, int H, const char *L = 0)
00095         : Fl_Double_Window(X, Y, W, H, L), draw_cb_(0) {}
00096
00097 protected:
00098     void draw() FL_OVERRIDE {
00099         Fl_Double_Window::draw();
00100         if (draw_cb_) { // call the Cairo draw callback
00101             // manual method ? if yes explicitly get a cairo_context here
00102             if (!Fl::cairo_autolink_context())
00103                 Fl::cairo_make_current(this);
00104             draw_cb_(this, Fl::cairo_cc());
00105             // flush Cairo drawings: necessary at least for Windows
00106             Fl::cairo_flush(Fl::cairo_cc());
00107         }
00108     }
00109
00110 public:
00111     typedef void (*cairo_draw_cb) (Fl_Cairo_Window* self, cairo_t* def);
00112     void set_draw_cb(cairo_draw_cb cb) { draw_cb_ = cb; }
00113 private:
00114     cairo_draw_cb draw_cb_;
00115 };
00116
00127 # endif // FLTK_HAVE_CAIRO
00128 #endif // FL_CAIRO_WINDOW_H
```

12.24 fl_callback_macros.H File Reference

This file provides macros for easy function and method callbacks with multiple type safe arguments.

```
#include <stdlib.h>
```

Macros

- `#define FL_FUNCTION_CALLBACK_3(WIDGET, FUNC, TYPE0, VALUE0, TYPE1, VALUE1, TYPE2, VALUE2)`
Declare a C function callback with custom parameters.
- `#define FL_INLINE_CALLBACK_2(WIDGET, TYPE0, NAME0, VALUE0, TYPE1, NAME1, VALUE1, LAMBDA)`

Creates code to declare a callback function in line with instantiating a widget.

- `#define FL_METHOD_CALLBACK_1(WIDGET, CLASS, SELF, METH, TYPE0, VALUE0)`

Declare a non-static class method callback with custom parameters.

12.24.1 Detailed Description

This file provides macros for easy function and method callbacks with multiple type safe arguments.

12.24.2 Macro Definition Documentation

12.24.2.1 FL_FUNCTION_CALLBACK_3

```
#define FL_FUNCTION_CALLBACK_3(
    WIDGET,
    FUNC,
    TYPE0,
    VALUE0,
    TYPE1,
    VALUE1,
    TYPE2,
    VALUE2)
```

Declare a C function callback with custom parameters.

You can declare a plain C function callback or a static method callback with custom parameters using this macro. It simplifies the process of calling arbitrary functions with up to five custom parameters. The macro generates code that ensures type safety and expands FLTK's standard callbacks, which are limited to a single `void*` or `long` argument.

To use the macro, you provide the widget that will handle the callback as the first argument. The second argument can be either a regular function or a static method in any class.

Following these arguments, you can include up to five pairs, where each pair consists of a type and a value. For example, `int, 3` specifies an integer parameter with a value of 3. If you need to pass two arguments, you can use two pairs, like this: `int, 3, int, 4`. The last digit of the macro name must be the same as the number of pairs (0..5)

Whenever the code generated by the macro is called, the custom parameters are duplicated and marked for automatic deallocation using `delete` when the callback widget is destroyed.

```
#include <FL/fl_callback_macros.H>
...
Fl_Button *btn1 = new Fl_Button(10, 10, 100, 20, "Beep");
FL_FUNCTION_CALLBACK_0(btn1, fl_beep);
...
Fl_Button *btn2 = new Fl_Button(10, 40, 100, 20, "Hello");
FL_FUNCTION_CALLBACK_5(btn2,
    fl_message,
    const char *, "Hello\n%d %d %d %d",
    int, 1, int, 2, int, 3, int, 4
);
```

You can find a small demonstration program showcasing the usage of `FL_*_CALLBACK_*` in the `examples/callbacks.cxx` file.

Parameters

<i>WIDGET</i>	the widget that will call the callback
<i>FUNC</i>	a C/C++ function or a static class method
<i>TYPE0,VALUE0,TYPE1,VALUE1,TYPE2,VALUE2</i>	a list of zero to five type/value pairs, all separated by commas

See also

[FL_METHOD_CALLBACK_1](#), [FL_INLINE_CALLBACK_2](#)

12.24.2.2 FL_INLINE_CALLBACK_2

```
#define FL_INLINE_CALLBACK_2(
    WIDGET,
```



```

    TYPE0,
    NAME0,
    VALUE0,
    TYPE1,
    NAME1,
    VALUE1,
    LAMBDA)

```

Creates code to declare a callback function in line with instantiating a widget.

You can use this macro to create a function as a callback, allowing you to define the callback function right where the widget and callback are declared, similar to a Lambda function.

The first argument of the macro specifies the widget that will handle the callback. Next, you can include up to five triplets, where each triplet consists of a type, a parameter name, and a value. For example, `int, x, 3` specifies an integer parameter with a value of 3. If you need to pass two arguments, you can use two triplets, such as `int, x, 3, int, y, 4`. The last digit of the macro name must be the same as the number of triplets (0..5).

The last argument is the actual function body itself.

The function body is limited to a syntax that the macro preprocessor can handle. It should include the leading '{' and trailing '}' and may contain local variable declarations, use global variables and functions, and use also the variables listed and initialized in the argument triples of the macro. Very large function bodies should be avoided because they may exceed the admissible size of a macro argument.

Whenever the code generated by the macro is called, the custom parameters are duplicated and marked for automatic deallocation using `delete` when the callback widget is destroyed.

```

#include <FL/fl_callback_macros.H>
...
Fl_Button *btn = new Fl_Button(10, 10, 100, 20, "Test");
FL_INLINE_CALLBACK_1(btn,
    const char *, name, btn->label(),
    {
        fl_message("Greetings from the %s button", name);
    }
);

```

You can find a small demonstration program showcasing the usage of `FL*_CALLBACK_*` in the `examples/callbacks.cxx` file.

Parameters

<i>WIDGET</i>	the widget that will call the callback
<i>TYPE0</i>	the type of the first parameter in the function call
<i>NAME0</i>	an arbitrary variable name that can be used as a parameter in the function body
<i>VALUE0</i>	a constant value or a variable; the value of the variable is copied when the callback is created
<i>TYPE1,NAME1,VALUE1</i>	as above; there are six macros that support 0 to 5 parameters
<i>LAMBDA</i>	the function body within the limits of the C macro preprocessor

See also

[FL_METHOD_CALLBACK_1](#), [FL_FUNCTION_CALLBACK_3](#)

12.24.2.3 FL_METHOD_CALLBACK_1

```

#define FL_METHOD_CALLBACK_1(
    WIDGET,
    CLASS,
    SELF,
    METH,
    TYPE0,
    VALUE0)

```

Declare a non-static class method callback with custom parameters.

You can declare a callback for a non-static class method with custom parameters using this macro. It provides a convenient way to call arbitrary methods in any class, overcoming FLTK's limitation of passing only a single `void*` or `long` argument. Furthermore, it ensures type safety.

The first argument of the macro specifies the widget that will handle the callback. The second argument indicates the class type to be called. The third argument must be a pointer to an instance of that class. The fourth argument is the name of the method within the class. That method must be public and should not be static.

Following these arguments, you can include up to five pairs, where each pair consists of a type and a value. For example, `int, 3` specifies an integer parameter with a value of 3. If you need to pass two arguments, you can use two pairs, like this: `int, 3, int, 4`. The last digit of the macro name must be the same as the number of pairs (0..5)

Whenever the code generated by the macro is called, the custom parameters are duplicated and marked for automatic deallocation using `delete` when the callback widget is destroyed.

```
#include <FL/fl_callback_macros.H>
```

```
...
```

```
Fl_Button *btn = new Fl_Button(10, 10, 100, 20, "Test");
```

```
FL_METHOD_CALLBACK_1(btn, Fl_Button, btn, color, Fl_Color, FL_GREEN);
```

You can find a small demonstration program showcasing the usage of `FL*_CALLBACK_*` in the `examples/callbacks.cxx` file.

Parameters

<i>WIDGET</i>	the widget that will call the callback
<i>CLASS</i>	the class type
<i>SELF</i>	a pointer to an instance of the class
<i>METH</i>	a C++ class method that must be public and not static
<i>TYPE0,VALUE0</i>	a list of zero to five type/value pairs, all separated by commas

See also

[FL_FUNCTION_CALLBACK_3](#), [FL_INLINE_CALLBACK_2](#)

12.25 fl_callback_macros.H

[Go to the documentation of this file.](#)

```
00001 /*
00002  * Macros for easy callbacks for the Fast Light Tool Kit (FLTK).
00003  *
00004  * Copyright 2023 by Bill Spitzak and others.
00005  *
00006  * This library is free software. Distribution and use rights are outlined in
00007  * the file "COPYING" which should have been included with this file. If this
00008  * file is missing or damaged, see the license at:
00009  *
00010  *     https://www.fltk.org/COPYING.php
00011  *
00012  * Please see the following page on how to report bugs and issues:
00013  *
00014  *     https://www.fltk.org/bugs.php
00015  */
00016
00017 #ifndef _FL_FL_CALLBACK_MACROS_H_
00018 #define _FL_FL_CALLBACK_MACROS_H_
00019
00020 #include <stdlib.h>
00021
00022 #ifdef FL_DOXYGEN
00023
00024 #define FL_FUNCTION_CALLBACK_3(WIDGET, FUNC, TYPE0, VALUE0, TYPE1, VALUE1, TYPE2, VALUE2)
00025
00026 #define FL_METHOD_CALLBACK_1(WIDGET, CLASS, SELF, METH, TYPE0, VALUE0)
00027
00028 #define FL_INLINE_CALLBACK_2(WIDGET, TYPE0, NAME0, VALUE0, TYPE1, NAME1, VALUE1, LAMBDA)
00029
00030 #else // FL_DOXYGEN
00031
00032 /*
00033  * These two macros make it possible to call macros with names that are created
00034  * by concatenating the name in x and (in this context) the number in y.
00035  */
00036 #define _FL_CBD_CONCAT_IMPL(x, y) x##y
00037 #define _FL_CBD_CONCAT(x, y) _FL_CBD_CONCAT_IMPL(x, y)
00038
00039 /*
00040  * Create a unique name for the derived class based on the current source code
00041  * line number.
00042  */
00043 #endif
00044
```

```

00189  */
00190 #define _FL_CBD_CLASS_NAME _FL_CBD_CONCAT(Fl_Callback_User_Data_,__LINE__)
00191
00192
00193 /*
00194 These macros create boilerplate code for callbacks to functions and
00195 static class methods with up to five arguments.
00196
00197 This macro invocation for example
00198 ```
00199 FL_FUNCTION_CALLBACK_2( func_cb_btn_2, hello_2_args_cb,
00200                        const char *, text, "FLTK",
00201                        int, number, 2 );
00202 ```
00203 will generate the following code:
00204 ```
00205
00206 do {
00207     class Fl_Callback_User_Data_92 : public Fl_Callback_User_Data {
00208     public:
00209         const char * p0_;
00210         int p1_;
00211         static void cb(Fl_Widget *w, void *user_data) {
00212             Fl_Callback_User_Data_92 *d = (Fl_Callback_User_Data_92*)user_data;
00213             hello_2_args_cb(d->p0_, d->p1_);
00214         };
00215         Fl_Callback_User_Data_92(const char * p0, int p1)
00216         : p0_(p0),
00217           p1_(p1)
00218         { }
00219     };
00220     func_cb_btn_2->callback(Fl_Callback_User_Data_92::cb,
00221                           new Fl_Callback_User_Data_92("FLTK", 2),
00222                           true);
00223 } while(0)
00224 ```
00225
00226 Clicking the Fl_Button `func_cb_btn_2` will call `hello_2_args_cb("FLTK", 2)`.
00227 Deleting the button will also delete the data that was created in our
00228 boilerplate code.
00229 */
00230 #define FL_FUNCTION_CALLBACK_5(WIDGET, FUNC, TYPE0, VALUE0, TYPE1, VALUE1, TYPE2, VALUE2, TYPE3,
VALUE3, TYPE4, VALUE4) \
00231 do { \
00232     class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00233     public: \
00234         TYPE0 p0_; TYPE1 p1_; TYPE2 p2_; TYPE3 p3_; TYPE4 p4_; \
00235         static void cb(Fl_Widget *w, void *user_data) { \
00236             _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00237             FUNC(d->p0_, d->p1_, d->p2_, d->p3_, d->p4_); \
00238         }; \
00239         _FL_CBD_CLASS_NAME(TYPE0 p0, TYPE1 p1, TYPE2 p2, TYPE3 p3, TYPE4 p4) \
00240         : p0_(p0), p1_(p1), p2_(p2), p3_(p3), p4_(p4) { }; \
00241     }; \
00242     WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0, VALUE1, VALUE2, VALUE3,
VALUE4), true); \
00243 } while(0)
00244
00245 #define FL_FUNCTION_CALLBACK_4(WIDGET, FUNC, TYPE0, VALUE0, TYPE1, VALUE1, TYPE2, VALUE2, TYPE3,
VALUE3) \
00246 do { \
00247     class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00248     public: \
00249         TYPE0 p0_; TYPE1 p1_; TYPE2 p2_; TYPE3 p3_; \
00250         static void cb(Fl_Widget *w, void *user_data) { \
00251             _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00252             FUNC(d->p0_, d->p1_, d->p2_, d->p3_); \
00253         }; \
00254         _FL_CBD_CLASS_NAME(TYPE0 p0, TYPE1 p1, TYPE2 p2, TYPE3 p3) \
00255         : p0_(p0), p1_(p1), p2_(p2), p3_(p3) { }; \
00256     }; \
00257     WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0, VALUE1, VALUE2, VALUE3),
true); \
00258 } while(0)
00259
00260 #define FL_FUNCTION_CALLBACK_3(WIDGET, FUNC, TYPE0, VALUE0, TYPE1, VALUE1, TYPE2, VALUE2) \
00261 do { \
00262     class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00263     public: \
00264         TYPE0 p0_; TYPE1 p1_; TYPE2 p2_; \
00265         static void cb(Fl_Widget *w, void *user_data) { \
00266             _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00267             FUNC(d->p0_, d->p1_, d->p2_); \
00268         }; \
00269         _FL_CBD_CLASS_NAME(TYPE0 p0, TYPE1 p1, TYPE2 p2) \
00270         : p0_(p0), p1_(p1), p2_(p2) { }; \
00271     }; \

```

```

00272     WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0, VALUE1, VALUE2), true); \
00273 } while(0)
00274
00275 #define FL_FUNCTION_CALLBACK_2(WIDGET, FUNC, TYPE0, VALUE0, TYPE1, VALUE1) \
00276 do { \
00277     class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00278     public: \
00279         TYPE0 p0_; TYPE1 p1_; \
00280         static void cb(Fl_Widget *w, void *user_data) { \
00281             _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00282             FUNC(d->p0_, d->p1_); \
00283         }; \
00284         _FL_CBD_CLASS_NAME(TYPE0 p0, TYPE1 p1) \
00285         : p0_(p0), p1_(p1) { }; \
00286     }; \
00287     WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0, VALUE1), true); \
00288 } while(0)
00289
00290 #define FL_FUNCTION_CALLBACK_1(WIDGET, FUNC, TYPE0, VALUE0) \
00291 do { \
00292     class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00293     public: \
00294         TYPE0 p0_; \
00295         static void cb(Fl_Widget *w, void *user_data) { \
00296             _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00297             FUNC(d->p0_); \
00298         }; \
00299         _FL_CBD_CLASS_NAME(TYPE0 p0) \
00300         : p0_(p0) { }; \
00301     }; \
00302     WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0), true); \
00303 } while(0)
00304
00305 #define FL_FUNCTION_CALLBACK_0(WIDGET, FUNC) \
00306 do { \
00307     class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00308     public: \
00309         static void cb(Fl_Widget *w, void *user_data) { \
00310             FUNC(); \
00311         }; \
00312         _FL_CBD_CLASS_NAME() { }; \
00313     }; \
00314     WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(), true); \
00315 } while(0)
00316
00317 /*
00318 These macros create boilerplate code for callbacks to class methods
00319 with up to five arguments.
00320
00321 This macro invocation for example
00322 ```
00323 FL_METHOD_CALLBACK_4(btn,
00324                     MyWindow, win, resize,
00325                     int, test_x+10,
00326                     int, test_y+10,
00327                     int, 320,
00328                     int, 400);
00329 ```
00330 will generate the following code:
00331 ```
00332 ```
00333 do {
00334     class Fl_Callback_User_Data_73 : public Fl_Callback_User_Data {
00335     public:
00336         int p0_;
00337         int p1_;
00338         int p2_;
00339         int p3_;
00340         MyWindow *self_;
00341         static void cb(Fl_Widget *w, void *user_data) {
00342             Fl_Callback_User_Data_73 *d = (Fl_Callback_User_Data_73*)user_data;
00343             d->self_->resize(d->p0_, d->p1_, d->p2_, d->p3_);
00344         };
00345         Fl_Callback_User_Data_73(MyWindow *self, int p0, int p1, int p2, int p3)
00346         : self_(self), p0_(p0), p1_(p1), p2_(p2), p3_(p3) { }
00347     };
00348     btn->callback(Fl_Callback_User_Data_73::cb,
00349                 new Fl_Callback_User_Data_73(win, test_x+10, test_y+10, 320, 400),
00350                 true);
00351 } while(0);
00352 ```
00353
00354 Clicking the Fl_Button `btn` will call
00355 `win->resize(test_x+10, test_y+10, 320, 400);`.
00356 Deleting the button will also delete the data that was created in our
00357 boilerplate code.
00358 */

```

```

00359
00360 #define FL_METHOD_CALLBACK_5(WIDGET, CLASS, SELF, METHOD, TYPE0, VALUE0, TYPE1, VALUE1, TYPE2, VALUE2,
    TYPE3, VALUE3, TYPE4, VALUE4) \
00361     do { \
00362         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00363             public: \
00364                 CLASS *self_; \
00365                 TYPE0 p0_; TYPE1 p1_; TYPE2 p2_; TYPE3 p3_; TYPE4 p4_; \
00366                 static void cb(Fl_Widget *w, void *user_data) { \
00367                     _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00368                     d->self_->METHOD(d->p0_, d->p1_, d->p2_, d->p3_, d->p4_); \
00369                 }; \
00370                 _FL_CBD_CLASS_NAME(CLASS *self, TYPE0 p0, TYPE1 p1, TYPE2 p2, TYPE3 p3, TYPE4 p4) \
00371                 : self_(self), p0_(p0), p1_(p1), p2_(p2), p3_(p3), p4_(p4) { }; \
00372             }; \
00373             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(SELF, VALUE0, VALUE1, VALUE2,
    VALUE3, VALUE4), true); \
00374         } while(0)
00375
00376 #define FL_METHOD_CALLBACK_4(WIDGET, CLASS, SELF, METHOD, TYPE0, VALUE0, TYPE1, VALUE1, TYPE2, VALUE2,
    TYPE3, VALUE3) \
00377     do { \
00378         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00379             public: \
00380                 CLASS *self_; \
00381                 TYPE0 p0_; TYPE1 p1_; TYPE2 p2_; TYPE3 p3_; \
00382                 static void cb(Fl_Widget *w, void *user_data) { \
00383                     _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00384                     d->self_->METHOD(d->p0_, d->p1_, d->p2_, d->p3_); \
00385                 }; \
00386                 _FL_CBD_CLASS_NAME(CLASS *self, TYPE0 p0, TYPE1 p1, TYPE2 p2, TYPE3 p3) \
00387                 : self_(self), p0_(p0), p1_(p1), p2_(p2), p3_(p3) { }; \
00388             }; \
00389             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(SELF, VALUE0, VALUE1, VALUE2,
    VALUE3), true); \
00390         } while(0)
00391
00392 #define FL_METHOD_CALLBACK_3(WIDGET, CLASS, SELF, METHOD, TYPE0, VALUE0, TYPE1, VALUE1, TYPE2, VALUE2)
    \
00393     do { \
00394         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00395             public: \
00396                 CLASS *self_; \
00397                 TYPE0 p0_; TYPE1 p1_; TYPE2 p2_; \
00398                 static void cb(Fl_Widget *w, void *user_data) { \
00399                     _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00400                     d->self_->METHOD(d->p0_, d->p1_, d->p2_); \
00401                 }; \
00402                 _FL_CBD_CLASS_NAME(CLASS *self, TYPE0 p0, TYPE1 p1, TYPE2 p2) \
00403                 : self_(self), p0_(p0), p1_(p1), p2_(p2) { }; \
00404             }; \
00405             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(SELF, VALUE0, VALUE1, VALUE2),
    true); \
00406         } while(0)
00407
00408 #define FL_METHOD_CALLBACK_2(WIDGET, CLASS, SELF, METHOD, TYPE0, VALUE0, TYPE1, VALUE1) \
00409     do { \
00410         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00411             public: \
00412                 CLASS *self_; \
00413                 TYPE0 p0_; TYPE1 p1_; \
00414                 static void cb(Fl_Widget *w, void *user_data) { \
00415                     _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00416                     d->self_->METHOD(d->p0_, d->p1_); \
00417                 }; \
00418                 _FL_CBD_CLASS_NAME(CLASS *self, TYPE0 p0, TYPE1 p1) \
00419                 : self_(self), p0_(p0), p1_(p1) { }; \
00420             }; \
00421             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(SELF, VALUE0, VALUE1), true); \
00422         } while(0)
00423
00424 #define FL_METHOD_CALLBACK_1(WIDGET, CLASS, SELF, METHOD, TYPE0, VALUE0) \
00425     do { \
00426         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00427             public: \
00428                 CLASS *self_; \
00429                 TYPE0 p0_; \
00430                 static void cb(Fl_Widget *w, void *user_data) { \
00431                     _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00432                     d->self_->METHOD(d->p0_); \
00433                 }; \
00434                 _FL_CBD_CLASS_NAME(CLASS *self, TYPE0 p0) \
00435                 : self_(self), p0_(p0) { }; \
00436             }; \
00437             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(SELF, VALUE0), true); \
00438         } while(0)
00439

```

```

00440 #define FL_METHOD_CALLBACK_0(WIDGET, CLASS, SELF, METHOD) \
00441 do { \
00442     class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00443     public: \
00444         CLASS *self_; \
00445         static void cb(Fl_Widget *w, void *user_data) { \
00446             _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00447             d->self_->METHOD(); \
00448         }; \
00449         _FL_CBD_CLASS_NAME(CLASS *self) \
00450         : self_(self) { }; \
00451     }; \
00452     WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(SELF), true); \
00453 } while(0)
00454
00455 /*
00456 These macros create boilerplate code for callback functions inlined into
00457 the widget creation code (similar to lambda functions in C++11 and up)
00458 with up to five arguments.
00459
00460 This macro invocation for example
00461 ```
00462 FL_INLINE_CALLBACK_2(           // callback has two parameters
00463     btn,                        // attach callback to this button
00464     const char *, text, "FLTK", // first parameter (type, name, value)
00465     int, number, 2,             // second parameter
00466     {                           // function body
00467         fl_message("We received the message %s with %d!", text, number);
00468     }
00469 );
00470 ```
00471 will generate the following code:
00472 ```
00473 do {
00474     class Fl_Callback_User_Data_133 : public Fl_Callback_User_Data {
00475     public:
00476         const char * p0_;        // store first parameter here
00477         int p1_;                 // store second parameter here
00478         // lambda style function
00479         static void fn(const char * text, int number ) {
00480             fl_message("We received the message %s with %d!", text, number);
00481         };
00482         // FLTK style callback
00483         static void cb(Fl_Widget *w, void *user_data) {
00484             Fl_Callback_User_Data_133 *d = (Fl_Callback_User_Data_133*)user_data;
00485             fn(d->p0_, d->p1_);
00486         };
00487         // class constructor
00488         Fl_Callback_User_Data_133(const char * p0, int p1)
00489         : p0_(p0),               // copy parameter 0
00490           p1_(p1)                // copy parameter 1
00491         { }                     // constructor body
00492     };
00493     // connect our class to the widget callback
00494     btn->callback(Fl_Callback_User_Data_133::cb,
00495                 new Fl_Callback_User_Data_133("FLTK", 2),
00496                 true);
00497 } while(0);                    // user code adds semicolon
00498 ```
00499
00500 Clicking the Fl_Button `btn` will call
00501 `fl_message("We received the message %s with %d!", "FLTK", 2);`.
00502 Deleting the button will also delete the data that was created in our
00503 boilerplate code.
00504 */
00505
00506 #define FL_INLINE_CALLBACK_5(WIDGET, TYPE0, NAME0, VALUE0, TYPE1, NAME1, VALUE1, TYPE2, NAME2, VALUE2,
00507     TYPE3, NAME3, VALUE3, TYPE4, NAME4, VALUE4, LAMBDA) \
00508 do { \
00509     class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00510     public: \
00511         TYPE0 p0_; TYPE1 p1_; TYPE2 p2_; TYPE3 p3_; TYPE4 p4_; \
00512         static void fn(TYPE0 NAME0, TYPE1 NAME1, TYPE2 NAME2, TYPE3 NAME3, TYPE4 NAME4) \
00513             LAMBDA; \
00514         static void cb(Fl_Widget *w, void *user_data) { \
00515             _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00516             _FL_CBD_CLASS_NAME::fn(d->p0_, d->p1_, d->p2_, d->p3_, d->p4_); \
00517         }; \
00518         _FL_CBD_CLASS_NAME(TYPE0 p0, TYPE1 p1, TYPE2 p2, TYPE3 p3, TYPE4 p4) \
00519         : p0_(p0), p1_(p1), p2_(p2), p3_(p3), p4_(p4) { }; \
00520     }; \
00521     WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0, VALUE1, VALUE2, VALUE3,
00522     VALUE4), true); \
00523 } while(0)
00524
00525 #define FL_INLINE_CALLBACK_4(WIDGET, TYPE0, NAME0, VALUE0, TYPE1, NAME1, VALUE1, TYPE2, NAME2, VALUE2,
00526     TYPE3, NAME3, VALUE3, LAMBDA) \

```

```

00524     do { \
00525         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00526             public: \
00527                 TYPE0 p0_; TYPE1 p1_; TYPE2 p2_; TYPE3 p3_; \
00528                 static void fn(TYPE0 NAME0, TYPE1 NAME1, TYPE2 NAME2, TYPE3 NAME3) \
00529                     LAMBDA; \
00530                 static void cb(Fl_Widget *w, void *user_data) { \
00531                     _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00532                     _FL_CBD_CLASS_NAME::fn(d->p0_, d->p1_, d->p2_, d->p3_); \
00533                 }; \
00534                 _FL_CBD_CLASS_NAME(TYPE0 p0, TYPE1 p1, TYPE2 p2, TYPE3 p3) \
00535                     : p0_(p0), p1_(p1), p2_(p2), p3_(p3) { }; \
00536             }; \
00537             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0, VALUE1, VALUE2, VALUE3),
true); \
00538         } while(0)
00539
00540 #define FL_INLINE_CALLBACK_3(WIDGET, TYPE0, NAME0, VALUE0, TYPE1, NAME1, VALUE1, TYPE2, NAME2, VALUE2,
LAMBDA) \
00541     do { \
00542         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00543             public: \
00544                 TYPE0 p0_; TYPE1 p1_; TYPE2 p2_; \
00545                 static void fn(TYPE0 NAME0, TYPE1 NAME1, TYPE2 NAME2) \
00546                     LAMBDA; \
00547                 static void cb(Fl_Widget *w, void *user_data) { \
00548                     _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00549                     _FL_CBD_CLASS_NAME::fn(d->p0_, d->p1_, d->p2_); \
00550                 }; \
00551                 _FL_CBD_CLASS_NAME(TYPE0 p0, TYPE1 p1, TYPE2 p2) \
00552                     : p0_(p0), p1_(p1), p2_(p2) { }; \
00553             }; \
00554             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0, VALUE1, VALUE2), true); \
00555         } while(0)
00556
00557 #define FL_INLINE_CALLBACK_2(WIDGET, TYPE0, NAME0, VALUE0, TYPE1, NAME1, VALUE1, LAMBDA) \
00558     do { \
00559         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00560             public: \
00561                 TYPE0 p0_; TYPE1 p1_; \
00562                 static void fn(TYPE0 NAME0, TYPE1 NAME1) \
00563                     LAMBDA; \
00564                 static void cb(Fl_Widget *w, void *user_data) { \
00565                     _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00566                     _FL_CBD_CLASS_NAME::fn(d->p0_, d->p1_); \
00567                 }; \
00568                 _FL_CBD_CLASS_NAME(TYPE0 p0, TYPE1 p1) \
00569                     : p0_(p0), p1_(p1) { }; \
00570             }; \
00571             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0, VALUE1), true); \
00572         } while(0)
00573
00574 #define FL_INLINE_CALLBACK_1(WIDGET, TYPE0, NAME0, VALUE0, LAMBDA) \
00575     do { \
00576         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00577             public: \
00578                 TYPE0 p0_; \
00579                 static void fn(TYPE0 NAME0) \
00580                     LAMBDA; \
00581                 static void cb(Fl_Widget *w, void *user_data) { \
00582                     _FL_CBD_CLASS_NAME *d = (_FL_CBD_CLASS_NAME*)user_data; \
00583                     _FL_CBD_CLASS_NAME::fn(d->p0_); \
00584                 }; \
00585                 _FL_CBD_CLASS_NAME(TYPE0 p0) \
00586                     : p0_(p0) { }; \
00587             }; \
00588             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(VALUE0), true); \
00589         } while(0)
00590
00591 #define FL_INLINE_CALLBACK_0(WIDGET, LAMBDA) \
00592     do { \
00593         class _FL_CBD_CLASS_NAME : public Fl_Callback_User_Data { \
00594             public: \
00595                 static void fn() \
00596                     LAMBDA; \
00597                 static void cb(Fl_Widget *w, void *user_data) { \
00598                     _FL_CBD_CLASS_NAME::fn(); \
00599                 }; \
00600                 _FL_CBD_CLASS_NAME() { }; \
00601             }; \
00602             WIDGET->callback(_FL_CBD_CLASS_NAME::cb, new _FL_CBD_CLASS_NAME(), true); \
00603         } while(0)
00604
00605 #endif // FL_DOXYGEN
00606
00607 #endif /* !FL_FL_CALLBACK_MACROS_H_ */

```

12.26 fl_casts.H

```

00001 //
00002 // Experimental inline "cast functions" for the Fast Light Toolkit (FLTK).
00003 // See also issue #109: "VS2017 warnings when building fltk 1.4.x"
00004 //
00005 // Copyright 1998-2021 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00018 #ifndef _FL_fl_casts_H_
00019 #define _FL_fl_casts_H_
00020
00021 #include <FL/platform_types.h>
00022
00023 inline char fl_char(void *v)          { return (char)(fl_intptr_t)v; }
00024 inline int  fl_int(void *v)           { return (int)(fl_intptr_t)v; }
00025 inline long fl_long(void *v)          { return (long)(fl_intptr_t)v; }
00026
00027 inline unsigned char fl_uchar(void *v) { return (unsigned char)(fl_uintptr_t)v; }
00028 inline unsigned int  fl_uint(void *v)  { return (unsigned int)(fl_uintptr_t)v; }
00029 inline unsigned long fl_ulong(void *v)  { return (unsigned long)(fl_uintptr_t)v; }
00030
00031 // the following conversions can be used to silence MSVC warning C4312:
00032 // 'type cast': conversion from '<type>' to 'void *' of greater size
00033
00034 inline void *fl_voidptr(int v)          { return (void *) (fl_intptr_t)v; }
00035 inline void *fl_voidptr(unsigned int v) { return (void *) (fl_uintptr_t)v; }
00036 inline void *fl_voidptr(long v)         { return (void *) (fl_intptr_t)v; }
00037 inline void *fl_voidptr(unsigned long v) { return (void *) (fl_uintptr_t)v; }
00038
00039 #endif /* _FL_fl_casts_H_ */

```

12.27 Fl_Chart.H File Reference

[Fl_Chart](#) widget.

```
#include "Fl_Widget.H"
```

Classes

- class [Fl_Chart](#)
Fl_Chart displays simple charts.
- struct [FL_CHART_ENTRY](#)
For internal use only.

Macros

- #define [FL_BAR_CHART](#) 0
type() for Bar Chart variant
- #define [FL_CHART_LABEL_MAX](#) 18
max label length for entry
- #define [FL_CHART_MAX](#) 128
max entries per chart
- #define [FL_FILL_CHART](#) 3
type() for Fill Line Chart variant
- #define [FL_FILLED_CHART](#) [FL_FILL_CHART](#)
for compatibility
- #define [FL_HORBAR_CHART](#) 1
type() for Horizontal Bar Chart variant

- `#define FL_LINE_CHART 2`
type() for Line Chart variant
- `#define FL_PIE_CHART 5`
type() for Pie Chart variant
- `#define FL_SPECIALPIE_CHART 6`
type() for Special Pie Chart variant
- `#define FL_SPIKE_CHART 4`
type() for Spike Chart variant

12.27.1 Detailed Description

[Fl_Chart](#) widget.

12.28 Fl_Chart.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Fl_Chart widget header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00021 #ifndef Fl_Chart_H
00022 #define Fl_Chart_H
00023
00024 #ifndef Fl_Widget_H
00025 #include "Fl_Widget.H"
00026 #endif
00027
00028 // values for type()
00029 #define FL_BAR_CHART           0
00030 #define FL_HORBAR_CHART       1
00031 #define FL_LINE_CHART         2
00032 #define FL_FILL_CHART         3
00033 #define FL_SPIKE_CHART        4
00034 #define FL_PIE_CHART          5
00035 #define FL_SPECIALPIE_CHART   6
00036
00037 #define FL_FILLED_CHART FL_FILL_CHART
00038
00039 #define FL_CHART_MAX           128
00040 #define FL_CHART_LABEL_MAX     18
00041
00043 struct FL_CHART_ENTRY {
00044     float val;
00045     unsigned col;
00046     char str[FL_CHART_LABEL_MAX + 1];
00047 };
00048
00071 class FL_EXPORT Fl_Chart : public Fl_Widget {
00072     int numb;
00073     int maxnumb;
00074     int sizenumb;
00075     FL_CHART_ENTRY *entries;
00076     double min, max;
00077     uchar autosize_;
00078     Fl_Font textfont_;
00079     Fl_Fonsize textsize_;
00080     Fl_Color textcolor_;
00081
00082 protected:
00083     void draw() FL_OVERRIDE;
00084
00085     // (static) protected draw methods (STR 2022)
00086     // these methods are documented in src/Fl_Chart.cxx
00087
00088     static void draw_barchart(int x, int y, int w, int h, int numb, FL_CHART_ENTRY entries[],

```

```

00089             double min, double max, int autosize, int maxnumb, Fl_Color textcolor);
00090
00091     static void draw_horbarchart(int x, int y, int w, int h, int numb, FL_CHART_ENTRY entries[],
00092             double min, double max, int autosize, int maxnumb,
00093             Fl_Color textcolor);
00094
00095     static void draw_linechart(int type, int x, int y, int w, int h, int numb,
00096             FL_CHART_ENTRY entries[], double min, double max, int autosize,
00097             int maxnumb, Fl_Color textcolor);
00098
00099     static void draw_piechart(int x, int y, int w, int h, int numb, FL_CHART_ENTRY entries[],
00100             int special, Fl_Color textcolor);
00101
00102 public:
00103     Fl_Chart(int X, int Y, int W, int H, const char *L = 0);
00104
00105     ~Fl_Chart();
00106
00107     void clear();
00108
00109     void add(double val, const char *str = 0, unsigned col = 0);
00110
00111     void insert(int ind, double val, const char *str = 0, unsigned col = 0);
00112
00113     void replace(int ind, double val, const char *str = 0, unsigned col = 0);
00114
00115     void bounds(double *a, double *b) const {
00116         *a = min;
00117         *b = max;
00118     }
00119
00120     void bounds(double a, double b);
00121
00122     int size() const { return numb; }
00123
00124     void size(int W, int H) { Fl_Widget::size(W, H); }
00125
00126     int maxsize() const { return maxnumb; }
00127
00128     void maxsize(int m);
00129
00130     Fl_Font textfont() const { return textfont_; }
00131
00132     void textfont(Fl_Font s) { textfont_ = s; }
00133
00134     Fl_Fontsize textsize() const { return textsize_; }
00135
00136     void textsize(Fl_Fontsize s) { textsize_ = s; }
00137
00138     Fl_Color textcolor() const { return textcolor_; }
00139
00140     void textcolor(Fl_Color n) { textcolor_ = n; }
00141
00142     uchar autosize() const { return autosize_; }
00143
00144     void autosize(uchar n) { autosize_ = n; }
00145 };
00146 #endif

```

12.29 Fl_Check_Browser.H

```

00001 //
00002 // Fl_Check_Browser header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2020 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Check_Browser widget . */
00019
00020 #ifndef Fl_Check_Browser_H
00021 #define Fl_Check_Browser_H
00022
00023 #include "Fl.H"

```

```

00024 #include "Fl_Browser_.H"
00025
00030 class FL_EXPORT Fl_Check_Browser : public Fl_Browser_ {
00031
00032 protected:
00033     /* required routines for Fl_Browser_ subclass: */
00034     void *item_first() const FL_OVERRIDE;
00035     void *item_next(void *) const FL_OVERRIDE;
00036     void *item_prev(void *) const FL_OVERRIDE;
00037     int item_height(void *) const FL_OVERRIDE;
00038     int item_width(void *) const FL_OVERRIDE;
00039     void item_draw(void *, int, int, int, int) const FL_OVERRIDE;
00040     void item_select(void *, int) FL_OVERRIDE;
00041     int item_selected(void *) const FL_OVERRIDE;
00042     const char *item_text(void *) const FL_OVERRIDE;
00043
00044 public:
00045     void *item_at(int index) const FL_OVERRIDE;
00046     void item_swap(int ia, int ib);
00047     void item_swap(void *a, void *b) FL_OVERRIDE;
00048
00049     /* private data */
00050
00051 public: // IRIX 5.3 C++ compiler doesn't support private structures...
00052
00053 #ifndef FL_DOXYGEN
00054     struct cb_item {
00055         cb_item *next;
00056         cb_item *prev;
00057         char checked;
00058         char selected;
00059         char *text;
00060     };
00061 #endif // !FL_DOXYGEN
00062
00063 private:
00064     cb_item *first;
00065     cb_item *last;
00066     cb_item *cache;
00067     int cached_item;
00068     int nitems_;
00069     int nchecked_;
00070     cb_item *find_item(int) const;
00071     int lineno(cb_item *) const;
00072
00073 public:
00074     Fl_Check_Browser(int x, int y, int w, int h, const char *l = 0);
00075     ~Fl_Check_Browser() { clear(); }
00076     int add(char *s); // add an (unchecked) item
00077     int add(char *s, int b); // add an item and set checked
00078     // both return the new nitems()
00079     int remove(int item); // delete an item. Returns nitems()
00080
00081     // inline const char * methods to avoid breaking binary compatibility...
00082     int add(const char *s) { return add((char *)s); }
00083     int add(const char *s, int b) { return add((char *)s, b); }
00084
00085     void clear(); // delete all items
00086     int nitems() const { return nitems_; }
00087     int nchecked() const { return nchecked_; }
00088     int checked(int item) const;
00089     void checked(int item, int b);
00090     void set_checked(int item) { checked(item, 1); }
00091     void check_all();
00092     void check_none();
00093     int value() const; // currently selected item
00094     char *text(int item) const; // returns pointer to internal buffer
00095
00096 protected:
00097     int handle(int) FL_OVERRIDE;
00098 };
00099
00100 #endif // Fl_Check_Button_H

```

12.30 Fl_Check_Button.H

```

00001 //
00002 // Check button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2014 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //

```

```

00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Check_Button_H
00018 #define Fl_Check_Button_H
00019
00020 #include "Fl_Light_Button.H"
00021
00022 /*
00023     class: Fl_Check_Button.
00024
00025     A button with a "checkmark" to show its status.
00026 */
00027
00028 class FL_EXPORT Fl_Check_Button : public Fl_Light_Button {
00029 public:
00030     Fl_Check_Button(int X, int Y, int W, int H, const char *L = 0);
00031 };
00032
00033 #endif

```

12.31 Fl_Choice.H

```

00001 //
00002 // Choice header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Choice widget . */
00019
00020 #ifndef Fl_Choice_H
00021 #define Fl_Choice_H
00022
00023 #include "Fl_Menu_.H"
00024
00025 class FL_EXPORT Fl_Choice : public Fl_Menu_ {
00026 protected:
00027     void draw() FL_OVERRIDE;
00028 public:
00029     int handle(int) FL_OVERRIDE;
00030
00031     Fl_Choice(int X, int Y, int W, int H, const char *L = 0);
00032
00033     int value() const {return Fl_Menu_::value();}
00034
00035     int value(int v);
00036
00037     int value(const Fl_Menu_Item* v);
00038 };
00039
00040 #endif

```

12.32 Fl_Clock.H

```

00001 //
00002 // Clock header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2017 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:

```

```

00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Clock, Fl_Clock_Output widgets . */
00019
00020 #ifndef Fl_Clock_H
00021 #define Fl_Clock_H
00022
00023 #ifndef Fl_Widget_H
00024 #include "Fl_Widget.H"
00025 #endif
00026
00027 // Values for type():
00028 // Please change doxygen documentation below (class Fl_Clock_Output)
00029 // accordingly as well when changing the following type values:
00030
00031 #define FL_SQUARE_CLOCK      0
00032 #define FL_ROUND_CLOCK      1
00033 #define FL_ANALOG_CLOCK FL_SQUARE_CLOCK
00034 #define FL_DIGITAL_CLOCK FL_SQUARE_CLOCK
00036 // fabien: Please keep the horizontal formatting of both images in class desc,
00037 // don't lose vertical space for nothing!
00038
00065 class FL_EXPORT Fl_Clock_Output : public Fl_Widget {
00066     int hour_, minute_, second_;
00067     ulong value_;
00068     int shadow_; // draw shadows of hands
00069     void drawhands(Fl_Color, Fl_Color); // part of draw
00070 protected:
00071     void draw() FL_OVERRIDE;
00072     void draw(int X, int Y, int W, int H);
00073 public:
00074
00075     Fl_Clock_Output(int X, int Y, int W, int H, const char *L = 0);
00076
00077     void value(ulong v); // set to this Unix time
00078
00079     void value(int H, int m, int s);
00080
00086     ulong value() const {return value_;}
00087
00092     int hour() const {return hour_;}
00093
00098     int minute() const {return minute_;}
00099
00104     int second() const {return second_;}
00105
00113     int shadow() const {return shadow_;}
00114
00127     void shadow(int mode) { shadow_ = mode ? 1 : 0; }
00128 };
00129
00130 // a Fl_Clock displays the current time always by using a timeout:
00131
00152 class FL_EXPORT Fl_Clock : public Fl_Clock_Output {
00153 public:
00154     int handle(int) FL_OVERRIDE;
00155
00156     Fl_Clock(int X, int Y, int W, int H, const char *L = 0);
00157
00158     Fl_Clock(uchar t, int X, int Y, int W, int H, const char *L);
00159
00160     ~Fl_Clock();
00161 };
00162
00163 #endif

```

12.33 Fl_Color_Chooser.H File Reference

[Fl_Color_Chooser](#) widget .

```

#include <FL/Fl_Group.H>
#include <FL/Fl_Box.H>
#include <FL/Fl_Return_Button.H>
#include <FL/Fl_Choice.H>
#include <FL/Fl_Value_Input.H>

```

Classes

- class [Fl_Color_Chooser](#)

The [Fl_Color_Chooser](#) widget provides a standard RGB color chooser.

12.33.1 Detailed Description

[Fl_Color_Chooser](#) widget .

12.34 Fl_Color_Chooser.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Color chooser header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2019 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00020 // The color chooser object and the color chooser popup. The popup
00021 // is just a window containing a single color chooser and some boxes
00022 // to indicate the current and cancelled color.
00023
00024 #ifndef Fl_Color_Chooser_H
00025 #define Fl_Color_Chooser_H
00026
00027 #include <FL/Fl_Group.H>
00028 #include <FL/Fl_Box.H>
00029 #include <FL/Fl_Return_Button.H>
00030 #include <FL/Fl_Choice.H>
00031 #include <FL/Fl_Value_Input.H>
00032
00033 #ifndef FL_DOXYGEN
00034
00036 class FL_EXPORT Flcc_HueBox : public Fl_Widget {
00037     int px, py;
00038 protected:
00039     void draw() FL_OVERRIDE;
00040     int handle_key(int);
00041 public:
00042     int handle(int) FL_OVERRIDE;
00043     Flcc_HueBox(int X, int Y, int W, int H) : Fl_Widget(X,Y,W,H) {
00044         px = py = 0;
00045     };
00046
00048 class FL_EXPORT Flcc_ValueBox : public Fl_Widget {
00049     int py;
00050 protected:
00051     void draw() FL_OVERRIDE;
00052     int handle_key(int);
00053 public:
00054     int handle(int) FL_OVERRIDE;
00055     Flcc_ValueBox(int X, int Y, int W, int H) : Fl_Widget(X,Y,W,H) {
00056         py = 0;
00057     };
00058
00060 class FL_EXPORT Flcc_Value_Input : public Fl_Value_Input {
00061 public:
00062     int format(char*) FL_OVERRIDE;
00063     Flcc_Value_Input(int X, int Y, int W, int H) : Fl_Value_Input(X,Y,W,H) {}
00064 };
00065
00066 #endif // !FL_DOXYGEN
00067
00112 class FL_EXPORT Fl_Color_Chooser : public Fl_Group {
00113     Flcc_HueBox huebox;
00114     Flcc_ValueBox valuebox;
00115     Fl_Choice choice;
00116     Flcc_Value_Input rvalue;
00117     Flcc_Value_Input gvalue;
00118     Flcc_Value_Input bvalue;
00119     Fl_Box resize_box;
```

```

00120     double hue_, saturation_, value_;
00121     double r_, g_, b_;
00122     void set_valuators();
00123     static void rgb_cb(Fl_Widget*, void*);
00124     static void mode_cb(Fl_Widget*, void*);
00125 public:
00126
00127     int handle(int e) FL_OVERRIDE;
00128
00133     int mode() {return choice.value();}
00134
00139     void mode(int newMode);
00140
00147     double hue() const {return hue_;}
00148
00153     double saturation() const {return saturation_;}
00154
00159     double value() const {return value_;}
00160
00165     double r() const {return r_;}
00166
00171     double g() const {return g_;}
00172
00177     double b() const {return b_;}
00178
00179     int hsv(double H, double S, double V);
00180
00181     int rgb(double R, double G, double B);
00182
00183     static void hsv2rgb(double H, double S, double V, double& R, double& G, double& B);
00184
00185     static void rgb2hsv(double R, double G, double B, double& H, double& S, double& V);
00186
00187     Fl_Color_Chooser(int X, int Y, int W, int H, const char *L = 0);
00188 };
00189
00190 FL_EXPORT int fl_color_chooser(const char* name, double& r, double& g, double& b, int m=-1);
00191 FL_EXPORT int fl_color_chooser(const char* name, uchar& r, uchar& g, uchar& b, int m=-1);
00192
00193 #endif

```

12.35 Fl_Copy_Surface.H

```

00001 //
00002 // Copy-to-clipboard code for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Copy_Surface_H
00018 #define Fl_Copy_Surface_H
00019
00020 #include <FL/Fl_Widget_Surface.H>
00021
00049 class FL_EXPORT Fl_Copy_Surface : public Fl_Widget_Surface {
00050 private:
00051     class Fl_Copy_Surface_Driver *platform_surface;
00052 protected:
00053     void translate(int x, int y) FL_OVERRIDE;
00054     void untranslate() FL_OVERRIDE;
00055 public:
00056     Fl_Copy_Surface(int w, int h);
00057     ~Fl_Copy_Surface();
00058     void set_current() FL_OVERRIDE;
00059     bool is_current() FL_OVERRIDE;
00061     int w();
00063     int h();
00064     void origin(int *x, int *y) FL_OVERRIDE;
00065     void origin(int x, int y) FL_OVERRIDE;
00066     int printable_rect(int *w, int *h) FL_OVERRIDE;
00067 };
00068
00069
00084 class Fl_Copy_Surface_Driver : public Fl_Widget_Surface {
00085     friend class Fl_Copy_Surface;

```

```

00086 protected:
00087     int width;
00088     int height;
00089     Fl_Copy_Surface_Driver(int w, int h) : Fl_Widget_Surface(NULL), width(w), height(h) {}
00090     virtual ~Fl_Copy_Surface_Driver() {}
00091     void set_current() FL_OVERRIDE = 0;
00092     void translate(int x, int y) FL_OVERRIDE = 0;
00093     void untranslate() FL_OVERRIDE = 0;
00094     int printable_rect(int *w, int *h) FL_OVERRIDE;
00099     static Fl_Copy_Surface_Driver *newCopySurfaceDriver(int w, int h);
00100 };
00101
00107 #endif // Fl_Copy_Surface_H

```

12.36 Fl_Counter.H

```

00001 //
00002 // Counter header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Counter widget . */
00019
00020 // A numerical value with up/down step buttons. From Forms.
00021
00022 #ifndef Fl_Counter_H
00023 #define Fl_Counter_H
00024
00025 #ifndef Fl_Valuator_H
00026 #include "Fl_Valuator.H"
00027 #endif
00028
00029 // values for type():
00030 #define FL_NORMAL_COUNTER      0
00031 #define FL_SIMPLE_COUNTER      1
00044 class FL_EXPORT Fl_Counter : public Fl_Valuator {
00045
00046     Fl_Font textfont_;
00047     Fl_Fontsize textsize_;
00048     Fl_Color textcolor_;
00049     double lstep_;
00050     uchar mouseobj_;
00051     static void repeat_callback(void *);
00052     int calc_mouseobj();
00053     void increment_cb();
00054
00055 protected:
00056
00057     void draw() FL_OVERRIDE;
00058     // compute widths of arrow boxes
00059     void arrow_widths(int &w1, int &w2);
00060
00061 public:
00062
00063     int handle(int) FL_OVERRIDE;
00064
00065     Fl_Counter(int X, int Y, int W, int H, const char* L = 0);
00066     ~Fl_Counter();
00067
00073     void lstep(double a) {lstep_ = a;}
00074
00079     void step(double a, double b) {Fl_Valuator::step(a); lstep_ = b;}
00080
00085     void step(double a) {Fl_Valuator::step(a);}
00086
00090     double step() const {return Fl_Valuator::step();}
00091
00093     Fl_Font textfont() const {return textfont_;}
00095     void textfont(Fl_Font s) {textfont_ = s;}
00096
00098     Fl_Fontsize textsize() const {return textsize_;}
00100     void textsize(Fl_Fontsize s) {textsize_ = s;}
00101

```



```

00103     Fl_Color textcolor() const {return textcolor_;}
00105     void textcolor(Fl_Color s) {textcolor_ = s;}
00106
00107 };
00108
00109 #endif

```

12.37 Fl_Device.H File Reference

declaration of classes [Fl_Surface_Device](#), [Fl_Display_Device](#), [Fl_Device_Plugin](#).

```

#include <FL/Fl_Plugin.H>
#include <FL/platform_types.h>

```

Classes

- class [Fl_Device_Plugin](#)
This plugin socket allows the integration of new device drivers for special window or screen types.
- class [Fl_Display_Device](#)
The computer's display.
- class [Fl_Surface_Device](#)
A drawing surface that's susceptible to receive graphical output.

12.37.1 Detailed Description

declaration of classes [Fl_Surface_Device](#), [Fl_Display_Device](#), [Fl_Device_Plugin](#).

12.38 Fl_Device.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Definition of classes Fl_Surface_Device, Fl_Display_Device
00003 // for the Fast Light Tool Kit (FLTK).
00004 //
00005 // Copyright 2010-2021 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00022 #ifndef Fl_Device_H
00023 #define Fl_Device_H
00024
00025 #include <FL/Fl_Plugin.H>
00026 #include <FL/platform_types.h>
00027
00028 class Fl_Graphics_Driver;
00029 class Fl_RGB_Image;
00030 class Fl_Widget;
00031 class Fl_Image_Surface;
00032
00057 class FL_EXPORT Fl_Surface_Device {
00059     Fl_Graphics_Driver *pGraphicsDriver;
00060     static Fl_Surface_Device *surface_; // the surface that currently receives graphics requests
00061     static Fl_Surface_Device *default_surface(); // create surface if none exists yet
00062 protected:
00067     virtual void end_current() { surface_ = 0; }
00069     Fl_Surface_Device(Fl_Graphics_Driver *graphics_driver) {pGraphicsDriver = graphics_driver; }
00071     inline void driver(Fl_Graphics_Driver *graphics_driver) {pGraphicsDriver = graphics_driver;}
00072 public:
00073     virtual void set_current(void);
00074     virtual bool is_current();
00076     inline Fl_Graphics_Driver *driver() {return pGraphicsDriver; }
00081     static inline Fl_Surface_Device *surface() {
00082         return surface_ ? surface_ : default_surface();

```

```

00083     }
00084     virtual ~Fl_Surface_Device();
00085     static void push_current(Fl_Surface_Device *new_current);
00086     static Fl_Surface_Device *pop_current();
00087 };
00088
00089
00090 class FL_EXPORT Fl_Display_Device : public Fl_Surface_Device {
00091     Fl_Display_Device(Fl_Graphics_Driver *graphics_driver);
00092 public:
00093     static Fl_Display_Device *display_device();
00094 };
00095
00096
00097 class Fl_Device_Plugin : public Fl_Plugin {
00098 public:
00099     Fl_Device_Plugin(const char *pluginName)
00100         : Fl_Plugin(klass(), pluginName) { }
00101     virtual const char *klass() { return "fltk:device"; }
00102     virtual const char *name() = 0;
00103     virtual int print(Fl_Widget* w) = 0;
00104     virtual Fl_RGB_Image* rectangle_capture(Fl_Widget *widget, int x, int y, int w, int h) = 0;
00105     static Fl_Device_Plugin *opengl_plugin();
00106 };
00107 #endif // Fl_Device_H

```

12.39 Fl_Dial.H

```

00001 //
00002 // Dial header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Dial widget . */
00019
00020 #ifndef Fl_Dial_H
00021 #define Fl_Dial_H
00022
00023 #ifndef Fl_Valuator_H
00024 #include "Fl_Valuator.H"
00025 #endif
00026
00027 // values for type():
00028 #define FL_NORMAL_DIAL 0
00029 #define FL_LINE_DIAL 1
00030 #define FL_FILL_DIAL 2
00031
00032 class FL_EXPORT Fl_Dial : public Fl_Valuator {
00033 public:
00034     short a1,a2;
00035 protected:
00036     // these allow subclasses to put the dial in a smaller area:
00037     void draw(int X, int Y, int W, int H);
00038     int handle(int event, int X, int Y, int W, int H);
00039     void draw() FL_OVERRIDE;
00040 public:
00041     int handle(int) FL_OVERRIDE;
00042     Fl_Dial(int x,int y,int w,int h, const char *l = 0);
00043     short angle1() const {return a1;}
00044     void angle1(short a) {a1 = a;}
00045     short angle2() const {return a2;}
00046     void angle2(short a) {a2 = a;}
00047     void angles(short a, short b) {a1 = a; a2 = b;}
00048 };
00049 #endif

```

12.40 Fl_Double_Window.H

```

00001 //
00002 // Double-buffered window header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Double_Window widget . */
00019
00020 #ifndef Fl_Double_Window_H
00021 #define Fl_Double_Window_H
00022
00023 #include "Fl_Window.H"
00024
00030 class FL_EXPORT Fl_Double_Window : public Fl_Window
00031 {
00032 public:
00033     Fl_Double_Window *as_double_window() FL_OVERRIDE {return this; }
00034     void show() FL_OVERRIDE;
00036     void show(int a, char **b) {Fl_Window::show(a,b);}
00037     void resize(int,int,int,int) FL_OVERRIDE;
00038     void hide() FL_OVERRIDE;
00039     void flush() FL_OVERRIDE;
00040     ~Fl_Double_Window();
00041
00046     Fl_Double_Window(int W, int H, const char *l = 0);
00047
00051     Fl_Double_Window(int X, int Y, int W, int H, const char *l = 0);
00052
00053 };
00054
00055 #endif

```

12.41 fl_draw.H File Reference

utility header to pull drawing functions together

```

#include <FL/Enumerations.H>
#include <FL/Fl_Graphics_Driver.H>
#include <FL/Fl_Rect.H>

```

Enumerations

- enum {
FL_SOLID = 0 , FL_DASH = 1 , FL_DOT = 2 , FL_DASHDOT = 3 ,
FL_DASHDOTDOT = 4 , FL_CAP_FLAT = 0x100 , FL_CAP_ROUND = 0x200 , FL_CAP_SQUARE = 0x300 ,
FL_JOIN_MITER = 0x1000 , FL_JOIN_ROUND = 0x2000 , FL_JOIN_BEVEL = 0x3000 }

Functions

- int fl_add_symbol (const char *name, void(*drawit)(Fl_Color), int scalable)
Adds a symbol to the system.
- int fl_antialias ()
Return whether line drawings are currently antialiased.
- void fl_antialias (int state)
Turn antialiased line drawings ON or OFF, if supported by platform.
- void fl_arc (double x, double y, double r, double start, double end)
Add a series of points to the current path on the arc of a circle.

- void **fl_arc** (int x, int y, int w, int h, double a1, double a2)
Draw ellipse sections using integer coordinates.
- void **fl_begin_complex_polygon** ()
Start drawing a complex filled polygon.
- void **fl_begin_line** ()
Start drawing a list of lines.
- void **fl_begin_loop** ()
Start drawing a closed sequence of lines.
- void **fl_begin_offscreen** (FI_Offscreen ctx)
Send all subsequent drawing commands to this offscreen buffer.
- void **fl_begin_points** ()
Start drawing a list of points.
- void **fl_begin_polygon** ()
Start drawing a convex filled polygon.
- char **fl_can_do_alpha_blending** ()
Check whether platform supports true alpha blending for RGBA images.
- FI_RGB_Image * **fl_capture_window** (FI_Window *win, int x, int y, int w, int h)
Captures the content of a rectangular zone of a mapped window.
- void **fl_chord** (int x, int y, int w, int h, double a1, double a2)
fl_chord declaration is a place holder - the function does not yet exist
- void **fl_circle** (double x, double y, double r)
fl_circle(x,y,r) is equivalent to fl_arc(x,y,r,0,360), but may be faster.
- void **fl_clip** (int x, int y, int w, int h)
Intersect the current clip region with a rectangle and push this new region onto the stack (deprecated).
- int **fl_clip_box** (int x, int y, int w, int h, int &X, int &Y, int &W, int &H)
Intersect a rectangle with the current clip region and return the bounding box of the result.
- FI_Region **fl_clip_region** ()
Return the current clipping region.
- void **fl_clip_region** (FI_Region r)
Replace the top of the clipping stack with a clipping region of any shape.
- FI_Color **fl_color** ()
Return the last fl_color() that was set.
- void **fl_color** (FI_Color c)
Set the color for all subsequent drawing operations.
- void **fl_color** (int c)
for back compatibility - use fl_color(FI_Color c) instead
- void **fl_color** (uchar r, uchar g, uchar b)
Set the color for all subsequent drawing operations.
- void **fl_copy_offscreen** (int x, int y, int w, int h, FI_Offscreen pixmap, int srcx, int srcy)
Copy a rectangular area of the given offscreen buffer into the current drawing destination.
- FI_Offscreen **fl_create_offscreen** (int w, int h)
Creation of an offscreen graphics buffer.
- void **fl_cursor** (FI_Cursor)
Sets the cursor for the current window to the specified shape and colors.
- void **fl_cursor** (FI_Cursor, FI_Color fg, FI_Color bg=FL_WHITE)
- void **fl_curve** (double X0, double Y0, double X1, double Y1, double X2, double Y2, double X3, double Y3)
Add a series of points on a Bézier curve to the path.
- void **fl_delete_offscreen** (FI_Offscreen ctx)
Deletion of an offscreen graphics buffer.
- int **fl_descent** ()

Return the recommended distance above the bottom of a `fl_height()` tall box to draw the text at so it looks centered vertically in that box.

- void **fl_draw** (const char *str, int n, int x, int y)

Draws starting at the given *x*, *y* location a UTF-8 string of length *n* bytes.

- void **fl_draw** (const char *str, int x, int y)

Draw a nul-terminated UTF-8 string starting at the given *x*, *y* location.

- void **fl_draw** (const char *str, int x, int y, int w, int h, **FL_Align** align, **FL_Image** *img=0, int draw_symbols=1, int spacing=0)

Fancy string drawing function which is used to draw all the labels.

- void **fl_draw** (const char *str, int x, int y, int w, int h, **FL_Align** align, void(*callthis)(const char *, int, int, int), **FL_Image** *img=0, int draw_symbols=1, int spacing=0)

The same as `fl_draw(const char*,int,int,int,int,FL_Align,FL_Image*,int)` with the addition of the *callthis* parameter, which is a pointer to a text drawing function such as `fl_draw(const char*, int, int, int)` to do the real work.

- void **fl_draw** (int angle, const char *str, int n, int x, int y)

Draw at the given *x*, *y* location a UTF-8 string of length *n* bytes rotating *angle* degrees counter-clockwise.

- void **fl_draw** (int angle, const char *str, int x, int y)

Draw a nul-terminated UTF-8 string starting at the given *x*, *y* location and rotating *angle* degrees counter-clockwise.

- void **fl_draw_arrow** (**FL_Rect** bb, **FL_Arrow_Type** t, **FL_Orientation** o, **FL_Color** color)

Draw an "arrow like" GUI element for the selected scheme.

- void **fl_draw_box** (**FL_Boxtype**, int x, int y, int w, int h, **FL_Color**)

Draws a box using given type, position, size and color.

- void **fl_draw_box_focus** (**FL_Boxtype**, int x, int y, int w, int h, **FL_Color**, **FL_Color**)

Draws the focus rectangle inside a box using given type, position, size and color.

- void **fl_draw_check** (**FL_Rect** bb, **FL_Color** col)

Draw a check mark inside the given bounding box.

- void **fl_draw_circle** (int x, int y, int d, **FL_Color** color)

Draw a potentially small, filled circle using a given color.

- void **fl_draw_image** (const uchar *buf, int X, int Y, int W, int H, int D=3, int L=0)

Draw an 8-bit per color RGB or luminance image.

- void **fl_draw_image** (**FL_Draw_Image_Cb** cb, void *data, int X, int Y, int W, int H, int D=3)

Draw an image using a callback function to generate image data.

- void **fl_draw_image_mono** (const uchar *buf, int X, int Y, int W, int H, int D=1, int L=0)

Draw a gray-scale (1 channel) image.

- void **fl_draw_image_mono** (**FL_Draw_Image_Cb** cb, void *data, int X, int Y, int W, int H, int D=1)

Draw a gray-scale image using a callback function to generate image data.

- int **fl_draw_pixmap** (char *const *data, int x, int y, **FL_Color** bg=FL_GRAY)

Draw XPM image data, with the top-left corner at the given position.

- int **fl_draw_pixmap** (const char *const *data, int x, int y, **FL_Color** bg=FL_GRAY)

Draw XPM image data, with the top-left corner at the given position.

- void **fl_draw_radio** (int x, int y, int d, **FL_Color** color)

Draw a round check mark (circle) of a radio button.

- int **fl_draw_symbol** (const char *label, int x, int y, int w, int h, **FL_Color**)

Draw the named symbol in the given rectangle using the given color.

- void **fl_end_complex_polygon** ()

End complex filled polygon, and draw.

- void **fl_end_line** ()

End list of lines, and draw.

- void **fl_end_loop** ()

End closed sequence of lines, and draw.

- void **fl_end_offscreen** ()

Quit sending drawing commands to the current offscreen buffer.

- void **fl_end_points** ()
End list of points, and draw.
- void **fl_end_polygon** ()
End convex filled polygon, and draw.
- const char * **fl_expand_text** (const char *from, char *buf, int maxbuf, double maxw, int &n, double &width, int wrap, int draw_symbols=0)
Copy from to buf, replacing control characters with ^X.
- void **fl_focus_rect** (int x, int y, int w, int h)
Draw a dotted rectangle, used to indicate keyboard focus on a widget.
- **Fl_Font** **fl_font** ()
Return the face set by the most recent call to fl_font().
- void **fl_font** (**Fl_Font** face, **Fl_Fontsize** fsize)
Sets the current font, which is then used in various drawing routines.
- void **fl_frame** (const char *s, int x, int y, int w, int h)
Draws a series of line segments around the given box.
- void **fl_frame2** (const char *s, int x, int y, int w, int h)
Draws a series of line segments around the given box.
- void **fl_gap** ()
Separate loops of the path.
- int **fl_height** ()
Return the recommended minimum line spacing for the current font.
- int **fl_height** (int font, int size)
This function returns the actual height of the specified font and size.
- const char * **fl_latin1_to_local** (const char *t, int n=-1)
Convert text from Windows/X11 latin1 character set to local encoding.
- void **fl_line** (int x, int y, int x1, int y1)
Draw a line from (x,y) to (x1,y1)
- void **fl_line** (int x, int y, int x1, int y1, int x2, int y2)
Draw a line from (x,y) to (x1,y1) and another from (x1,y1) to (x2,y2)
- void **fl_line_style** (int style, int width=0, char *dashes=0)
Set how to draw lines (the "pen").
- void **fl_load_identity** ()
Set the transformation matrix to identity.
- void **fl_load_matrix** (double a, double b, double c, double d, double x, double y)
Set the current transformation matrix.
- const char * **fl_local_to_latin1** (const char *t, int n=-1)
Convert text from local encoding to Windows/X11 latin1 character set.
- const char * **fl_local_to_mac_roman** (const char *t, int n=-1)
Convert text from local encoding to Mac Roman character set.
- void **fl_loop** (int x, int y, int x1, int y1, int x2, int y2)
Outline a 3-sided polygon with lines.
- void **fl_loop** (int x, int y, int x1, int y1, int x2, int y2, int x3, int y3)
Outline a 4-sided polygon with lines.
- const char * **fl_mac_roman_to_local** (const char *t, int n=-1)
Convert text from Mac Roman character set to local encoding.
- void **fl_measure** (const char *str, int &x, int &y, int draw_symbols=1)
Measure how wide and tall the string will be when printed by the fl_draw() function with align parameter.
- int **fl_measure_pixmap** (char *const *data, int &w, int &h)
Get the dimensions of a pixmap.
- int **fl_measure_pixmap** (const char *const *cdata, int &w, int &h)
Get the dimensions of a pixmap.

- void [fl_mult_matrix](#) (double a, double b, double c, double d, double x, double y)
Concatenate another transformation onto the current one.
- int [fl_not_clipped](#) (int x, int y, int w, int h)
Does the rectangle intersect the current clip region?
- unsigned int [fl_old_shortcut](#) (const char *s)
Emulation of XForms named shortcuts.
- void [fl_overlay_clear](#) ()
Erase a selection rectangle without drawing a new one.
- void [fl_overlay_rect](#) (int x, int y, int w, int h)
Draw a transient dotted selection rectangle.
- float [fl_override_scale](#) ()
Removes any GUI scaling factor in subsequent drawing operations.
- void [fl_pie](#) (int x, int y, int w, int h, double a1, double a2)
Draw filled ellipse sections using integer coordinates.
- void [fl_point](#) (int x, int y)
Draw a single pixel at the given coordinates.
- void [fl_polygon](#) (int x, int y, int x1, int y1, int x2, int y2)
Fill a 3-sided polygon.
- void [fl_polygon](#) (int x, int y, int x1, int y1, int x2, int y2, int x3, int y3)
Fill a 4-sided polygon.
- void [fl_pop_clip](#) ()
Restore the previous clip region.
- void [fl_pop_matrix](#) ()
Restore the current transformation matrix from the stack.
- void [fl_push_clip](#) (int x, int y, int w, int h)
Intersect the current clip region with a rectangle and push this new region onto the stack.
- void [fl_push_matrix](#) ()
Save the current transformation matrix on the stack.
- void [fl_push_no_clip](#) ()
Push an empty clip region onto the stack so nothing will be clipped.
- [uchar](#) * [fl_read_image](#) ([uchar](#) *p, int X, int Y, int W, int H, int alpha=0)
Reads an RGB(A) image from the current window or off-screen buffer.
- void [fl_rect](#) ([Fl_Rect](#) r)
Draw a border inside the given bounding box.
- void [fl_rect](#) (int x, int y, int w, int h)
Draw a border inside the given bounding box.
- void [fl_rect](#) (int x, int y, int w, int h, [Fl_Color](#) c)
Draw with passed color a border inside the given bounding box.
- void [fl_rectf](#) ([Fl_Rect](#) bb, [uchar](#) r, [uchar](#) g, [uchar](#) b)
*Color a rectangle with "exactly" the passed *r*, *g*, *b* color.*
- void [fl_rectf](#) ([Fl_Rect](#) r)
Color with current color a rectangle that exactly fills the given bounding box.
- void [fl_rectf](#) ([Fl_Rect](#) r, [Fl_Color](#) c)
Color with passed color a rectangle that exactly fills the given bounding box.
- void [fl_rectf](#) (int x, int y, int w, int h)
Color with current color a rectangle that exactly fills the given bounding box.
- void [fl_rectf](#) (int x, int y, int w, int h, [Fl_Color](#) c)
Color with passed color a rectangle that exactly fills the given bounding box.
- void [fl_rectf](#) (int x, int y, int w, int h, [uchar](#) r, [uchar](#) g, [uchar](#) b)
*Color a rectangle with "exactly" the passed *r*, *g*, *b* color.*
- void [fl_rescale_offscreen](#) ([Fl_Offscreen](#) &ctx)

- Adapts an offscreen buffer to a changed value of the scale factor.*

 - void [fl_reset_spot](#) (void)

Resets marked text.
 - void [fl_restore_clip](#) ()

Undo any clobbering of the clip region done by your program.
 - void [fl_restore_scale](#) (float s)

Restores the GUI scaling factor in subsequent drawing operations.
 - void [fl_rotate](#) (double d)

Concatenate rotation transformation onto the current one.
 - void [fl_rounded_rect](#) (int x, int y, int w, int h, int r)

Draw a rounded border inside the given bounding box.
 - void [fl_rounded_rectf](#) (int x, int y, int w, int h, int r)

Color with current color a rounded rectangle that exactly fills the given bounding box.
 - void [fl_rtl_draw](#) (const char *str, int n, int x, int y)

*Draw a UTF-8 string of length *n* bytes right to left starting at the given *x*, *y* location.*
 - void [fl_scale](#) (double x)

Concatenate scaling transformation onto the current one.
 - void [fl_scale](#) (double x, double y)

Concatenate scaling transformation onto the current one.
 - void [fl_scroll](#) (int X, int Y, int W, int H, int dx, int dy, void(*draw_area)(void *, int, int, int, int), void *data)

Scroll a rectangle and draw the newly exposed portions.
 - void [fl_set_spot](#) (int font, int size, int X, int Y, int W, int H, [Fl_Window](#) *win=0)

Inform text input methods about the current text insertion cursor.
 - void [fl_set_status](#) (int X, int Y, int W, int H)

Related to text input methods under X11.
 - const char * [fl_shortcut_label](#) (unsigned int shortcut)

Get a human-readable string from a shortcut value.
 - const char * [fl_shortcut_label](#) (unsigned int shortcut, const char **eom)

Get a human-readable string from a shortcut value.
 - [Fl_Fontsize](#) [fl_size](#) ()

*Return the *size* set by the most recent call to [fl_font\(\)](#).*
 - void [fl_text_extents](#) (const char *, int &dx, int &dy, int &w, int &h)

Determine the minimum pixel dimensions of a nul-terminated string using the current [fl_font\(\)](#).
 - void [fl_text_extents](#) (const char *t, int n, int &dx, int &dy, int &w, int &h)

*Determine the minimum pixel dimensions of a sequence of *n* characters (bytes) using the current [fl_font\(\)](#).*
 - double [fl_transform_dx](#) (double x, double y)

Transform distance using current transformation matrix.
 - double [fl_transform_dy](#) (double x, double y)

Transform distance using current transformation matrix.
 - double [fl_transform_x](#) (double x, double y)

Transform coordinate using the current transformation matrix.
 - double [fl_transform_y](#) (double x, double y)

Transform coordinate using the current transformation matrix.
 - void [fl_transformed_vertex](#) (double xf, double yf)

Add coordinate pair to the vertex list without further transformations.
 - void [fl_translate](#) (double x, double y)

Concatenate translation transformation onto the current one.
 - void [fl_vertex](#) (double x, double y)

Add a single vertex to the current path.
 - double [fl_width](#) (const char *txt)

Return the typographical width of a nul-terminated string using the current font face and size.

- double **fl_width** (const char *txt, int n)
Return the typographical width of a sequence of n characters using the current font face and size.
- double **fl_width** (unsigned int c)
Return the typographical width of a single character using the current font face and size.
- void **fl_xylene** (int x, int y, int x1)
Draw a horizontal line from (x,y) to $(x1,y)$.
- void **fl_xylene** (int x, int y, int x1, int y2)
Draw a horizontal line from (x,y) to $(x1,y)$, then vertical from $(x1,y)$ to $(x1,y2)$.
- void **fl_xylene** (int x, int y, int x1, int y2, int x3)
Draw a horizontal line from (x,y) to $(x1,y)$, then a vertical from $(x1,y)$ to $(x1,y2)$ and then another horizontal from $(x1,y2)$ to $(x3,y2)$.
- void **fl_yxline** (int x, int y, int y1)
Draw a vertical line from (x,y) to $(x,y1)$
- void **fl_yxline** (int x, int y, int y1, int x2)
Draw a vertical line from (x,y) to $(x,y1)$, then a horizontal from $(x,y1)$ to $(x2,y1)$.
- void **fl_yxline** (int x, int y, int y1, int x2, int y3)
Draw a vertical line from (x,y) to $(x,y1)$, then a horizontal from $(x,y1)$ to $(x2,y1)$, then another vertical from $(x2,y1)$ to $(x2,y3)$.

Variables

- char **fl_draw_shortcut**

12.41.1 Detailed Description

utility header to pull drawing functions together

12.42 fl_draw.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Portable drawing function header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00022 #ifndef fl_draw_H
00023 #define fl_draw_H
00024
00025 #include <FL/Enumerations.H> // color names
00026 #include <FL/Fl_Graphics_Driver.H> // fl_graphics_driver + Fl_Region
00027 #include <FL/Fl_Rect.H>
00028
00029 // Image class...
00030 class Fl_Image;
00031 class Fl_Window;
00032
00033 // Label flags...
00034 FL_EXPORT extern char fl_draw_shortcut;
00035
00040 // Colors:
00050 inline void fl_color(Fl_Color c) {
00051     fl_graphics_driver->color(c);
00052 } // select indexed color
00054 inline void fl_color(int c) {
00055     fl_color((Fl_Color)c);
00056 }
00067 inline void fl_color(uchar r, uchar g, uchar b) {
```

```

00068     fl_graphics_driver->color(r, g, b);
00069 }
00074 inline Fl_Color fl_color() {
00075     return fl_graphics_driver->color();
00076 }
00082 // clip:
00088 inline void fl_push_clip(int x, int y, int w, int h) {
00089     fl_graphics_driver->push_clip(x, y, w, h);
00090 }
00099 inline void fl_clip(int x, int y, int w, int h) {
00100     fl_graphics_driver->push_clip(x, y, w, h);
00101 }
00105 inline void fl_push_no_clip() {
00106     fl_graphics_driver->push_no_clip();
00107 }
00115 inline void fl_pop_clip() {
00116     fl_graphics_driver->pop_clip();
00117 }
00118
00132 inline int fl_not_clipped(int x, int y, int w, int h) {
00133     return fl_graphics_driver->not_clipped(x, y, w, h);
00134 }
00135
00178 inline int fl_clip_box(int x, int y, int w, int h, int &X, int &Y, int &W, int &H) {
00179     return fl_graphics_driver->clip_box(x, y, w, h, X, Y, W, H);
00180 }
00181
00183 inline void fl_restore_clip() {
00184     fl_graphics_driver->restore_clip();
00185 }
00186
00196 inline void fl_clip_region(Fl_Region r) {
00197     fl_graphics_driver->clip_region(r);
00198 }
00199
00206 inline Fl_Region fl_clip_region() {
00207     return fl_graphics_driver->clip_region();
00208 }
00209
00210
00211 // points:
00215 inline void fl_point(int x, int y) {
00216     fl_graphics_driver->point(x, y);
00217 }
00218
00219 // line type:
00252 inline void fl_line_style(int style, int width = 0, char *dashes = 0) {
00253     fl_graphics_driver->line_style(style, width, dashes);
00254 }
00255
00261 enum {
00262     FL_SOLID      = 0,
00263     FL_DASH       = 1,
00264     FL_DOT        = 2,
00265     FL_DASHDOT    = 3,
00266     FL_DASHDOTDOT = 4,
00267
00268     FL_CAP_FLAT   = 0x100,
00269     FL_CAP_ROUND  = 0x200,
00270     FL_CAP_SQUARE = 0x300,
00271
00272     FL_JOIN_MITER = 0x1000,
00273     FL_JOIN_ROUND = 0x2000,
00274     FL_JOIN_BEVEL = 0x3000
00275 };
00276
00282 inline void fl_antialias(int state) {
00283     fl_graphics_driver->antialias(state);
00284 }
00285
00287 inline int fl_antialias() {
00288     return fl_graphics_driver->antialias();
00289 }
00290
00291 // rectangles tweaked to exactly fill the pixel rectangle:
00292
00298 inline void fl_rect(int x, int y, int w, int h) {
00299     fl_graphics_driver->rect(x, y, w, h);
00300 }
00301
00307 inline void fl_rounded_rect(int x, int y, int w, int h, int r) {
00308     fl_graphics_driver->rounded_rect(x, y, w, h, r);
00309 }
00310
00316 inline void fl_rect(Fl_Rect r) {
00317     fl_rect(r.x(), r.y(), r.w(), r.h());
00318 }

```

```

00319
00326 inline void fl_focus_rect(int x, int y, int w, int h) {
00327     fl_graphics_driver->focus_rect(x, y, w, h);
00328 }
00329
00333 inline void fl_rect(int x, int y, int w, int h, Fl_Color c) {
00334     fl_color(c);
00335     fl_rect(x, y, w, h);
00336 }
00337
00339 inline void fl_rectf(int x, int y, int w, int h) {
00340     fl_graphics_driver->rectf(x, y, w, h);
00341 }
00342
00347 inline void fl_rounded_rectf(int x, int y, int w, int h, int r) {
00348     fl_graphics_driver->rounded_rectf(x, y, w, h, r);
00349 }
00350
00354 inline void fl_rectf(int x, int y, int w, int h, Fl_Color c) {
00355     fl_color(c);
00356     fl_rectf(x, y, w, h);
00357 }
00358
00360 inline void fl_rectf(Fl_Rect r) {
00361     fl_graphics_driver->rectf(r.x(), r.y(), r.w(), r.h());
00362 }
00363
00367 inline void fl_rectf(Fl_Rect r, Fl_Color c) {
00368     fl_color(c);
00369     fl_rectf(r);
00370 }
00371
00379 inline void fl_rectf(int x, int y, int w, int h, uchar r, uchar g, uchar b) {
00380     fl_graphics_driver->colored_rectf(x, y, w, h, r, g, b);
00381 }
00382
00389 inline void fl_rectf(Fl_Rect bb, uchar r, uchar g, uchar b) {
00390     fl_graphics_driver->colored_rectf(bb.x(), bb.y(), bb.w(), bb.h(), r, g, b);
00391 }
00392
00393 // line segments:
00397 inline void fl_line(int x, int y, int x1, int y1) {
00398     fl_graphics_driver->line(x, y, x1, y1);
00399 }
00403 inline void fl_line(int x, int y, int x1, int y1, int x2, int y2) {
00404     fl_graphics_driver->line(x, y, x1, y1, x2, y2);
00405 }
00406
00407 // closed line segments:
00411 inline void fl_loop(int x, int y, int x1, int y1, int x2, int y2) {
00412     fl_graphics_driver->loop(x, y, x1, y1, x2, y2);
00413 }
00417 inline void fl_loop(int x, int y, int x1, int y1, int x2, int y2, int x3, int y3) {
00418     fl_graphics_driver->loop(x, y, x1, y1, x2, y2, x3, y3);
00419 }
00420
00421 // filled polygons
00425 inline void fl_polygon(int x, int y, int x1, int y1, int x2, int y2) {
00426     fl_graphics_driver->polygon(x, y, x1, y1, x2, y2);
00427 }
00431 inline void fl_polygon(int x, int y, int x1, int y1, int x2, int y2, int x3, int y3) {
00432     fl_graphics_driver->polygon(x, y, x1, y1, x2, y2, x3, y3);
00433 }
00434
00435 // draw rectilinear lines, horizontal segment first:
00439 inline void fl_xyline(int x, int y, int x1) {
00440     fl_graphics_driver->xyline(x, y, x1);
00441 }
00445 inline void fl_xyline(int x, int y, int x1, int y2) {
00446     fl_graphics_driver->xyline(x, y, x1, y2);
00447 }
00452 inline void fl_xyline(int x, int y, int x1, int y2, int x3) {
00453     fl_graphics_driver->xyline(x, y, x1, y2, x3);
00454 }
00455
00456 // draw rectilinear lines, vertical segment first:
00460 inline void fl_yxline(int x, int y, int y1) {
00461     fl_graphics_driver->yxline(x, y, y1);
00462 }
00466 inline void fl_yxline(int x, int y, int y1, int x2) {
00467     fl_graphics_driver->yxline(x, y, y1, x2);
00468 }
00473 inline void fl_yxline(int x, int y, int y1, int x2, int y3) {
00474     fl_graphics_driver->yxline(x, y, y1, x2, y3);
00475 }
00476
00477 // circular lines and pie slices (code in fl_arci.C):

```

```

00503 inline void fl_arc(int x, int y, int w, int h, double a1, double a2) {
00504     fl_graphics_driver->arc(x, y, w, h, a1, a2);
00505 }
00521 inline void fl_pie(int x, int y, int w, int h, double a1, double a2) {
00522     fl_graphics_driver->pie(x, y, w, h, a1, a2);
00523 }
00525 FL_EXPORT void fl_chord(int x, int y, int w, int h, double a1, double a2); // nyi
00526
00527 // scalable drawing code (code in fl_vertex.cxx and fl_arc.cxx):
00532 inline void fl_push_matrix() {
00533     fl_graphics_driver->push_matrix();
00534 }
00538 inline void fl_pop_matrix() {
00539     fl_graphics_driver->pop_matrix();
00540 }
00545 inline void fl_scale(double x, double y) {
00546     fl_graphics_driver->mult_matrix(x, 0, 0, y, 0, 0);
00547 }
00552 inline void fl_scale(double x) {
00553     fl_graphics_driver->mult_matrix(x, 0, 0, x, 0, 0);
00554 }
00559 inline void fl_translate(double x, double y) {
00560     fl_graphics_driver->translate(x, y);
00561 }
00566 inline void fl_rotate(double d) {
00567     fl_graphics_driver->rotate(d);
00568 }
00572 inline void fl_load_identity() {
00573     fl_graphics_driver->load_identity();
00574 }
00579 inline void fl_load_matrix(double a, double b, double c, double d, double x, double y) {
00580     fl_graphics_driver->load_matrix(a, b, c, d, x, y);
00581 }
00588 inline void fl_mult_matrix(double a, double b, double c, double d, double x, double y) {
00589     fl_graphics_driver->mult_matrix(a, b, c, d, x, y);
00590 }
00594 inline void fl_begin_points() {
00595     fl_graphics_driver->begin_points();
00596 }
00600 inline void fl_begin_line() {
00601     fl_graphics_driver->begin_line();
00602 }
00606 inline void fl_begin_loop() {
00607     fl_graphics_driver->begin_loop();
00608 }
00612 inline void fl_begin_polygon() {
00613     fl_graphics_driver->begin_polygon();
00614 }
00619 inline void fl_vertex(double x, double y) {
00620     fl_graphics_driver->vertex(x, y);
00621 }
00630 inline void fl_curve(double X0, double Y0, double X1, double Y1, double X2, double Y2, double X3,
00631     double Y3) {
00632     fl_graphics_driver->curve(X0, Y0, X1, Y1, X2, Y2, X3, Y3);
00633 }
00668 inline void fl_arc(double x, double y, double r, double start, double end) {
00669     fl_graphics_driver->arc(x, y, r, start, end);
00670 }
00683 inline void fl_circle(double x, double y, double r) {
00684     fl_graphics_driver->circle(x, y, r);
00685 }
00689 inline void fl_end_points() {
00690     fl_graphics_driver->end_points();
00691 }
00695 inline void fl_end_line() {
00696     fl_graphics_driver->end_line();
00697 }
00701 inline void fl_end_loop() {
00702     fl_graphics_driver->end_loop();
00703 }
00707 inline void fl_end_polygon() {
00708     fl_graphics_driver->end_polygon();
00709 }
00723 inline void fl_begin_complex_polygon() {
00724     fl_graphics_driver->begin_complex_polygon();
00725 }
00732 inline void fl_gap() {
00733     fl_graphics_driver->gap();
00734 }
00738 inline void fl_end_complex_polygon() {
00739     fl_graphics_driver->end_complex_polygon();
00740 }
00741 // get and use transformed positions:
00746 inline double fl_transform_x(double x, double y) {
00747     return fl_graphics_driver->transform_x(x, y);
00748 }
00753 inline double fl_transform_y(double x, double y) {

```

```

00754     return fl_graphics_driver->transform_y(x, y);
00755 }
00760 inline double fl_transform_dx(double x, double y) {
00761     return fl_graphics_driver->transform_dx(x, y);
00762 }
00767 inline double fl_transform_dy(double x, double y) {
00768     return fl_graphics_driver->transform_dy(x, y);
00769 }
00774 inline void fl_transformed_vertex(double xf, double yf) {
00775     fl_graphics_driver->transformed_vertex(xf, yf);
00776 }
00777
00784 inline void fl_copy_offscreen(int x, int y, int w, int h, Fl_Offscreen pixmap, int srcx, int srcy) {
00785     fl_graphics_driver->copy_offscreen(x, y, w, h, pixmap, srcx, srcy);
00786 }
00787
00788 FL_EXPORT Fl_Offscreen fl_create_offscreen(int w, int h);
00789 FL_EXPORT void fl_begin_offscreen(Fl_Offscreen b);
00790 FL_EXPORT void fl_end_offscreen(void);
00791 FL_EXPORT void fl_delete_offscreen(Fl_Offscreen bitmap);
00792 FL_EXPORT void fl_rescale_offscreen(Fl_Offscreen &ctx);
00793
00798 /* NOTE: doxygen comments here to avoid triplication in os-specific sources */
00799
00800 // Fonts:
00801 /*
00802     Set the current font, which is then used in various drawing routines.
00803     Implemented and documented in src/fl_draw.cxx
00804 */
00805 FL_EXPORT void fl_font(Fl_Font face, Fl_Fontsize fsize);
00806
00811 inline Fl_Font fl_font() {
00812     return fl_graphics_driver->font();
00813 }
00818 inline Fl_Fontsize fl_size() {
00819     return fl_graphics_driver->size();
00820 }
00821
00822 // Information you can get about the current font:
00827 inline int fl_height() {
00828     return fl_graphics_driver->height();
00829 }
00830 FL_EXPORT int fl_height(int font, int size);
00835 inline int fl_descent() {
00836     return fl_graphics_driver->descent();
00837 }
00841 FL_EXPORT double fl_width(const char *txt);
00842
00846 inline double fl_width(const char *txt, int n) {
00847     return fl_graphics_driver->width(txt, n);
00848 }
00855 inline double fl_width(unsigned int c) {
00856     return fl_graphics_driver->width(c);
00857 }
00883 FL_EXPORT void fl_text_extents(const char *, int &dx, int &dy, int &w, int &h);
00884
00891 inline void fl_text_extents(const char *t, int n, int &dx, int &dy, int &w, int &h) {
00892     fl_graphics_driver->text_extents(t, n, dx, dy, w, h);
00893 }
00894
00895 // font encoding:
00896 // Note: doxygen comments here to avoid duplication for os-specific cases
00903 FL_EXPORT const char *fl_latin1_to_local(const char *t, int n = -1);
00910 FL_EXPORT const char *fl_local_to_latin1(const char *t, int n = -1);
00917 FL_EXPORT const char *fl_mac_roman_to_local(const char *t, int n = -1);
00924 FL_EXPORT const char *fl_local_to_mac_roman(const char *t, int n = -1);
00930 FL_EXPORT float fl_override_scale();
00931
00932 FL_EXPORT void fl_restore_scale(float s);
00933
00944 FL_EXPORT void fl_draw(const char *str, int x, int y);
00952 FL_EXPORT void fl_draw(int angle, const char *str, int x, int y);
00956 inline void fl_draw(const char *str, int n, int x, int y) {
00957     fl_graphics_driver->draw(str, n, x, y);
00958 }
00968 inline void fl_draw(int angle, const char *str, int n, int x, int y) {
00969     fl_graphics_driver->draw(angle, str, n, x, y);
00970 }
00974 inline void fl_rtl_draw(const char *str, int n, int x, int y) {
00975     fl_graphics_driver->rtl_draw(str, n, x, y);
00976 }
00977 FL_EXPORT void fl_measure(const char *str, int &x, int &y, int draw_symbols = 1);
01000 FL_EXPORT void fl_draw(const char *str, int x, int y, int w, int h,
01001                        Fl_Align align, Fl_Image *img = 0,
01002                        int draw_symbols = 1, int spacing = 0);
01018 FL_EXPORT void fl_draw(const char *str, int x, int y, int w, int h, Fl_Align align,
01019                        void (*callthis)(const char *, int, int, int),

```

```

01020         Fl_Image *img = 0, int draw_symbols = 1, int spacing = 0);
01021
01022 // boxtypes:
01023
01024 FL_EXPORT void fl_frame(const char *s, int x, int y, int w, int h);
01025 FL_EXPORT void fl_frame2(const char *s, int x, int y, int w, int h);
01026 FL_EXPORT void fl_draw_box(Fl_Boxtype, int x, int y, int w, int h, Fl_Color);
01027 FL_EXPORT void fl_draw_box_focus(Fl_Boxtype, int x, int y, int w, int h, Fl_Color, Fl_Color);
01028
01029 // basic GUI objects (check marks, arrows, more to come ...):
01030
01031 // Draw a check mark in the given color inside the bounding box bb.
01032 void fl_draw_check(Fl_Rect bb, Fl_Color col);
01033
01034 // Draw one or more "arrows" (triangles)
01035 FL_EXPORT void fl_draw_arrow(Fl_Rect bb, Fl_Arrow_Type t, Fl_Orientation o, Fl_Color color);
01036
01037 // Draw a potentially small, filled circle
01038 FL_EXPORT void fl_draw_circle(int x, int y, int d, Fl_Color color);
01039
01040 // Draw the full "radio button" of a radio menu entry or radio button
01041 // This requires scheme specific handling (particularly gtk+ scheme)
01042 FL_EXPORT void fl_draw_radio(int x, int y, int d, Fl_Color color);
01043
01044 // images:
01045
01081 inline void fl_draw_image(const uchar *buf, int X, int Y, int W, int H, int D = 3, int L = 0) {
01082     fl_graphics_driver->draw_image_general_(buf, X, Y, W, H, D, L);
01083 }
01084
01089 inline void fl_draw_image_mono(const uchar *buf, int X, int Y, int W, int H, int D = 1, int L = 0) {
01090     fl_graphics_driver->draw_image_mono_general_(buf, X, Y, W, H, D, L);
01091 }
01092
01128 inline void fl_draw_image(Fl_Draw_Image_Cb cb, void *data, int X, int Y, int W, int H, int D = 3) {
01129     fl_graphics_driver->draw_image(cb, data, X, Y, W, H, D);
01130 }
01131
01136 inline void fl_draw_image_mono(Fl_Draw_Image_Cb cb, void *data, int X, int Y, int W, int H, int D = 1)
01137 {
01138     fl_graphics_driver->draw_image_mono(cb, data, X, Y, W, H, D);
01139 }
01145 inline char fl_can_do_alpha_blending() {
01146     return Fl_Graphics_Driver::default_driver().can_do_alpha_blending();
01147 }
01148
01149 FL_EXPORT uchar *fl_read_image(uchar *p, int X, int Y, int W, int H, int alpha = 0);
01150 FL_EXPORT Fl_RGB_Image *fl_capture_window(Fl_Window *win, int x, int y, int w, int h);
01151
01152 // pixmaps:
01164 FL_EXPORT int fl_draw_pixmap(const char *const *data, int x, int y, Fl_Color bg = FL_GRAY);
01169 inline int fl_draw_pixmap(/*const*/ char *const *data, int x, int y, Fl_Color bg = FL_GRAY) {
01170     return fl_draw_pixmap((const char *const *)data, x, y, bg);
01171 }
01172 FL_EXPORT int fl_measure_pixmap(/*const*/ char *const *data, int &w, int &h);
01173 FL_EXPORT int fl_measure_pixmap(const char *const *cdata, int &w, int &h);
01174
01175 // other:
01176 FL_EXPORT void fl_scroll(int X, int Y, int W, int H, int dx, int dy,
01177     void (*draw_area)(void *, int, int, int, int), void *data);
01178 FL_EXPORT const char *fl_shortcut_label(unsigned int shortcut);
01179 FL_EXPORT const char *fl_shortcut_label(unsigned int shortcut, const char **eom);
01180 FL_EXPORT unsigned int fl_old_shortcut(const char *s);
01181 FL_EXPORT void fl_overlay_rect(int x, int y, int w, int h);
01182 FL_EXPORT void fl_overlay_clear();
01183 FL_EXPORT void fl_cursor(Fl_Cursor);
01184 FL_EXPORT void fl_cursor(Fl_Cursor, Fl_Color fg, Fl_Color bg = FL_WHITE);
01185 FL_EXPORT const char *fl_expand_text(const char *from, char *buf, int maxbuf, double maxw,
01186     int &n, double &width, int wrap, int draw_symbols = 0);
01187
01188 // XIM:
01189 FL_EXPORT void fl_set_status(int X, int Y, int W, int H);
01197 FL_EXPORT void fl_set_spot(int font, int size, int X, int Y, int W, int H, Fl_Window *win = 0);
01209 FL_EXPORT void fl_reset_spot(void);
01210
01211
01212 // XForms symbols:
01213 FL_EXPORT int fl_draw_symbol(const char *label, int x, int y, int w, int h, Fl_Color);
01214 FL_EXPORT int fl_add_symbol(const char *name, void (*drawit)(Fl_Color), int scalable);
01217 #endif

```

12.43 Fl_Export.H

```
00001 /*
```

```

00002  * Windows DLL export .
00003  *
00004  * Copyright 1998-2018 by Bill Spitzak and others.
00005  *
00006  * This library is free software. Distribution and use rights are outlined in
00007  * the file "COPYING" which should have been included with this file. If this
00008  * file is missing or damaged, see the license at:
00009  *
00010  *      https://www.fltk.org/COPYING.php
00011  *
00012  * Please see the following page on how to report bugs and issues:
00013  *
00014  *      https://www.fltk.org/bugs.php
00015  */
00016
00017 #ifndef Fl_Export_H
00018 #   define Fl_Export_H
00019
00020 /*
00021  * The following is used when building DLLs under Windows
00022  * and when building .so's under unix/linux.
00023  */
00024
00025 #   if defined(FL_DLL)
00026 #       ifdef FL_LIBRARY
00027 #           define FL_EXPORT __declspec(dllexport)
00028 #       else
00029 #           define FL_EXPORT __declspec(dllimport)
00030 #       endif /* FL_LIBRARY */
00031 #       elif __GNUC__ >= 4
00032 #           define FL_EXPORT __attribute__ ((visibility ("default")))
00033 #       else
00034 #           define FL_EXPORT
00035 #       endif /* FL_DLL */
00036
00037 #endif /* !Fl_Export_H */

```

12.44 Fl_File_Browser.H

```

00001 //
00002 // FileBrowser definitions.
00003 //
00004 // Copyright 1999-2010 by Michael Sweet.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_File_Browser widget . */
00019
00020 //
00021 // Include necessary header files...
00022 //
00023
00024 #ifndef _Fl_File_Browser_H_
00025 #   define _Fl_File_Browser_H_
00026
00027 #   include "Fl_Browser.H"
00028 #   include "Fl_File_Icon.H"
00029 #   include "filename.H"
00030
00031
00032 //
00033 // Fl_File_Browser class...
00034 //
00035
00037 class FL_EXPORT Fl_File_Browser : public Fl_Browser {
00038
00039     int             filetype_;
00040     const char      *directory_;
00041     uchar           iconsize_;
00042     const char      *pattern_;
00043     const char      *errmsg_;
00044
00045     int             full_height() const FL_OVERRIDE;
00046     int             item_height(void *) const FL_OVERRIDE;
00047     int             item_width(void *) const FL_OVERRIDE;

```

```

00048 void item_draw(void *, int, int, int, int) const FL_OVERRIDE;
00049 int incr_height() const FL_OVERRIDE { return (item_height(0) + linespacing()); }
00050
00051 public:
00052     enum { FILES, DIRECTORIES };
00053
00054     Fl_File_Browser(int, int, int, int, const char * = 0);
00055     ~Fl_File_Browser();
00056
00058     uchar iconsze() const { return (iconsze_); }
00060     void iconsze(uchar s) { iconsze_ = s; redraw(); }
00061
00067     void filter(const char *pattern);
00073     const char *filter() const { return (pattern_); }
00074     int load(const char *directory, Fl_File_Sort_F *sort = fl_numericsort);
00075     Fl_Fontsize textsize() const { return Fl_Browser::textsize(); }
00076     void textsize(Fl_Fontsize s) { Fl_Browser::textsize(s); iconsze_ = (uchar)(3 * s / 2); }
00077
00084     int filetype() const { return (filetype_); }
00091     void filetype(int t) { filetype_ = t; }
00092     void errmsg(const char *errmsg);
00096     const char* errmsg() const { return errmsg_; }
00097 };
00098
00099 #endif // !_Fl_File_Browser_H_

```

12.45 Fl_File_Chooser.H

```

00001 //
00002 // Fl_File_Chooser dialog for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 // =====
00017 // DO NOT EDIT FL/Fl_File_Chooser.H and src/Fl_File_Chooser.cxx !!!
00018 // =====
00019 // Please use fluid to change src/Fl_File_Chooser.fl interactively
00020 // and then use fluid to "write code" or edit and use fluid -c .
00021 // =====
00022 //
00023
00024 // generated by Fast Light User Interface Designer (fluid) version 1.0500
00025
00026 #ifndef Fl_File_Chooser_H
00027 #define Fl_File_Chooser_H
00028 #include <FL/Fl.H>
00029 #include <FL/Fl_Double_Window.H>
00030 #include <FL/Fl_Group.H>
00031 #include <FL/Fl_Choice.H>
00032 #include <FL/Fl_Menu_Button.H>
00033 #include <FL/Fl_Button.H>
00034 #include <FL/Fl_Preferences.H>
00035 #include <FL/Fl_Tile.H>
00036 #include <FL/Fl_File_Browser.H>
00037 #include <FL/Fl_Box.H>
00038 #include <FL/Fl_Check_Button.H>
00039 #include <FL/Fl_File_Input.H>
00040 #include <FL/Fl_Return_Button.H>
00041 #include <FL/fl_ask.H>
00042
00043 class FL_EXPORT Fl_File_Chooser {
00044 public:
00049     enum Type {
00050         SINGLE = 0,
00051         MULTI = 1,
00052         CREATE = 2,
00055         DIRECTORY = 4
00056     };
00057 private:
00058     static Fl_Preferences *prefs_;
00059     void (*callback_)(Fl_File_Chooser*, void *);
00060     void *data_;
00061     char directory_[FL_PATH_MAX];
00062     char pattern_[FL_PATH_MAX];
00063     char preview_text_[2048];

```



```

00064     int type_;
00065     void favoritesButtonCB();
00066     void favoritesCB(Fl_Widget *w);
00067     void fileListCB();
00068     void fileNameCB();
00069     void newdir();
00070     static void previewCB(Fl_File_Chooser *fc);
00071     void showChoiceCB();
00072     void update_favorites();
00073     void update_preview();
00074 public:
00075     Fl_File_Chooser(const char *pathname, const char *pattern, int type_val, const char *title);
00076 private:
00077     Fl_Double_Window *window;
00078     inline void cb_window_i(Fl_Double_Window*, void*);
00079     static void cb_window(Fl_Double_Window*, void*);
00080     Fl_Choice *showChoice;
00081     inline void cb_showChoice_i(Fl_Choice*, void*);
00082     static void cb_showChoice(Fl_Choice*, void*);
00083     Fl_Menu_Button *favoritesButton;
00084     inline void cb_favoritesButton_i(Fl_Menu_Button*, void*);
00085     static void cb_favoritesButton(Fl_Menu_Button*, void*);
00086 public:
00087     Fl_Button *newButton;
00088 private:
00089     inline void cb_newButton_i(Fl_Button*, void*);
00090     static void cb_newButton(Fl_Button*, void*);
00091     inline void cb_i(Fl_File*, void*);
00092     static void cb_(Fl_File*, void*);
00093     Fl_File_Browser *fileList;
00094     inline void cb_fileList_i(Fl_File_Browser*, void*);
00095     static void cb_fileList(Fl_File_Browser*, void*);
00096     Fl_Box *errorBox;
00097     Fl_Box *previewBox;
00098 public:
00099     Fl_Check_Button *previewButton;
00100 private:
00101     inline void cb_previewButton_i(Fl_Check_Button*, void*);
00102     static void cb_previewButton(Fl_Check_Button*, void*);
00103 public:
00104     Fl_Check_Button *showHiddenButton;
00105 private:
00106     inline void cb_showHiddenButton_i(Fl_Check_Button*, void*);
00107     static void cb_showHiddenButton(Fl_Check_Button*, void*);
00108     Fl_File_Input *fileName;
00109     inline void cb_fileName_i(Fl_File_Input*, void*);
00110     static void cb_fileName(Fl_File_Input*, void*);
00111     Fl_Return_Button *okButton;
00112     inline void cb_okButton_i(Fl_Return_Button*, void*);
00113     static void cb_okButton(Fl_Return_Button*, void*);
00114     Fl_Button *cancelButton;
00115     inline void cb_cancelButton_i(Fl_Button*, void*);
00116     static void cb_cancelButton(Fl_Button*, void*);
00117     Fl_Double_Window *favWindow;
00118     Fl_File_Browser *favList;
00119     inline void cb_favList_i(Fl_File_Browser*, void*);
00120     static void cb_favList(Fl_File_Browser*, void*);
00121     Fl_Button *favUpButton;
00122     inline void cb_favUpButton_i(Fl_Button*, void*);
00123     static void cb_favUpButton(Fl_Button*, void*);
00124     Fl_Button *favDeleteButton;
00125     inline void cb_favDeleteButton_i(Fl_Button*, void*);
00126     static void cb_favDeleteButton(Fl_Button*, void*);
00127     Fl_Button *favDownButton;
00128     inline void cb_favDownButton_i(Fl_Button*, void*);
00129     static void cb_favDownButton(Fl_Button*, void*);
00130     Fl_Button *favCancelButton;
00131     inline void cb_favCancelButton_i(Fl_Button*, void*);
00132     static void cb_favCancelButton(Fl_Button*, void*);
00133     Fl_Return_Button *favOkButton;
00134     inline void cb_favOkButton_i(Fl_Return_Button*, void*);
00135     static void cb_favOkButton(Fl_Return_Button*, void*);
00136 public:
00137     ~Fl_File_Chooser();
00138     void callback(void (*cb)(Fl_File_Chooser *, void *), void *d = 0);
00139     void color(Fl_Color c);
00140     Fl_Color color();
00141     int count();
00142     void directory(const char *d);
00143     char * directory();
00144     void filter(const char *p);
00145     const char * filter();
00146     int filter_value();
00147     void filter_value(int f);
00148     void iconsize(uchar s);
00149     uchar iconsize();
00150     void label(const char *l);

```

```

00151     const char * label();
00152     void ok_label(const char *l);
00153     const char * ok_label();
00154     void preview(int e);
00155     int preview() const { return previewButton->value(); }
00156 private:
00157     void showHidden(int e);
00158     void remove_hidden_files();
00159 public:
00160     void rescan();
00161     void rescan_keep_filename();
00162     void show();
00163     void hide();
00164     int shown();
00165     void textcolor(Fl_Color c);
00166     Fl_Color textcolor();
00167     void textfont(Fl_Font f);
00168     Fl_Font textfont();
00169     void textsize(Fl_Fontsize s);
00170     Fl_Fontsize textsize();
00171     void type(int t);
00172     int type();
00173     void * user_data() const;
00174     void user_data(void *d);
00175     const char *value(int f = 1);
00176     void value(const char *filename);
00177     int visible();
00178     void position(int x, int y);
00179     int x() const;
00180     int y() const;
00181     int w() const;
00182     int h() const;
00183     void size(int w, int h);
00184     void resize(int x, int y, int w, int h);
00188     static const char *add_favorites_label;
00192     static const char *all_files_label;
00196     static const char *custom_filter_label;
00200     static const char *existing_file_label;
00204     static const char *favorites_label;
00208     static const char *filename_label;
00212     static const char *filesystems_label;
00216     static const char *manage_favorites_label;
00220     static const char *new_directory_label;
00224     static const char *new_directory_tooltip;
00228     static const char *preview_label;
00232     static const char *save_label;
00236     static const char *show_label;
00240     static const char *hidden_label;
00245     static Fl_File_Sort_F *sort;
00246 private:
00247     Fl_Widget* ext_group;
00248 public:
00249     Fl_Widget* add_extra(Fl_Widget* gr);
00250 protected:
00251     void show_error_box(int val);
00252 };
00253 FL_EXPORT char *fl_dir_chooser(const char *message, const char *fname, int relative=0);
00254 FL_EXPORT char *fl_file_chooser(const char *message, const char *pat, const char *fname, int relative=0);
00255 FL_EXPORT void fl_file_chooser_callback(void (*cb)(const char*));
00256 FL_EXPORT void fl_file_chooser_ok_label(const char *l);
00257 #endif

```

12.46 Fl_File_Icon.H

```

00001 //
00002 // Fl_File_Icon definitions.
00003 //
00004 // Copyright 1999-2010 by Michael Sweet.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 /* \file
00017    Fl_File_Icon widget . */
00018
00019
00020 //
00021 // Include necessary header files...

```

```

00022 //
00023
00024 #ifndef _Fl_Fl_File_Icon_H_
00025 # define _Fl_Fl_File_Icon_H_
00026
00027 # include "Fl.H"
00028
00029
00030 //
00031 // Special color value for the icon color.
00032 //
00033
00034 # define FL_ICON_COLOR (Fl_Color)0xffffffff
00037 //
00038 // Fl_File_Icon class...
00039 //
00040
00041 class FL_EXPORT Fl_File_Icon {
00042
00043     static Fl_File_Icon *first_; // Pointer to first icon/filetype
00044     Fl_File_Icon *next_; // Pointer to next icon/filetype
00045     const char *pattern_; // Pattern string
00046     int type_; // Match only if directory or file?
00047     int num_data_; // Number of data elements
00048     int alloc_data_; // Number of allocated elements
00049     short *data_; // Icon data
00050
00051 public:
00052
00053     enum // File types
00054     {
00055         ANY, // Any kind of file
00056         PLAIN, // Only plain files
00057         FIFO, // Only named pipes
00058         DEVICE, // Only character and block devices
00059         LINK, // Only symbolic links
00060         DIRECTORY, // Only directories
00061     };
00062
00063     enum // Data opcodes
00064     {
00065         END, // End of primitive/icon
00066         COLOR, // Followed by color value (2 shorts)
00067         LINE, // Start of line
00068         CLOSEDLINE, // Start of closed line
00069         POLYGON, // Start of polygon
00070         OUTLINEPOLYGON, // Followed by outline color (2 shorts)
00071         VERTEX, // Followed by scaled X,Y
00072     };
00073
00074     Fl_File_Icon(const char *p, int t, int nd = 0, short *d = 0);
00075     ~Fl_File_Icon();
00076
00077     short *add(short d);
00078
00079     short *add_color(Fl_Color c)
00080     { short *d = add((short)COLOR); add((short)(c >> 16)); add((short)c); return (d); }
00081
00082     short *add_vertex(int x, int y)
00083     { short *d = add((short)VERTEX); add((short)x); add((short)y); return (d); }
00084
00085     short *add_vertex(float x, float y)
00086     { short *d = add((short)VERTEX); add((short)(x * 10000.0));
00087       add((short)(y * 10000.0)); return (d); }
00088
00089     void clear() { num_data_ = 0; }
00090
00091     void draw(int x, int y, int w, int h, Fl_Color ic, int active = 1);
00092
00093     void label(Fl_Widget *w);
00094
00095     static void labeltype(const Fl_Label *o, int x, int y, int w, int h, Fl_Align a);
00096     void load(const char *f);
00097     int load_fti(const char *fti);
00098     int load_image(const char *i);
00099
00100     Fl_File_Icon *next() { return (next_); }
00101
00102     const char *pattern() { return (pattern_); }
00103
00104     int size() { return (num_data_); }
00105
00106     int type() { return (type_); }
00107
00108     short *value() { return (data_); }
00109
00110     static Fl_File_Icon *find(const char *filename, int filetype = ANY);

```

```

00147
00149     static Fl_File_Icon *first() { return (first_); }
00150     static void    load_system_icons(void);
00151 };
00152
00153 #endif // !_Fl_Fl_File_Icon_H_

```

12.47 Fl_File_Input.H

```

00001 //
00002 // File_Input header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 // Original version Copyright 1998 by Curtis Edwards.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00018 /* \file
00019     Fl_File_Input widget . */
00020
00021 #ifndef Fl_File_Input_H
00022 #    define Fl_File_Input_H
00023
00024 #    include <FL/Fl_Input.H>
00025
00045 class FL_EXPORT Fl_File_Input : public Fl_Input {
00046
00047     char            ok_entry_;
00048     uchar           down_box_;
00049     short           buttons_[200];
00050     short           pressed_;
00051
00052     void            draw_buttons();
00053     int             handle_button(int event);
00054     void            update_buttons();
00055
00056 public:
00057
00058     Fl_File_Input(int X, int Y, int W, int H, const char *L=0);
00059
00060     int handle(int event) FL_OVERRIDE;
00061
00062 protected:
00063     void draw() FL_OVERRIDE;
00064
00065 public:
00067     Fl_Boxtype      down_box() const { return (Fl_Boxtype)down_box_; }
00069     void            down_box(Fl_Boxtype b) { down_box_ = b; }
00070
00080     Fl_Color        errorcolor() const { return FL_RED; }
00081
00090     void            errorcolor(Fl_Color c) {(void)c;}
00091
00092     int value(const char *str);
00093     int value(const char *str, int len);
00094
00099     const char      *value() { return Fl_Input_::value(); }
00100 };
00101
00102 #endif // !Fl_File_Input_H

```

12.48 Fl_Fill_Dial.H

```

00001 //
00002 // Filled dial header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php

```

```

00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Fill_Dial widget . */
00019
00020 #ifndef Fl_Fill_Dial_H
00021 #define Fl_Fill_Dial_H
00022
00023 #include "Fl_Dial.H"
00024
00026 class FL_EXPORT Fl_Fill_Dial : public Fl_Dial {
00027 public:
00029     Fl_Fill_Dial(int X,int Y,int W,int H, const char *L = NULL);
00030 };
00031
00032 #endif

```

12.49 Fl_Fill_Slider.H

```

00001 //
00002 // Filled slider header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Fill_Slider widget . */
00019
00020 #ifndef Fl_Fill_Slider_H
00021 #define Fl_Fill_Slider_H
00022
00023 #include "Fl_Slider.H"
00025 class FL_EXPORT Fl_Fill_Slider : public Fl_Slider {
00026 public:
00028     Fl_Fill_Slider(int X,int Y,int W,int H,const char *L=0);
00029 };
00030
00031 #endif

```

12.50 Fl_Flex.H

```

00001 //
00002 // Fl_Flex widget header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2020 by Karsten Pedersen
00005 // Copyright 2022-2024 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file.  If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00018 #ifndef Fl_Flex_H
00019 #define Fl_Flex_H
00020
00021 #include <FL/Fl_Group.H>
00022
00014 class FL_EXPORT Fl_Flex : public Fl_Group {
00015
00016     int margin_left_;           // left margin
00017     int margin_top_;            // top margin
00018     int margin_right_;          // right margin

```

```

00119     int margin_bottom_;           // bottom margin
00120     int gap_;                     // gap between widgets
00121     int fixed_size_size_;         // number of fixed size widgets in array
00122     int fixed_size_alloc_;        // allocated size of fixed size array
00123     Fl_Widget **fixed_size_;      // array of fixed size widgets
00124     bool need_layout_;            // true if layout needs to be calculated
00125
00126 public:
00127
00128     enum { // values for type(int)
00129         VERTICAL    = 0,
00130         HORIZONTAL  = 1,
00131         COLUMN      = 0,
00132         ROW         = 1
00133     };
00134
00135     // FLTK standard constructor
00136     Fl_Flex(int X, int Y, int W, int H, const char *L = 0);
00137
00138     // original Fl_Flex constructors:
00139     // backwards compatible if direction enums { ROW | COLUMN } are used
00140
00141     Fl_Flex(int direction);
00142     Fl_Flex(int w, int h, int direction);
00143     Fl_Flex(int x, int y, int w, int h, int direction);
00144
00145     virtual ~Fl_Flex();
00146
00147     virtual void end();
00148     void resize(int x, int y, int w, int h) FL_OVERRIDE;
00149
00150     void fixed(Fl_Widget &w, int size) {
00151         fixed(&w, size);
00152     }
00153
00154     void fixed(Fl_Widget *w, int size);
00155     int fixed(Fl_Widget *w) const;
00156
00157 protected:
00158
00159     void init(int t = VERTICAL);
00160
00161     virtual int alloc_size(int size) const;
00162
00163     void on_remove(int) FL_OVERRIDE;
00164     void draw() FL_OVERRIDE;
00165
00166 public:
00167
00168     void need_layout(int set) {
00169         if (set) need_layout_ = true;
00170         else need_layout_ = false;
00171     }
00172
00173     bool need_layout() const {
00174         return need_layout_;
00175     }
00176
00177     int margin() const { return margin_left_; }
00178
00179     int margin(int *left, int *top, int *right, int *bottom) const {
00180         if (left) *left = margin_left_;
00181         if (top) *top = margin_top_;
00182         if (right) *right = margin_right_;
00183         if (bottom) *bottom = margin_bottom_;
00184         if (margin_left_ == margin_top_ && margin_top_ == margin_right_ && margin_right_ ==
margin_bottom_)
00185             return 1;
00186         return 0;
00187     }
00188
00189     void margin(int m, int g = -1) {
00190         if (m < 0)
00191             m = 0;
00192         margin_left_ = margin_top_ = margin_right_ = margin_bottom_ = m;
00193         if (g >= 0)
00194             gap_ = g;
00195         need_layout(1);
00196     }
00197
00198     void margin(int left, int top, int right, int bottom) {
00199         margin_left_ = left < 0 ? 0 : left;
00200         margin_top_ = top < 0 ? 0 : top;
00201         margin_right_ = right < 0 ? 0 : right;
00202         margin_bottom_ = bottom < 0 ? 0 : bottom;
00203         need_layout(1);
00204     }

```

```

00289
00293     int gap() const {
00294         return gap_;
00295     }
00296
00305     void gap(int g) {
00306         gap_ = g < 0 ? 0 : g;
00307         need_layout(1);
00308     }
00309
00318     int horizontal() const {
00319         return type() == Fl_Flex::HORIZONTAL ? 1 : 0;
00320     }
00321
00322     // Calculate the layout of the widget and redraw it.
00323     void layout();
00324
00334     int spacing() const {
00335         return gap_;
00336     }
00337
00347     void spacing(int i) {
00348         gap(i);
00349         need_layout(1);
00350     }
00351
00352 };
00353
00354 #endif // Fl_Flex_H

```

12.51 Fl_Float_Input.H

```

00001 //
00002 // Floating point input header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2011 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Float_Input widget . */
00019
00020 #ifndef Fl_Float_Input_H
00021 #define Fl_Float_Input_H
00022
00023 #include "Fl_Input.H"
00024
00030 class FL_EXPORT Fl_Float_Input : public Fl_Input {
00031 public:
00038     Fl_Float_Input(int X,int Y,int W,int H,const char *l = 0);
00039 };
00040
00041 #endif

```

12.52 Fl_FormsBitmap.H

```

00001 //
00002 // Forms bitmap header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file

```

```

00018     Fl_FormsBitmap widget . */
00019
00020 #ifndef Fl_FormsBitmap_H
00021 #define Fl_FormsBitmap_H
00022
00023 #include "Fl_Bitmap.H"
00024
00028 class FL_EXPORT Fl_FormsBitmap : public Fl_Widget {
00029     Fl_Bitmap *b;
00030 protected:
00031     void draw() FL_OVERRIDE;
00032 public:
00033     Fl_FormsBitmap(Fl_Boxtype, int, int, int, int, const char * = 0);
00034     void set(int W, int H, const uchar *bits);
00036     void bitmap(Fl_Bitmap *B) {b = B;}
00038     Fl_Bitmap *bitmap() const {return b;}
00039 };
00040
00041 #endif

```

12.53 Fl_FormsPixmap.H

```

00001 //
00002 // Forms pixmap header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_FormsPixmap widget . */
00019
00020 #ifndef Fl_FormsPixmap_H
00021 #define Fl_FormsPixmap_H
00022
00023 #include "Fl_Pixmap.H"
00024
00029 class FL_EXPORT Fl_FormsPixmap : public Fl_Widget {
00030     Fl_Pixmap *b;
00031 protected:
00032     void draw() FL_OVERRIDE;
00033 public:
00034     Fl_FormsPixmap(Fl_Boxtype t, int X, int Y, int W, int H, const char *L= 0);
00035
00036     void set(/*const*/char * const * bits);
00037
00042     void Pixmap(Fl_Pixmap *B) {b = B;}
00043
00045     Fl_Pixmap *Pixmap() const {return b;}
00046 };
00047
00048 #endif

```

12.54 Fl_Free.H

```

00001 //
00002 // Forms free header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Free widget . */

```



```

00019
00020 #ifndef Fl_Free_H
00021 #define Fl_Free_H
00022
00023 #ifndef Fl_Widget_H
00024 #include "Fl_Widget.H"
00025 #endif
00026
00027 #define FL_NORMAL_FREE      1
00028 #define FL_SLEEPING_FREE    2
00029 #define FL_INPUT_FREE       3
00030 #define FL_CONTINUOUS_FREE  4
00031 #define FL_ALL_FREE         5
00032 typedef int (*FL_HANDLEPTR)(Fl_Widget *, int , float, float, char);
00033
00034 class FL_EXPORT Fl_Free : public Fl_Widget {
00035     FL_HANDLEPTR hfunc;
00036     static void step(void *);
00037 protected:
00038     void draw() FL_OVERRIDE;
00039 public:
00040     int handle(int e) FL_OVERRIDE;
00041     Fl_Free(uchar t,int X,int Y,int W,int H,const char *L,FL_HANDLEPTR hdl);
00042     ~Fl_Free();
00043 };
00044
00045 // old event names for compatibility:
00046 #define FL_MOUSE      FL_DRAG
00047 #define FL_DRAW       100
00048 #define FL_STEP       101
00049 #define FL_FREEMEM    102
00050 #define FL_FREEZE     103
00051 #define FL_THAW       104
00052 #endif

```

12.55 Fl_GIF_Image.H

```

00001 //
00002 // GIF image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_GIF_Image widget . */
00019
00020 #ifndef Fl_GIF_Image_H
00021 #define Fl_GIF_Image_H
00022 #include "Fl_Pixmap.H"
00023
00024 class FL_EXPORT Fl_GIF_Image : public Fl_Pixmap {
00025 public:
00026     Fl_GIF_Image(const char* filename);
00027     // deprecated constructor w/o length (for backwards compatibility)
00028     Fl_GIF_Image(const char* imagename, const unsigned char *data);
00029     // constructor with length (since 1.4.0)
00030     Fl_GIF_Image(const char* imagename, const unsigned char *data, const size_t length);
00031
00032     static bool is_animated(const char *name_);
00033     static bool animate;
00034
00035 protected:
00036     // Protected constructors needed for animated GIF support through Fl_Anim_GIF_Image.
00037     Fl_GIF_Image(const char* filename, bool anim);
00038     Fl_GIF_Image(const char* imagename, const unsigned char *data, const size_t length, bool anim);
00039     // Protected default constructor needed for Fl_Anim_GIF_Image.
00040     Fl_GIF_Image();
00041
00042     void load_gif_(class Fl_Image_Reader &rdr, bool anim=false);
00043
00044     void load(const char* filename, bool anim);
00045     void load(const char* imagename, const unsigned char *data, const size_t length, bool anim);

```

```

00058
00059 // Internal structure to "glue" animated GIF support into Fl_GIF_Image.
00060 // This data is passed during decoding to the Fl_Anim_GIF_Image class.
00061 struct GIF_FRAME {
00062     int ifrm, width, height, x, y, w, h,
00063     clr, bkgd, trans,
00064     dispose, delay;
00065     const uchar *bptr;
00066     const struct CPAL {
00067         uchar r, g, b;
00068     } *cpal;
00069     GIF_FRAME(int frame, uchar *data) : ifrm(frame), bptr(data) {}
00070     GIF_FRAME(int frame, int W, int H, int fx, int fy, int fw, int fh, uchar *data) :
00071         ifrm(frame), width(W), height(H), x(fx), y(fy), w(fw), h(fh), bptr(data) {}
00072     void disposal(int mode, int time) { dispose = mode; this->delay = time; }
00073     void colors(int nclr, int bg, int tp) { clr = nclr; bkgd = bg; trans = tp; }
00074 };
00075
00076 // Internal virtual methods, which are called during decoding to pass data
00077 // to the Fl_Anim_GIF_Image class.
00078 virtual void on_frame_data(GIF_FRAME &) {}
00079 virtual void on_extension_data(GIF_FRAME &) {}
00080
00081 private:
00082
00083     void lzw_decode(Fl_Image_Reader &rdr, uchar *Image, int Width, int Height, int CodeSize, int
        ColorMapSize, int Interlace);
00084 };
00085
00086 #endif

```

12.56 Fl_Gl_Window.H

```

00001 //
00002 // OpenGL header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Gl_Window widget . */
00019
00020 #ifndef Fl_Gl_Window_H
00021 #define Fl_Gl_Window_H
00022
00023 #include "Fl_Window.H"
00024
00025 class Fl_Gl_Choice; // structure to hold result of glXChooseVisual
00026 class Fl_Gl_Window_Driver;
00027 class FL_EXPORT Fl_Gl_Window : public Fl_Window {
00028     friend class Fl_Gl_Window_Driver;
00029     Fl_Gl_Window_Driver *pGlWindowDriver;
00030
00031     int mode_;
00032     const int *alist;
00033     Fl_Gl_Choice *g;
00034     GLContext context_;
00035     char valid_f_;
00036     char damage1_; // damage() of back buffer
00037     virtual void draw_overlay();
00038     void init();
00039
00040     void *overlay;
00041
00042     static int can_do(int, const int *);
00043     int mode(int, const int *);
00044     static int gl_plugin_linkage();
00045 protected:
00046     void draw_begin();
00047     void draw() FL_OVERRIDE;
00048     void draw_end();
00049
00050 public:
00051     void show() FL_OVERRIDE;
00052     void show(int a, char **b) {Fl_Window::show(a,b);}

```

```

00083 void flush() FL_OVERRIDE;
00084 void hide() FL_OVERRIDE;
00085 void resize(int,int,int,int) FL_OVERRIDE;
00086 int handle(int) FL_OVERRIDE;
00087
00112 char valid() const {return valid_f_ & 1;}
00116 void valid(char v) {if (v) valid_f_ |= 1; else valid_f_ &= 0xfe;}
00117 void invalidate();
00118
00125 char context_valid() const {return valid_f_ & 2;}
00129 void context_valid(char v) {if (v) valid_f_ |= 2; else valid_f_ &= 0xfd;}
00130
00132 static int can_do(int m) {return can_do(m,0);}
00135 static int can_do(const int *m) {return can_do(0, m);}
00137 int can_do() {return can_do(mode_,alist);}
00141 Fl_Mode mode() const {return (Fl_Mode)mode_;}
00186 int mode(int a) {return mode(a,0);}
00198 int mode(const int *a) {return mode(0, a);}
00202 GLContext context() const {return context_;}
00203 void context(GLContext, int destroy_flag = 0);
00204 void make_current();
00205 void swap_buffers();
00206 void swap_interval(int);
00207 int swap_interval() const;
00208 void ortho();
00209
00210 int can_do_overlay();
00211 void redraw_overlay();
00212 void hide_overlay();
00213 void make_overlay_current();
00214
00215 // Note: Doxygen docs in Fl_Widget.H to avoid redundancy.
00216 Fl_Gl_Window* as_gl_window() FL_OVERRIDE { return this; }
00217 Fl_Gl_Window const* as_gl_window() const FL_OVERRIDE { return this; }
00218
00219 float pixels_per_unit();
00229 int pixel_w() { return int(pixels_per_unit() * w() + 0.5f); }
00239 int pixel_h() { return int(pixels_per_unit() * h() + 0.5f); }
00240
00241 ~Fl_Gl_Window();
00246 Fl_Gl_Window(int W, int H, const char *l=0) : Fl_Window(W,H,l) {init();}
00253 Fl_Gl_Window(int X, int Y, int W, int H, const char *l=0)
00254 : Fl_Window(X,Y,W,H,l) {init();}
00255 };
00256
00257 #endif // Fl_Gl_Window_H

```

12.57 Fl_Graphics_Driver.H

```

00001 //
00002 // Declaration of classes Fl_Graphics_Driver, Fl_Scalable_Graphics_Driver,
00003 // and Fl_Font_Descriptor for the Fast Light Tool Kit (FLTK).
00004 //
00005 // Copyright 2010-2024 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 // https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 // https://www.fltk.org/bugs.php
00016 //
00017
00028 #ifndef FL_GRAPHICS_DRIVER_H
00029 #define FL_GRAPHICS_DRIVER_H
00030
00031 #include <FL/Fl_Device.H>
00032 #include <FL/Fl_Image.H>
00033 #include <FL/Fl_Bitmap.H>
00034 #include <FL/Fl_Pixmap.H>
00035 #include <FL/Fl_RGB_Image.H>
00036
00037 class Fl_Graphics_Driver;
00038 class Fl_Font_Descriptor;
00039 class Fl_Image_Surface;
00040 FL_EXPORT extern Fl_Graphics_Driver *fl_graphics_driver;
00041
00050 typedef void (*Fl_Draw_Image_Cb)(void* data,int x,int y,int w,uchar* buf);
00051
00052 struct Fl_Fontdesc;
00053 typedef struct _PangoFontDescription PangoFontDescription;
00054

```

```

00055 // FIXME: The following constants are deprecated and will be removed in FLTK 1.5.0
00056 // in favor of dynamic clipping stack allocation. This needs C++11 features.
00057 // See issue #1139: "FL_REGION_STACK_SIZE could be increased"
00058 // and issue #1140: "Fix static array allocation".
00059
00060 #if FL_ABI_VERSION >= 10401
00061 # define FL_REGION_STACK_SIZE 64
00062 #else
00063 # define FL_REGION_STACK_SIZE 10
00064 #endif
00065
00066 #define FL_MATRIX_STACK_SIZE 32
00067
00086 class FL_EXPORT Fl_Graphics_Driver {
00087     friend class Fl_Surface_Device;
00088     friend class Fl_Pixmap;
00089     friend class Fl_Bitmap;
00090     friend class Fl_RGB_Image;
00091     friend class Fl_SVG_Image;
00092     friend void fl_draw_image(const uchar* buf, int X,int Y,int W,int H, int D, int L);
00093     friend void fl_draw_image_mono(const uchar* buf, int X,int Y,int W,int H, int D, int L);
00094     friend void fl_draw_image_mono(Fl_Draw_Image_Cb cb, void* data, int X,int Y,int W,int H, int D);
00095     friend void fl_draw_image(Fl_Draw_Image_Cb cb, void* data, int X,int Y,int W,int H, int D);
00096     friend void fl_copy_offscreen(int x, int y, int w, int h, Fl_Offscreen pixmap, int srcx, int srcy);
00097     friend int fl_convert_pixmap(const char*const* cdata, uchar* out, Fl_Color bg);
00098     friend FL_EXPORT int fl_draw_pixmap(const char*const* cdata, int x, int y, Fl_Color bg);
00099     friend FL_EXPORT void gl_start();
00100     /* ===== Implementation note about image drawing =====
00101     A graphics driver can implement up to 6 virtual member functions to draw images:
00102     virtual void draw_pixmap(Fl_Pixmap *pxm,int XP, int YP, int WP, int HP, int cx, int cy)
00103     virtual void draw_bitmap(Fl_Bitmap *bm,int XP, int YP, int WP, int HP, int cx, int cy)
00104     virtual void draw_rgb(Fl_RGB_Image *rgb,int XP, int YP, int WP, int HP, int cx, int cy)
00105 and
00106     virtual void draw_fixed(Fl_Pixmap *pxm,int XP, int YP, int WP, int HP, int cx, int cy)
00107     virtual void draw_fixed(Fl_Bitmap *bm,int XP, int YP, int WP, int HP, int cx, int cy)
00108     virtual void draw_fixed(Fl_RGB_Image *rgb,int XP, int YP, int WP, int HP, int cx, int cy)
00109     - The 1st group of functions is used when the driver can directly map the image data,
00110     sized at data_w() x data_h(), to the image drawing area, sized at w()*scale x h()*scale
00111     where scale is the current GUI scale factor.
00112     - If the driver does not support such scale-and-draw operation for a given image type,
00113     it should implement the draw_fixed() function which is called by the library after the
00114     image has been internally resized to the drawing size and cached.
00115     - The platform-independent Fl_Graphics_Driver class implements the 1st group of functions.
00116     Each resizes the image, caches it, and calls the platform-specific implementation of
00117     draw_fixed(image-class *,...) with the cached image.
00118     - Consider an image object, say from class Fl_RGB_Image. Fl_RGB_Image::draw()
00119     calls the virtual member function draw_rgb(Fl_RGB_Image *,...). If Fl_XXX_Graphics_Driver
00120     re-implements this function, this code runs and is expected to draw the image
00121     adequately scaled to its drawing size. If Fl_XXX_Graphics_Driver does not re-implement
00122     this function, Fl_Graphics_Driver::draw_rgb(Fl_RGB_Image *,...) runs. It internally
00123     resizes the image, caches it, and calls Fl_XXX_Graphics_Driver::draw_fixed(Fl_RGB_Image *,...)
00124     that draws the image from its cached form which already has the adequate size.
00125     - Some drivers implement, for a given image class, only the function of the 1st group or
00126     only draw_fixed() as in these examples:
00127     - Fl_Quartz_Graphics_Driver implements only draw_rgb(Fl_RGB_Image *,...) because it
00128     can perform the scale-and-draw operation whatever the RGB image and the required scaling.
00129     - Fl_Xlib_Graphics_Driver implements only draw_fixed(Fl_Pixmap *,...). The library
00130     takes care of resizing and caching the Pixmap to the adequate drawing size.
00131     - Some drivers implement, for a given image class, the function of both groups, e.g. :
00132     Fl_GDI_Graphics_Driver implements both draw_rgb(Fl_RGB_Image *,...) and
00133     draw_fixed(Fl_RGB_Image *,...) because scale-and-draw may require function Alphablend()
00134     from MSIMG32.DLL. In the absence of that, the draw_rgb() implementation calls
00135     Fl_Graphics_Driver::draw_rgb() which runs Fl_GDI_Graphics_Driver::draw_fixed(Fl_RGB_Image*,...).
00136
00137     Graphics drivers also implement cache(Fl_Pixmap*), cache(Fl_Bitmap*) and cache(Fl_RGB_Image*)
00138     to compute the cached form of all image types, and uncache(Fl_RGB_Image *,...),
00139     uncache_pixmap(fl_uintptr_t) and delete_bitmask(fl_uintptr_t) to destroy cached image forms.
00140     Graphics drivers that use the mask_ variable of class Fl_Pixmap to cache an Fl_Pixmap
00141     object also reimplement the uchar **Fl_Graphics_Driver::mask_bitmap() member function.
00142     */
00143 private:
00144     virtual void draw_fixed(Fl_Pixmap *pxm,int XP, int YP, int WP, int HP, int cx, int cy);
00145     virtual void draw_fixed(Fl_Bitmap *bm,int XP, int YP, int WP, int HP, int cx, int cy);
00146     virtual void draw_fixed(Fl_RGB_Image *rgb,int XP, int YP, int WP, int HP, int cx, int cy);
00147     // the default implementation of make_unused_color_() is most probably enough
00148     virtual void make_unused_color_(unsigned char &r, unsigned char &g, unsigned char &b, int
color_count, void **data);
00149     // some platforms may need to reimplement this
00150     virtual void set_current_();
00151     void draw_image_general_(const uchar *buf, int X, int Y, int W, int H, int D, int L);
00152     void draw_image_mono_general_(const uchar *buf, int X, int Y, int W, int H, int D, int L);
00153     float scale_; // scale between FLTK and drawing coordinates: drawing = FLTK * scale_
00154 public:
00155     static Fl_Graphics_Driver *newMainGraphicsDriver();
00156     struct matrix {double a, b, c, d, x, y;};
00157     typedef enum {
00158         NATIVE = 1,

```

```

00162     PRINTER = 2;
00163 } driver_feature;
00164
00165 protected:
00166     int fl_clip_state_number;
00167     static const matrix m0;
00168     Fl_Font font_;
00169     Fl_Fontsize size_;
00170     Fl_Color color_;
00171     int sptr;
00172     static const int matrix_stack_size = FL_MATRIX_STACK_SIZE;
00173     matrix stack[FL_MATRIX_STACK_SIZE];
00174     matrix m;
00175     int n;
00176     int gap_;
00177     enum SHAPE {NONE=0, LINE, LOOP, POLYGON, POINTS, COMPLEX_POLYGON} what;
00178     int rstackptr;
00179     static const int region_stack_max = FL_REGION_STACK_SIZE - 1;
00180     Fl_Region rstack[FL_REGION_STACK_SIZE];
00181     Fl_Font_Descriptor *font_descriptor_;
00182     int p_size;
00183     typedef struct { float x; float y; } XPOINT;
00184     XPOINT *xpoint;
00185     virtual void global_gc();
00186     virtual void cache(Fl_Pixmap *img);
00187     virtual void cache(Fl_Bitmap *img);
00188     virtual void cache(Fl_RGB_Image *img);
00189     virtual void uncache(Fl_RGB_Image *img, fl_uintptr_t &id_, fl_uintptr_t &mask_);
00190     // --- implementation is in src/drivers/xxx/Fl_xxx_Graphics_Driver_image.cxx
00191     virtual void draw_image(const uchar* buf, int X,int Y,int W,int H, int D=3, int L=0);
00192     virtual void draw_image_mono(const uchar* buf, int X,int Y,int W,int H, int D=1, int L=0);
00193     virtual void draw_image(Fl_Draw_Image_Cb cb, void* data, int X,int Y,int W,int H, int D=3);
00194     virtual void draw_image_mono(Fl_Draw_Image_Cb cb, void* data, int X,int Y,int W,int H, int D=1);
00195     virtual void draw_rgb(Fl_RGB_Image * rgb,int XP, int YP, int WP, int HP, int cx, int cy);
00196     virtual void draw_pixmap(Fl_Pixmap * pxm,int XP, int YP, int WP, int HP, int cx, int cy);
00197     virtual void draw_bitmap(Fl_Bitmap *bm, int XP, int YP, int WP, int HP, int cx, int cy);
00198     virtual void copy_offscreen(int x, int y, int w, int h, Fl_Offscreen pixmap, int srcx, int srcy);
00199
00200     static void change_image_size(Fl_Image *img, int W, int H) {
00201         img->w(W);
00202         img->h(H);
00203     }
00204
00205     // Support function for image drawing
00206     virtual void uncache_pixmap(fl_uintptr_t p);
00207     // accessor functions to protected image members
00208     int start_image(Fl_Image *img, int XP, int YP, int WP, int HP, int &cx, int &cy,
00209         int &X, int &Y, int &W, int &H);
00210     static fl_uintptr_t* id(Fl_RGB_Image *rgb) {return &(rgb->id_);}
00211     static fl_uintptr_t* id(Fl_Pixmap *pm) {return &(pm->id_);}
00212     static fl_uintptr_t* id(Fl_Bitmap *bm) {return &(bm->id_);}
00213     static fl_uintptr_t* mask(Fl_RGB_Image *rgb) {return &(rgb->mask_);}
00214     static fl_uintptr_t* mask(Fl_Pixmap *pm) {return &(pm->mask_);}
00215     static void cache_w_h(Fl_Pixmap *pm, int*& pwidth, int*& pheight) {
00216         pwidth = &(pm->cache_w_);
00217         pheight = &(pm->cache_h_);
00218     }
00219     static void cache_w_h(Fl_Bitmap *bm, int*& pwidth, int*& pheight) {
00220         pwidth = &(bm->cache_w_);
00221         pheight = &(bm->cache_h_);
00222     }
00223     static void cache_w_h(Fl_RGB_Image *rgb, int*& pwidth, int*& pheight) {
00224         pwidth = &(rgb->cache_w_);
00225         pheight = &(rgb->cache_h_);
00226     }
00227     static Fl_Offscreen get_offscreen_and_delete_image_surface(Fl_Image_Surface*);
00228     static void draw_empty(Fl_Image* img, int X, int Y) {img->draw_empty(X, Y);}
00229
00230     Fl_Graphics_Driver();
00231     virtual void cache_size(Fl_Image *img, int &width, int &height);
00232     void cache_size_finalize(Fl_Image *img, int &width, int &height);
00233     static unsigned need_pixmap_bg_color;
00234 public:
00235     virtual ~Fl_Graphics_Driver();
00236     static Fl_Graphics_Driver &default_driver();
00237     // support of "complex shapes"
00238     void push_matrix();
00239     void pop_matrix();
00240     void load_identity();
00241     void load_matrix(double a, double b, double c, double d, double x, double y);
00242     void mult_matrix(double a, double b, double c, double d, double x, double y);
00243     void rotate(double d);
00244     void translate(double x,double y);
00245     double transform_x(double x, double y);
00246     double transform_y(double x, double y);
00247     double transform_dx(double x, double y);
00248     double transform_dy(double x, double y);
00249     inline Fl_Font_Descriptor *font_descriptor() { return font_descriptor_;}

```

```

00261 inline void font_descriptor(Fl_Font_Descriptor *d) { font_descriptor_ = d;}
00263 float scale() { return scale_; }
00265 virtual void scale(float f);
00267 virtual char can_do_alpha_blending();
00268 virtual void point(int x, int y);
00269 virtual void rect(int x, int y, int w, int h);
00270 virtual void focus_rect(int x, int y, int w, int h);
00271 virtual void rectf(int x, int y, int w, int h);
00272 virtual void _rbox(int fill, int x, int y, int w, int h, int r);
00273 virtual void rounded_rect(int x, int y, int w, int h, int r);
00274 virtual void rounded_rectf(int x, int y, int w, int h, int r);
00275 // the default implementation is most likely enough
00276 virtual void colored_rectf(int x, int y, int w, int h, uchar r, uchar g, uchar b);
00277 virtual void line(int x, int y, int x1, int y1);
00279 virtual void line(int x, int y, int x1, int y1, int x2, int y2);
00281 virtual void xyline(int x, int y, int x1);
00283 virtual void xyline(int x, int y, int x1, int y2);
00285 virtual void xyline(int x, int y, int x1, int y2, int x3);
00287 virtual void yxline(int x, int y, int y1);
00289 virtual void yxline(int x, int y, int y1, int x2);
00291 virtual void yxline(int x, int y, int y1, int x2, int y3);
00293 virtual void loop(int x0, int y0, int x1, int y1, int x2, int y2);
00295 virtual void loop(int x0, int y0, int x1, int y1, int x2, int y2, int x3, int y3);
00296 virtual void polygon(int x0, int y0, int x1, int y1, int x2, int y2);
00298 virtual void polygon(int x0, int y0, int x1, int y1, int x2, int y2, int x3, int y3);
00299 // --- clipping
00300 virtual void push_clip(int x, int y, int w, int h);
00301 virtual int clip_box(int x, int y, int w, int h, int &X, int &Y, int &W, int &H);
00302 virtual int not_clipped(int x, int y, int w, int h);
00304 virtual void push_no_clip(); // has default implementation
00306 virtual void pop_clip(); // has default implementation
00307 virtual Fl_Region clip_region(); // has default implementation
00308 virtual void clip_region(Fl_Region r); // has default implementation
00309 virtual void restore_clip();
00310 virtual void begin_points();
00311 virtual void begin_line();
00312 virtual void begin_loop();
00313 virtual void begin_polygon();
00314 virtual void begin_complex_polygon();
00315 virtual void transformed_vertex(double xf, double yf);
00316 virtual void transformed_vertex0(float x, float y);
00317 virtual void vertex(double x, double y);
00318 virtual void end_points();
00319 virtual void end_line();
00320 virtual void end_loop();
00321 virtual void fixloop();
00322 virtual void end_polygon();
00323 virtual void end_complex_polygon();
00324 // default implementation is most probably enough
00325 virtual bool can_fill_non_convex_polygon() { return true; }
00326 virtual void gap();
00327 virtual void circle(double x, double y, double r);
00328 virtual void arc(double x, double y, double r, double start, double end);
00329 virtual void arc(int x, int y, int w, int h, double a1, double a2);
00330 virtual void pie(int x, int y, int w, int h, double a1, double a2);
00331 // To support fl_draw_circle(int x, int y, int d, Fl_Color color),
00332 // the default implementation is most probably enough.
00333 virtual void draw_circle(int x, int y, int d, Fl_Color c);
00334 virtual void curve(double X0, double Y0, double X1, double Y1, double X2, double Y2, double X3,
double Y3);
00335 virtual void line_style(int style, int width=0, char* dashes=0);
00336 virtual void color(Fl_Color c);
00337 virtual void set_color(Fl_Color i, unsigned int c);
00338 virtual void free_color(Fl_Color i, int overlay);
00339 virtual Fl_Color color();
00340 virtual void color(uchar r, uchar g, uchar b);
00341 virtual void draw(const char *str, int nChars, int x, int y);
00342 virtual void draw(const char *str, int nChars, float x, float y);
00343 virtual void draw(int angle, const char *str, int nChars, int x, int y);
00344 virtual void rtl_draw(const char *str, int nChars, int x, int y);
00345 virtual int has_feature(driver_feature feature);
00346 virtual void font(Fl_Font face, Fl_Fontsize fsize);
00347 virtual Fl_Font font();
00348 virtual Fl_Fontsize size();
00349 virtual double width(const char *str, int nChars);
00350 virtual double width(unsigned int c);
00351 virtual void text_extents(const char*, int n, int& dx, int& dy, int& w, int& h);
00352 virtual int height();
00353 virtual int descent();
00354 virtual void gc(void*);
00355 virtual void *gc(void);
00356 virtual uchar **mask_bitmap();
00357 // default implementation may be enough
00358 virtual float scale_font_for_PostScript(Fl_Font_Descriptor *desc, int s);
00359 // default implementation may be enough
00360 virtual float scale_bitmap_for_PostScript();
00361 // each platform implements these 3 functions its own way

```

```

00362     virtual void add_rectangle_to_region(Fl_Region r, int x, int y, int w, int h);
00363     virtual Fl_Region XRectangleRegion(int x, int y, int w, int h);
00364     virtual void XDestroyRegion(Fl_Region r);
00365     virtual const char* get_font_name(Fl_Font fnum, int* ap);
00366     virtual int get_font_sizes(Fl_Font fnum, int*& sizep);
00367     virtual Fl_Font set_fonts(const char *name);
00368     virtual Fl_Fontdesc* calc_fl_fonts(void);
00369     virtual unsigned font_desc_size();
00370     virtual const char *font_name(int num);
00371     virtual void font_name(int num, const char *name);
00372     // Default implementation may be enough
00373     virtual void overlay_rect(int x, int y, int w, int h);
00374     virtual float override_scale();
00375     virtual void restore_scale(float);
00376     virtual PangoFontDescription* pango_font_description() { return NULL; }
00377     virtual void antialias(int state);
00378     virtual int antialias();
00379     virtual void delete_bitmask(fl_uintptr_t bm);
00380 };
00381
00382 #ifndef FL_DOXYGEN
00383
00384 /* This class is not part of FLTK's public API.
00385 Platforms usually define a derived class called Fl_XXX_Font_Descriptor
00386 containing extra platform-specific data/functions.
00387 This is a class for an actual system font, with junk to
00388 help choose it and info on character sizes. Each Fl_Fontdesc has a
00389 linked list of these. These are created the first time each system
00390 font/size combination is used.
00391 */
00392 class Fl_Font_Descriptor {
00393 public:
00394     Fl_Font_Descriptor *next;
00395     Fl_Fontsize size;
00396     Fl_Font_Descriptor(const char* fontname, Fl_Fontsize size);
00397     virtual FL_EXPORT ~Fl_Font_Descriptor() {}
00398     int ascent, descent;
00399     unsigned int listbase; // base of display list, 0 = none
00400 };
00401
00402
00403 // This struct is not part of FLTK's public API.
00404 struct Fl_Fontdesc {
00405     const char *name;
00406     char fontname[128]; // "Pretty" font name
00407     Fl_Font_Descriptor *first; // linked list of sizes of this style
00408 };
00409
00410 /* Abstract class Fl_Scalable_Graphics_Driver is platform-independent.
00411 It supports the scaling of all graphics coordinates by a
00412 float factor helpful to support HiDPI displays.
00413 This class does :
00414 - compute scaled coordinates
00415 - scale the cached offscreen of image objects
00416 - scale the pixel arrays used when performing direct image draws
00417 - call the member functions of a platform-specific,
00418 Fl_Scalable_Graphics_Driver-derived class that do the drawings with adequately
00419 scaled coordinates. The member functions are named with the _unscaled suffix.
00420 - scale and unscale the clipping region.
00421
00422 This class is presently used by the X11 and Windows platforms to support HiDPI displays.
00423 In the future, it may also be used by other platforms.
00424 */
00425 class FL_EXPORT Fl_Scalable_Graphics_Driver : public Fl_Graphics_Driver {
00426     Fl_Fontsize fontsize_; // scale-independent font size value
00427 public:
00428     Fl_Scalable_Graphics_Driver();
00429     static int floor(int x, float s);
00430     inline int floor(int x) { return Fl_Scalable_Graphics_Driver::floor(x, scale()); }
00431 protected:
00432     int line_width_;
00433     bool is_solid_;
00434     virtual Fl_Region scale_clip(float f);
00435     void unscale_clip(Fl_Region r);
00436     void point(int x, int y) FL_OVERRIDE;
00437     virtual void point_unscaled(float x, float y);
00438     void rect(int x, int y, int w, int h) FL_OVERRIDE;
00439     void rectf(int x, int y, int w, int h) FL_OVERRIDE;
00440     virtual void rect_unscaled(int x, int y, int w, int h);
00441     virtual void rectf_unscaled(int x, int y, int w, int h);
00442     void line(int x, int y, int x1, int y1) FL_OVERRIDE;
00443     virtual void line_unscaled(int x, int y, int x1, int y1);
00444     void line(int x, int y, int x1, int y1, int x2, int y2) FL_OVERRIDE;
00445     virtual void line_unscaled(int x, int y, int x1, int y1, int x2, int y2);
00446     void xyline(int x, int y, int x1) FL_OVERRIDE;
00447     virtual void xyline_unscaled(int x, int y, int x1);
00448     void xyline(int x, int y, int x1, int y2) FL_OVERRIDE {Fl_Graphics_Driver::xyline(x, y, x1, y2);}
00449     void xyline(int x, int y, int x1, int y2, int x3) FL_OVERRIDE {Fl_Graphics_Driver::xyline(x, y, x1,

```



```

        y2, x3);}
00450     void yxline(int x, int y, int y1) FL_OVERRIDE;
00451     virtual void yxline_unscaled(int x, int y, int y1);
00452     void yxline(int x, int y, int y1, int x2) FL_OVERRIDE {Fl_Graphics_Driver::yxline(x, y, y1, x2);}
00453     void yxline(int x, int y, int y1, int x2, int y3) FL_OVERRIDE {Fl_Graphics_Driver::yxline(x, y, y1,
x2, y3);}
00454     void loop(int x0, int y0, int x1, int y1, int x2, int y2) FL_OVERRIDE;
00455     virtual void loop_unscaled(int x0, int y0, int x1, int y1, int x2, int y2);
00456     void loop(int x0, int y0, int x1, int y1, int x2, int y2, int x3, int y3) FL_OVERRIDE;
00457     virtual void loop_unscaled(int x0, int y0, int x1, int y1, int x2, int y2, int x3, int y3);
00458     void polygon(int x0, int y0, int x1, int y1, int x2, int y2) FL_OVERRIDE;
00459     virtual void polygon_unscaled(int x0, int y0, int x1, int y1, int x2, int y2);
00460     void polygon(int x0, int y0, int x1, int y1, int x2, int y2, int x3, int y3) FL_OVERRIDE;
00461     virtual void polygon_unscaled(int x0, int y0, int x1, int y1, int x2, int y2, int x3, int y3);
00462     void circle(double x, double y, double r) FL_OVERRIDE;
00463     virtual void ellipse_unscaled(double xt, double yt, double rx, double ry);
00464     void font(Fl_Font face, Fl_Fontsize size) FL_OVERRIDE;
00465     Fl_Font font() FL_OVERRIDE;
00466     virtual void font_unscaled(Fl_Font face, Fl_Fontsize size);
00467     double width(const char *str, int n) FL_OVERRIDE;
00468     double width(unsigned int c) FL_OVERRIDE;
00469     virtual double width_unscaled(const char *str, int n);
00470     virtual double width_unscaled(unsigned int c);
00471     Fl_Fontsize size() FL_OVERRIDE;
00472     virtual Fl_Fontsize size_unscaled();
00473     void text_extents(const char *str, int n, int &dx, int &dy, int &w, int &h) FL_OVERRIDE;
00474     virtual void text_extents_unscaled(const char *str, int n, int &dx, int &dy, int &w, int &h);
00475     int height() FL_OVERRIDE;
00476     int descent() FL_OVERRIDE;
00477     virtual int height_unscaled();
00478     virtual int descent_unscaled();
00479     void draw(const char *str, int n, int x, int y) FL_OVERRIDE;
00480     virtual void draw_unscaled(const char *str, int n, int x, int y);
00481     void draw(int angle, const char *str, int n, int x, int y) FL_OVERRIDE;
00482     virtual void draw_unscaled(int angle, const char *str, int n, int x, int y);
00483     void draw(const char *str, int nChars, float x, float y) FL_OVERRIDE;
00484     void rtl_draw(const char* str, int n, int x, int y) FL_OVERRIDE;
00485     virtual void rtl_draw_unscaled(const char* str, int n, int x, int y);
00486     void arc(double x, double y, double r, double start, double end) FL_OVERRIDE;
00487     void arc(int x, int y, int w, int h, double a1, double a2) FL_OVERRIDE;
00488     virtual void arc_unscaled(int x, int y, int w, int h, double a1, double a2);
00489     void pie(int x, int y, int w, int h, double a1, double a2) FL_OVERRIDE;
00490     virtual void pie_unscaled(int x, int y, int w, int h, double a1, double a2);
00491     void draw_circle(int x, int y, int d, Fl_Color c) FL_OVERRIDE;
00492     void line_style(int style, int width=0, char* dashes=0) FL_OVERRIDE;
00493     virtual void line_style_unscaled(int style, int width, char* dashes);
00494     void draw_image_rescale(void *buf, Fl_Draw_Image_Cb cb, int X, int Y, int W, int H, int D, int L,
bool mono);
00495     virtual void draw_image_unscaled(const uchar* buf, int X,int Y,int W,int H, int D=3, int L=0);
00496     virtual void draw_image_unscaled(Fl_Draw_Image_Cb cb, void* data, int X,int Y,int W,int H, int D=3);
00497     void draw_image(const uchar* buf, int X,int Y,int W,int H, int D=3, int L=0) FL_OVERRIDE;
00498     void draw_image(Fl_Draw_Image_Cb cb, void* data, int X,int Y,int W,int H, int D=3) FL_OVERRIDE;
00499     virtual void draw_image_mono_unscaled(const uchar* buf, int x, int y, int w, int h, int d, int l);
00500     void draw_image_mono(const uchar* buf, int X,int Y,int W,int H, int D=1, int L=0) FL_OVERRIDE;
00501     virtual void draw_image_mono_unscaled(Fl_Draw_Image_Cb cb, void* data, int X,int Y,int W,int H, int
D=1);
00502     void draw_image_mono(Fl_Draw_Image_Cb cb, void* data, int X,int Y,int W,int H, int D=1) FL_OVERRIDE;
00503
00504     void transformed_vertex(double xf, double yf) FL_OVERRIDE;
00505     void vertex(double x, double y) FL_OVERRIDE;
00506     float override_scale() FL_OVERRIDE;
00507     void restore_scale(float) FL_OVERRIDE;
00508     virtual void *change_pen_width(int lwidth);
00509     virtual void reset_pen_width(void *data);
00510 };
00511 #endif // FL_DOXYGEN
00512
00513 #endif // FL_GRAPHICS_DRIVER_H
00514

```

12.58 Fl_Grid.H File Reference

[Fl_Grid](#) container widget.

```
#include <FL/Fl_Group.H>
```

```
#include <FL/Fl_Rect.H>
```

Classes

- class [Fl_Grid::Cell](#)

- class `Fl_Grid`

`Fl_Grid` is a container (layout) widget with multiple columns and rows.

Typedefs

- typedef unsigned short `Fl_Grid_Align`

`Fl_Grid` type for child widget alignment control.

Variables

- const `Fl_Grid_Align` `FL_GRID_BOTTOM` = 0x0002
Align the widget at the bottom of the cell.
- const `Fl_Grid_Align` `FL_GRID_BOTTOM_LEFT` = `FL_GRID_BOTTOM` | `FL_GRID_LEFT`
- const `Fl_Grid_Align` `FL_GRID_BOTTOM_RIGHT` = `FL_GRID_BOTTOM` | `FL_GRID_RIGHT`
- const `Fl_Grid_Align` `FL_GRID_CENTER` = 0x0000
Align the widget in the middle of the cell (default).
- const `Fl_Grid_Align` `FL_GRID_FILL` = 0x0030
Stretch the widget in both directions to fill the cell.
- const `Fl_Grid_Align` `FL_GRID_HORIZONTAL` = 0x0010
Stretch the widget horizontally to fill the cell.
- const `Fl_Grid_Align` `FL_GRID_LEFT` = 0x0004
Align the widget at the left side of the cell.
- const `Fl_Grid_Align` `FL_GRID_PROPORTIONAL` = 0x0040
Stretch the widget proportionally.
- const `Fl_Grid_Align` `FL_GRID_RIGHT` = 0x0008
Align the widget at the right side of the cell.
- const `Fl_Grid_Align` `FL_GRID_TOP` = 0x0001
Align the widget at the top of the cell.
- const `Fl_Grid_Align` `FL_GRID_TOP_LEFT` = `FL_GRID_TOP` | `FL_GRID_LEFT`
- const `Fl_Grid_Align` `FL_GRID_TOP_RIGHT` = `FL_GRID_TOP` | `FL_GRID_RIGHT`
- const `Fl_Grid_Align` `FL_GRID_VERTICAL` = 0x0020
Stretch the widget vertically to fill the cell.

12.58.1 Detailed Description

`Fl_Grid` container widget.

12.59 Fl_Grid.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Fl_Grid widget header for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2021-2022 by Albrecht Schlosser.
00005 // Copyright 2022-2025 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00018 #ifndef _FL_FL_GRID_H_
00019 #define _FL_FL_GRID_H_
00020
00025 #include <FL/Fl_Group.H>
00026 #include <FL/Fl_Rect.H>
```

```

00027
00029 typedef unsigned short Fl_Grid_Align;
00030
00032 const Fl_Grid_Align FL_GRID_CENTER          = 0x0000;
00033
00035 const Fl_Grid_Align FL_GRID_TOP              = 0x0001;
00036
00038 const Fl_Grid_Align FL_GRID_BOTTOM          = 0x0002;
00039
00041 const Fl_Grid_Align FL_GRID_LEFT            = 0x0004;
00042
00044 const Fl_Grid_Align FL_GRID_RIGHT           = 0x0008;
00045
00047 const Fl_Grid_Align FL_GRID_HORIZONTAL      = 0x0010;
00048
00050 const Fl_Grid_Align FL_GRID_VERTICAL        = 0x0020;
00051
00053 const Fl_Grid_Align FL_GRID_FILL            = 0x0030;
00054
00056 const Fl_Grid_Align FL_GRID_PROPORTIONAL    = 0x0040;
00057
00058 const Fl_Grid_Align FL_GRID_TOP_LEFT        = FL_GRID_TOP | FL_GRID_LEFT;
00059 const Fl_Grid_Align FL_GRID_TOP_RIGHT       = FL_GRID_TOP | FL_GRID_RIGHT;
00060 const Fl_Grid_Align FL_GRID_BOTTOM_LEFT     = FL_GRID_BOTTOM | FL_GRID_LEFT;
00061 const Fl_Grid_Align FL_GRID_BOTTOM_RIGHT    = FL_GRID_BOTTOM | FL_GRID_RIGHT;
00062
00147 class FL_EXPORT Fl_Grid : public Fl_Group {
00148     friend class Fl_Grid_Type;
00149
00150 public:
00151     class Cell {
00152         friend class Fl_Grid;
00153     private:
00154         Cell *next_;           // next cell in the same row
00155         short row_;            // row number
00156         short col_;            // column number
00157         short rowspan_;        // row span (1 - n)
00158         short colspan_;        // column span (1 - n)
00159         Fl_Grid_Align align_;   // widget alignment in its cell
00160         Fl_Widget *widget_;    // assigned widget
00161         int w_;                // minimal widget width
00162         int h_;                // minimal widget height
00163
00164     public:
00165
00166         void Cell_() {          // common initialization
00167             next_ = NULL;
00168             row_ = 0;
00169             col_ = 0;
00170             rowspan_ = 1;
00171             colspan_ = 1;
00172             widget_ = NULL;
00173             w_ = 0;
00174             h_ = 0;
00175             align_ = 0;
00176         }
00177
00178         Cell(int row, int col) { // constructor
00179             Cell_();
00180             row_ = row;
00181             col_ = col;
00182         }
00183
00184         Cell(Fl_Widget *w, int row, int col) { // widget assignment
00185             Cell_();
00186             widget_ = w;
00187             row_ = row;
00188             col_ = col;
00189         }
00190
00191         ~Cell() {}
00192
00202         Cell *next() {
00203             return next_;
00204         }
00205
00217         void next(Cell *c) {
00218             next_ = c;
00219         }
00220
00221         Fl_Widget *widget() const { return widget_; }
00222
00223         short row() const { return row_; }
00224         short col() const { return col_; }
00225
00226         void rowspan(short v) { rowspan_ = v; }
00227         void colspan(short v) { colspan_ = v; }

```

```

00228     short rowspan() const { return rowspan_; }
00229     short colspan() const { return colspan_; }
00230
00231     void align(Fl_Grid_Align align) { align_ = align; }
00232     Fl_Grid_Align align() const { return align_; }
00233
00234     void minimum_size(int w, int h) { if (w>=0) w_ = w; if (h>=0) h_ = h; }
00235     void minimum_size(int *w, int *h) const { if (w) *w = w_; if (h) *h = h_; }
00236 }; // class Cell
00237
00238 private:
00239     class Row;
00240     class Col;
00241     short rows_;
00242     short cols_;
00243
00244     short margin_left_;           // left margin
00245     short margin_top_;           // top margin
00246     short margin_right_;         // right margin
00247     short margin_bottom_;        // bottom margin
00248     short gap_row_;              // gap between rows
00249     short gap_col_;              // gap between columns
00250     Fl_Rect old_size;             // only for resize callback (TBD)
00251     Col *Cols_;                  // array of columns
00252     Row *Rows_;                  // array of rows
00253     bool need_layout_;           // true if layout needs to be calculated
00254
00255 protected:
00256     Fl_Color grid_color;          // color for drawing the grid lines (design helper)
00257     bool draw_grid_;             // draw the grid for testing / design
00258
00259 protected:
00260     void init();
00261     Cell *add_cell(int row, int col);
00262     void remove_cell(int row, int col);
00263
00264 public:
00265     Fl_Grid(int X, int Y, int W, int H, const char *L = 0);
00266     virtual ~Fl_Grid();
00267
00268     // define and manage the layout and resizing
00269
00270     virtual void layout(int rows, int cols, int margin = -1, int gap = -1);
00271     virtual void layout();
00272     virtual void clear_layout();
00273     virtual void resize(int X, int Y, int W, int H) FL_OVERRIDE;
00274
00275     short rows() const { return rows_; }
00276     short cols() const { return cols_; }
00277
00288     void need_layout(int set) {
00289         if (set) {
00290             need_layout_ = true;
00291             redraw();
00292         }
00293         else {
00294             need_layout_ = false;
00295         }
00296     }
00297
00301     bool need_layout() const {
00302         return need_layout_;
00303     }
00304
00305 protected:
00306     virtual void draw() FL_OVERRIDE;
00307     void on_remove(int) FL_OVERRIDE;
00308     virtual void draw_grid();      // draw grid lines for debugging
00309
00310 public:
00311
00312     // get and set individual margins
00313
00314     virtual void margin(int left, int top = -1, int right = -1, int bottom = -1);
00315     int margin(int *left, int *top, int *right, int *bottom) const;
00316
00317     // get and set default row and column gaps for all rows and columns, respectively
00318
00319     virtual void gap(int row_gap, int col_gap = -1); // set default row and column gap(s)
00320     void gap(int *row_gap, int *col_gap) const;
00321
00322     // find cells, get cell pointers
00323
00324     Fl_Grid::Cell* cell(int row, int col) const;
00325     Fl_Grid::Cell* cell(Fl_Widget *widget) const;
00326
00327     // assign a widget to a cell

```

```

00328
00329     Fl_Grid::Cell* widget(Fl_Widget *wi, int row, int col, Fl_Grid_Align align = FL_GRID_FILL);
00330     Fl_Grid::Cell* widget(Fl_Widget *wi, int row, int col, int rowspan, int colspan, Fl_Grid_Align align
= FL_GRID_FILL);
00331
00332     // set minimal column and row sizes (widths and heights, respectively),
00333     // set row and column specific gaps and weights
00334
00335     void col_width(int col, int value);
00336     void col_width(const int *value, size_t size);
00337     int col_width(int col) const;
00338
00339     void col_weight(int col, int value);
00340     void col_weight(const int *value, size_t size);
00341     int col_weight(int col) const;
00342
00343     void col_gap(int col, int value);
00344     void col_gap(const int *value, size_t size);
00345     int col_gap(int col) const;
00346
00347     void row_height(int row, int value);
00348     void row_height(const int *value, size_t size);
00349     int row_height(int row) const;
00350
00351     void row_weight(int row, int value);
00352     void row_weight(const int *value, size_t size);
00353     int row_weight(int row) const;
00354
00355     void row_gap(int row, int value);
00356     void row_gap(const int *value, size_t size);
00357     int row_gap(int row) const;
00358
00359     int computed_col_width(int col) const;
00360     int computed_row_height(int row) const;
00361
00379     void show_grid(int set) {
00380         draw_grid_ = set ? true : false;
00381     }
00382
00396     void show_grid(int set, Fl_Color col) {
00397         draw_grid_ = set ? true : false;
00398         grid_color = col;
00399     }
00400
00401     void debug(int level = 127);
00402
00403 }; // class Fl_Grid
00404
00405 #endif // _FL_FL_GRID_H_

```

12.60 Fl_Group.H File Reference

[Fl_Group](#) and [Fl_End](#) classes.

```

#include "Fl_Widget.H"
#include <vector>

```

Classes

- class [Fl_End](#)

This is a dummy class that allows you to end a [Fl_Group](#) in a constructor list of a class:

- class [Fl_Group](#)

The [Fl_Group](#) class is the main FLTK container widget.

12.60.1 Detailed Description

[Fl_Group](#) and [Fl_End](#) classes.

12.61 Fl_Group.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Group header file for the Fast Light Tool Kit (FLTK).

```

```

00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00021 #ifndef Fl_Group_H
00022 #define Fl_Group_H
00023
00024 #include "Fl_Widget.H"
00025
00026 #include <vector>
00027
00028 // Don't #include Fl_Rect.H because this would introduce lots
00029 // of unnecessary dependencies on Fl_Rect.H
00030 class Fl_Rect;
00031
00032
00033 class FL_EXPORT Fl_Group : public Fl_Widget {
00034
00035     std::vector<Fl_Widget*> child_; // vector of children
00036     Fl_Widget* savedfocus_;
00037     Fl_Widget* resizable_;
00038     Fl_Rect *bounds_; // remembered initial sizes of children
00039     int *sizes_; // remembered initial sizes of children (FLTK 1.3 compat.)
00040
00041     int navigation(int);
00042     static Fl_Group *current_;
00043
00044     // unimplemented copy ctor and assignment operator
00045     Fl_Group(const Fl_Group&);
00046     Fl_Group& operator=(const Fl_Group&);
00047
00048 protected:
00049     void draw() FL_OVERRIDE;
00050     void draw_child(Fl_Widget& widget) const;
00051     void draw_children();
00052     void draw_outside_label(const Fl_Widget& widget) const;
00053     void update_child(Fl_Widget& widget) const;
00054     Fl_Rect *bounds();
00055     int *sizes(); // FLTK 1.3 compatibility
00056     virtual int on_insert(Fl_Widget*, int);
00057     virtual int on_move(int, int);
00058     virtual void on_remove(int);
00059
00060 public:
00061     int handle(int) FL_OVERRIDE;
00062     void begin();
00063     void end();
00064     static Fl_Group *current();
00065     static void current(Fl_Group *g);
00066
00067     int children() const { return (int)child_.size(); }
00068
00069     Fl_Widget *child(int n) const {
00070         if (n < 0 || n > children() - 1) return nullptr;
00071         return child_[n];
00072     }
00073
00074     int find(const Fl_Widget*) const;
00075     int find(const Fl_Widget& o) const { return find(&o); }
00076
00077     Fl_Widget* const* array() const;
00078
00079     void resize(int,int,int,int) FL_OVERRIDE;
00080     Fl_Group(int,int,int,int, const char * = 0);
00081     virtual ~Fl_Group();
00082     void add(Fl_Widget&);
00083     void add(Fl_Widget* o) {add(*o);}
00084     void insert(Fl_Widget&, int i);
00085     void insert(Fl_Widget& o, Fl_Widget* before) {insert(o,find(before));}
00086     void remove(int index);
00087     void remove(Fl_Widget&);
00088     void remove(Fl_Widget* o) {remove(*o);}
00089     void clear();
00090
00091     /* delete child n (by index) */
00092     virtual int delete_child(int n);

```

```

00151
00156 void resizable(Fl_Widget& o) {resizable_ = &o;}
00210 void resizable(Fl_Widget* o) {resizable_ = o;}
00215 Fl_Widget* resizable() const {return resizable_;}
00219 void add_resizable(Fl_Widget& o) {resizable_ = &o; add(o);}
00220 void init_sizes();
00221
00231 void clip_children(int c) { if (c) set_flag(CLIP_CHILDREN); else clear_flag(CLIP_CHILDREN); }
00239 unsigned int clip_children() { return (flags() & CLIP_CHILDREN) != 0; }
00240
00241 // Note: Doxygen docs in Fl_Widget.H to avoid redundancy.
00242 Fl_Group* as_group() FL_OVERRIDE { return this; }
00243 Fl_Group const* as_group() const FL_OVERRIDE { return this; }
00244
00245 // back compatibility functions:
00246
00252 void focus(Fl_Widget* W) {W->take_focus();}
00253
00255 Fl_Widget* & _ddfdesign_kludge() {return resizable_;}
00256
00258 void forms_end();
00259 };
00260
00261 // dummy class used to end child groups in constructors for complex
00262 // subclasses of Fl_Group:
00283 class FL_EXPORT Fl_End {
00284 public:
00286 Fl_End() {Fl_Group::current()->end();}
00287 };
00288
00289 #endif

```

12.62 Fl_Help_Dialog.H

```

00001 //
00002 // Fl_Help_Dialog dialog for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2021 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 // =====
00017 // DO NOT EDIT FL/Fl_Help_Dialog.H and src/Fl_Help_Dialog.cxx !!!
00018 // =====
00019 // Please use fluid to change src/Fl_Help_Dialog.fl interactively
00020 // and then use fluid to "write code" or edit and use fluid -c .
00021 // =====
00022 //
00023
00024 // generated by Fast Light User Interface Designer (fluid) version 1.0500
00025
00026 #ifndef Fl_Help_Dialog_H
00027 #define Fl_Help_Dialog_H
00028 #include <FL/Fl.H>
00029 #include <FL/Fl_Double_Window.H>
00030 #include <FL/Fl_Group.H>
00031 #include <FL/Fl_Button.H>
00032 #include <FL/Fl_Input.H>
00033 #include <FL/Fl_Box.H>
00034 #include <FL/Fl_Help_View.H>
00035
00036 class FL_EXPORT Fl_Help_Dialog {
00037     int index_;
00038     int max_;
00039     int line_[100]; // FIXME: we must remove those static numbers
00040     char file_[100][FL_PATH_MAX]; // FIXME: we must remove those static numbers
00041     int find_pos_;
00042 public:
00043     Fl_Help_Dialog();
00044 private:
00045     Fl_Double_Window *window_;
00046     Fl_Button *back_;
00047     inline void cb_back__i(Fl_Button*, void*);
00048     static void cb_back_(Fl_Button*, void*);
00049     Fl_Button *forward_;
00050     inline void cb_forward__i(Fl_Button*, void*);
00051     static void cb_forward_(Fl_Button*, void*);

```

```

00052     Fl_Button *smaller_;
00053     inline void cb_smaller__i(Fl_Button*, void*);
00054     static void cb_smaller_(Fl_Button*, void*);
00055     Fl_Button *larger_;
00056     inline void cb_larger__i(Fl_Button*, void*);
00057     static void cb_larger_(Fl_Button*, void*);
00058     Fl_Input *find_;
00059     inline void cb_find__i(Fl_Input*, void*);
00060     static void cb_find_(Fl_Input*, void*);
00061     Fl_Help_View *view_;
00062     inline void cb_view__i(Fl_Help_View*, void*);
00063     static void cb_view_(Fl_Help_View*, void*);
00064 public:
00065     ~Fl_Help_Dialog();
00066     int h();
00067     void hide();
00068     int load(const char *f);
00069     void position(int xx, int yy);
00070     void resize(int xx, int yy, int ww, int hh);
00071     void show();
00072     void show(int argc, char **argv);
00073     void textsize(Fl_Fontsize s);
00074     Fl_Fontsize textsize();
00075     void topline(const char *n);
00076     void topline(int n);
00077     void value(const char *f);
00078     const char * value() const;
00079     int visible();
00080     int w();
00081     int x();
00082     int y();
00083 };
00084 #endif

```

12.63 Fl_Help_View.H

```

00001 //
00002 // Help Viewer widget definitions.
00003 //
00004 // Copyright 1997-2010 by Easy Software Products.
00005 // Image support by Matthias Melcher, Copyright 2000-2009.
00006 // Copyright 2011-2024 by Bill Spitzak and others.
00007 //
00008 // This library is free software. Distribution and use rights are outlined in
00009 // the file "COPYING" which should have been included with this file. If this
00010 // file is missing or damaged, see the license at:
00011 //
00012 //     https://www.fltk.org/COPYING.php
00013 //
00014 // Please see the following page on how to report bugs and issues:
00015 //
00016 //     https://www.fltk.org/bugs.php
00017 //
00018 //
00019 /* \file
00020    Fl_Help_View widget . */
00021
00022 #ifndef Fl_Help_View_H
00023 #define Fl_Help_View_H
00024
00025 //
00026 // Include necessary header files...
00027 //
00028 #include "Fl.H"
00029 #include "Fl_Group.H"
00030 #include "Fl_Scrollbar.H"
00031 #include "fl_draw.H"
00032 #include "filename.H"
00033
00034 class Fl_Shared_Image;
00035 //
00036 // Fl_Help_Func type - link callback function for files...
00037 //
00038 typedef const char *(Fl_Help_Func)(Fl_Widget *, const char *);
00039
00040 // Fl_Help_Block structure...
00041 //
00042 struct Fl_Help_Block {
00043     const char *start, // Start of text
00044     const char *end;    // End of text
00045     uchar border;       // Draw border?
00046 };

```

```

00050  Fl_Color      bgcolor;          // Background color
00051  int           x,                // Indentation/starting X coordinate
00052             y,                // Starting Y coordinate
00053             w,                // Width
00054             h;                // Height
00055  int           line[32];        // Left starting position for each line
00056  int           ol;              // is ordered list <OL> element
00057  int           ol_num;          // item number in ordered list
00058 };
00059
00060 //
00061 // Fl_Help_Link structure...
00062 //
00064 struct Fl_Help_Link {
00065     char        filename[192],
00066             name[32];
00067     int         x,
00068             y,
00069             w,
00070             h;
00071 };
00072
00073 /*
00074  * Fl_Help_View font stack opaque implementation
00075  */
00076
00078 struct FL_EXPORT Fl_Help_Font_Style {
00079     Fl_Font      f;
00080     Fl_Fontsize  s;
00081     Fl_Color     c;
00082     void get(Fl_Font &afont, Fl_Fontsize &asize, Fl_Color &acolor) {afont=f; asize=s; acolor=c;}
00083     void set(Fl_Font afont, Fl_Fontsize asize, Fl_Color acolor) {f=afont; s=asize; c=acolor;}
00084     Fl_Help_Font_Style(Fl_Font afont, Fl_Fontsize asize, Fl_Color acolor) {set(afont, asize, acolor);}
00085     Fl_Help_Font_Style(){} // For in table use
00086 };
00087
00089 const size_t MAX_FL_HELP_FSELTS = 100;
00090
00091 struct FL_EXPORT Fl_Help_Font_Stack {
00093     Fl_Help_Font_Stack() {
00094         nfonts_ = 0;
00095     }
00096
00097     void init(Fl_Font f, Fl_Fontsize s, Fl_Color c) {
00098         nfonts_ = 0;
00099         elts_[nfonts_].set(f, s, c);
00100         fl_font(f, s);
00101         fl_color(c);
00102     }
00104     void top(Fl_Font &f, Fl_Fontsize &s, Fl_Color &c) { elts_[nfonts_].get(f, s, c); }
00106     void push(Fl_Font f, Fl_Fontsize s, Fl_Color c) {
00107         if (nfonts_ < MAX_FL_HELP_FSELTS-1) nfonts_ ++;
00108         elts_[nfonts_].set(f, s, c);
00109         fl_font(f, s); fl_color(c);
00110     }
00112     void pop(Fl_Font &f, Fl_Fontsize &s, Fl_Color &c) {
00113         if (nfonts_ > 0) nfonts_ --;
00114         top(f, s, c);
00115         fl_font(f, s); fl_color(c);
00116     }
00118     size_t count() const {return nfonts_;} // Gets the current number of fonts in the stack
00119
00120 protected:
00121     size_t nfonts_;
00122     Fl_Help_Font_Style elts_[MAX_FL_HELP_FSELTS];
00123 };
00124
00127 struct Fl_Help_Target {
00128     char        name[32];
00129     int         y;
00130 };
00131
00201 class FL_EXPORT Fl_Help_View : public Fl_Group {          // Help viewer widget
00202
00203     enum { RIGHT = -1, CENTER, LEFT };
00204
00205     char        title_[1024];
00206     Fl_Color     defcolor_,
00207             bgcolor_,
00208             textcolor_,
00209             linkcolor_;
00210     Fl_Font      textfont_;
00211     Fl_Fontsize  textsize_;
00212     const char   *value_;
00213     Fl_Help_Font_Stack fstack_;
00214     int          nblocks_,
00215             ablocks_;

```



```

00216 Fl_Help_Block *blocks_;
00217
00218 Fl_Help_Func *link_;
00219
00220 int          nlinks_,
00221             alinks_;
00222 Fl_Help_Link *links_;
00223
00224 int          ntargets_,
00225             atargets_;
00226 Fl_Help_Target *targets_;
00227
00228 #if FL_ABI_VERSION >= 10401
00229     char      directory_[2 * FL_PATH_MAX + 15];
00230 #else
00231     char      directory_[FL_PATH_MAX];
00232 #endif
00233
00234     char      filename_[FL_PATH_MAX];
00235     int        topline_,
00236             leftline_,
00237             size_,
00238             hsize_,
00239             scrollbar_size_;
00240 Fl_Scrollbar scrollbar_,
00241             hscrollbar_;
00242
00243     static int selection_first_;
00244     static int selection_last_;
00245     static int selection_push_first_;
00246     static int selection_push_last_;
00247     static int selection_drag_first_;
00248     static int selection_drag_last_;
00249     static int selected_;
00250     static int draw_mode_;
00251     static int mouse_x_;
00252     static int mouse_y_;
00253     static int current_pos_;
00254     static Fl_Help_View *current_view_;
00255     static Fl_Color hv_selection_color_;
00256     static Fl_Color hv_selection_text_color_;
00257
00258
00259 void initfont(Fl_Font &f, Fl_Fontsize &s, Fl_Color &c) { f = textfont_; s = textsize_; c =
    textcolor_; fstack_.init(f, s, c); }
00260 void pushfont(Fl_Font f, Fl_Fontsize s) {fstack_.push(f, s, textcolor_);}
00261 void pushfont(Fl_Font f, Fl_Fontsize s, Fl_Color c) {fstack_.push(f, s, c);}
00262 void popfont(Fl_Font &f, Fl_Fontsize &s, Fl_Color &c) {fstack_.pop(f, s, c);}
00263
00264 Fl_Help_Block *add_block(const char *s, int xx, int yy, int ww, int hh, uchar border = 0);
00265 void          add_link(const char *n, int xx, int yy, int ww, int hh);
00266 void          add_target(const char *n, int yy);
00267 static int    compare_targets(const Fl_Help_Target *t0, const Fl_Help_Target *t1);
00268 int          do_align(Fl_Help_Block *block, int line, int xx, int a, int &l);
00269 protected:
00270 void          draw() FL_OVERRIDE;
00271 private:
00272 void          format();
00273 void          format_table(int *table_width, int *columns, const char *table);
00274 void          free_data();
00275 int          get_align(const char *p, int a);
00276 const char   *get_attr(const char *p, const char *n, char *buf, int bufsize);
00277 Fl_Color      get_color(const char *n, Fl_Color c);
00278 Fl_Shared_Image *get_image(const char *name, int W, int H);
00279 int          get_length(const char *l);
00280 public:
00281 int          handle(int) FL_OVERRIDE;
00282 private:
00283
00284 void          hv_draw(const char *t, int x, int y, int entity_extra_length = 0);
00285 char          begin_selection();
00286 char          extend_selection();
00287 void          end_selection(int c=0);
00288 void          clear_global_selection();
00289 Fl_Help_Link *find_link(int, int);
00290 void          follow_link(Fl_Help_Link*);
00291
00292 public:
00293
00294     static const char *copy_menu_text;
00295
00296 Fl_Help_View(int xx, int yy, int ww, int hh, const char *l = 0);
00297 ~Fl_Help_View();
00299 const char   *directory() const { if (directory_[0]) return (directory_);
00300                                     else return ((const char *)0); }
00302 const char   *filename() const { if (filename_[0]) return (filename_);
00303                                     else return ((const char *)0); }

```

```

00304     int             find(const char *s, int p = 0);
00327     void            link(Fl_Help_Func *fn) { link_ = fn; }
00328     int             load(const char *f);
00329     void            resize(int,int,int,int) FL_OVERRIDE;
00331     int             size() const { return (size_); }
00332     void            size(int W, int H) { FL_Widget::size(W, H); }
00334     void            textcolor(Fl_Color c) { if (textcolor_ == defcolor_) textcolor_ = c; defcolor_ = c; }
00336     Fl_Color        textcolor() const { return (defcolor_); }
00338     void            textfont(Fl_Font f) { textfont_ = f; format(); }
00340     Fl_Font         textfont() const { return (textfont_); }
00342     void            textsize(Fl_Fontsize s) { textsize_ = s; format(); }
00344     Fl_Fontsize     textsize() const { return (textsize_); }
00346     const char      *title() { return (title_); }
00347     void            topline(const char *n);
00348     void            topline(int);
00350     int             topline() const { return (topline_); }
00351     void            leftline(int);
00353     int             leftline() const { return (leftline_); }
00354     void            value(const char *val);
00356     const char      *value() const { return (value_); }
00357     void            clear_selection();
00358     void            select_all();
00368     int scrollbar_size() const {
00369         return(scrollbar_size_);
00370     }
00390     void scrollbar_size(int newSize) {
00391         scrollbar_size_ = newSize;
00392     }
00393
00394     // Check if the user selected text in this view.
00395     int text_selected();
00396
00397     // If text is selected in this view, copy it to a clipboard.
00398     int copy(int clipboard=1);
00399 };
00400
00401 #endif // !Fl_Help_View_H

```

12.64 Fl_Hold_Browser.H

```

00001 //
00002 // Hold browser header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Hold_Browser widget . */
00019
00020 #ifndef Fl_Hold_Browser_H
00021 #define Fl_Hold_Browser_H
00022
00023 #include "Fl_Browser.H"
00024
00038 class FL_EXPORT Fl_Hold_Browser : public Fl_Browser {
00039 public:
00046     Fl_Hold_Browser(int X,int Y,int W,int H,const char *L=0);
00047 };
00048
00049 #endif

```

12.65 Fl_Hor_Fill_Slider.H

```

00001 //
00002 // Horizontal fill slider header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //

```

```

00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Hor_Fill_Slider widget . */
00019
00020 #ifndef Fl_Hor_Fill_Slider_H
00021 #define Fl_Hor_Fill_Slider_H
00022
00023 #include "Fl_Slider.H"
00024
00025 class FL_EXPORT Fl_Hor_Fill_Slider : public Fl_Slider {
00026 public:
00027     Fl_Hor_Fill_Slider(int X,int Y,int W,int H,const char *L=0);
00028 };
00029
00030 #endif

```

12.66 Fl_Hor_Nice_Slider.H

```

00001 //
00002 // Horizontal "nice" slider header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Hor_Nice_Slider widget . */
00019
00020 #ifndef Fl_Hor_Nice_Slider_H
00021 #define Fl_Hor_Nice_Slider_H
00022
00023 #include "Fl_Slider.H"
00024
00025 class FL_EXPORT Fl_Hor_Nice_Slider : public Fl_Slider {
00026 public:
00027     Fl_Hor_Nice_Slider(int X,int Y,int W,int H,const char *L=0);
00028 };
00029
00030 #endif

```

12.67 Fl_Hor_Slider.H

```

00001 //
00002 // Horizontal slider header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2011 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Hor_Slider widget . */
00019
00020 #ifndef Fl_Hor_Slider_H
00021 #define Fl_Hor_Slider_H
00022
00023 #include "Fl_Slider.H"
00024
00025 class FL_EXPORT Fl_Hor_Slider : public Fl_Slider {

```

```

00030 public:
00031
00036     Fl_Hor_Slider(int X,int Y,int W,int H,const char *l=0);
00037 };
00038
00039 #endif

```

12.68 Fl_Hor_Value_Slider.H

```

00001 //
00002 // Horizontal value slider header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Hor_Value_Slider widget . */
00019
00020 #ifndef Fl_Hor_Value_Slider_H
00021 #define Fl_Hor_Value_Slider_H
00022
00023 #include "Fl_Value_Slider.H"
00024
00025 class FL_EXPORT Fl_Hor_Value_Slider : public Fl_Value_Slider {
00026 public:
00027     Fl_Hor_Value_Slider(int X,int Y,int W,int H,const char *l=0);
00028 };
00029
00030 #endif

```

12.69 Fl_ICO_Image.H

```

00001 //
00002 // ICO image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2022-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     http://www.fltk.org/COPYING.php
00011 //
00012 // Please report all bugs and problems on the following page:
00013 //
00014 //     http://www.fltk.org/str.php
00015 //
00016
00017 // https://en.wikipedia.org/wiki/ICO_(file_format)
00018 // http://www.daubnet.com/en/file-format-ico
00019
00020 #ifndef Fl_ICO_Image_H
00021 # define Fl_ICO_Image_H
00022 # include "Fl_BMP_Image.H"
00023
00027 class FL_EXPORT Fl_ICO_Image : public Fl_BMP_Image {
00028
00029 public:
00031     struct IconDirEntry {
00032         int bWidth;
00033         int bHeight;
00034         int bColorCount;
00035         int bReserved;
00036         int wPlanes;
00037         int wBitCount;
00038         int dwBytesInRes;
00039         int dwImageOffset;
00040     };
00041
00042     Fl_ICO_Image(const char *filename, int id = -1, const unsigned char *data = NULL, const size_t
datasize = 0);
00043     ~Fl_ICO_Image();

```

```

00044
00046     int idcount() const { return idcount_; }
00047
00049     const IconDirEntry * icondirentry() const { return icondirentry_; }
00050
00051 private:
00052     int idcount_;
00053     struct IconDirEntry *icondirentry_;
00054     void load_ico_(class Fl_Image_Reader &rdr, int id);
00055 };
00056
00057 #endif // Fl_ICO_Image_H

```

12.70 Fl_Image.H File Reference

[Fl_Image](#), [Fl_RGB_Image](#) classes.

```
#include "Enumerations.H"
```

```
#include "Fl_Widget.H"
```

Classes

- class [Fl_Image](#)
Base class for image caching, scaling and drawing.
- class [Fl_RGB_Image](#)
The [Fl_RGB_Image](#) class supports caching and drawing of full-color images with 1 to 4 channels of color information.

Enumerations

- enum [Fl_RGB_Scaling](#) { [FL_RGB_SCALING_NEAREST](#) = 0 , [FL_RGB_SCALING_BILINEAR](#) }
The scaling algorithm to use for RGB images.

12.70.1 Detailed Description

[Fl_Image](#), [Fl_RGB_Image](#) classes.

12.70.2 Enumeration Type Documentation

12.70.2.1 Fl_RGB_Scaling

```
enum Fl\_RGB\_Scaling
```

The scaling algorithm to use for RGB images.

Enumerator

FL_RGB_SCALING_NEAREST	default RGB image scaling algorithm
FL_RGB_SCALING_BILINEAR	more accurate, but slower RGB image scaling algorithm

12.71 Fl_Image.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:

```

```

00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00020 #ifndef Fl_Image_H
00021 #define Fl_Image_H
00022
00023 #include "Enumerations.H"
00024 #include "Fl_Widget.H" // for fl_uintptr_t
00025
00026 class Fl_Widget;
00027 class Fl_Pixmap;
00028 struct Fl_Menu_Item;
00029 struct Fl_Label;
00030 class Fl_RGB_Image;
00031
00032
00036 enum Fl_RGB_Scaling {
00037     FL_RGB_SCALING_NEAREST = 0,
00038     FL_RGB_SCALING_BILINEAR
00039 };
00040
00041
00060 class FL_EXPORT Fl_Image {
00061     friend class Fl_Graphics_Driver;
00062 public:
00063     static const int ERR_NO_IMAGE      = -1;
00064     static const int ERR_FILE_ACCESS   = -2;
00065     static const int ERR_FORMAT        = -3;
00066     static const int ERR_MEMORY_ACCESS = -4;
00067
00068 private:
00069     int w_, h_, d_, ld_, count_;
00070     int data_w_, data_h_;
00071     const char * const *data_;
00072     static Fl_RGB_Scaling RGB_scaling_; // method used when copying RGB images
00073     static Fl_RGB_Scaling scaling_algorithm_; // method used to rescale RGB source images before drawing
00074     // Forbid use of copy constructor and assign operator
00075     Fl_Image & operator=(const Fl_Image &);
00076     Fl_Image(const Fl_Image &);
00077     // Presently redefined in Fl_SVG_Image
00078     virtual void cache_size_(int & /*width*/, int & /*height*/) {}
00079
00080 protected:
00081
00087     void w(int W) {w_ = W; data_w_ = W;}
00093     void h(int H) {h_ = H; data_h_ = H;}
00097     void d(int D) {d_ = D;}
00109     void ld(int LD) {ld_ = LD;}
00117     void data(const char * const *p, int c) {data_ = p; count_ = c;}
00118     void draw_empty(int X, int Y);
00119
00120     static void labeltype(const Fl_Label *lo, int lx, int ly, int lw, int lh, Fl_Align la);
00121     static void measure(const Fl_Label *lo, int &lw, int &lh);
00122     int draw_scaled(int X, int Y, int W, int H);
00123
00124 public:
00125
00131     int w() const {return w_;}
00137     int h() const {return h_;}
00141     int data_w() const {return data_w_;}
00145     int data_h() const {return data_h_;}
00151     int d() const {return d_;}
00156     int ld() const {return ld_;}
00165     int count() const {return count_;}
00191     const char * const *data() const {return data_;}
00192     int fail() const;
00220     virtual void release() {
00221         delete this;
00222     }
00223
00233     virtual class Fl_Shared_Image *as_shared_image() {
00234         return 0;
00235     }
00236
00237     Fl_Image(int W, int H, int D);
00238     virtual ~Fl_Image();
00239     virtual Fl_Image *copy(int W, int H) const;
00266     Fl_Image *copy() const { return copy(w(), h()); }
00267     virtual void color_average(Fl_Color c, float i);
00279     void inactive() { color_average(FL_GRAY, .33f); }
00280     virtual void desaturate();
00281     virtual void label(Fl_Widget*w);
00282     virtual void label(Fl_Menu_Item*m);
00294     virtual void draw(int X, int Y, int W, int H, int cx=0, int cy=0); // platform dependent
00299     void draw(int X, int Y) {draw(X, Y, w(), h(), 0, 0);} // platform dependent
00300     virtual void uncache();

```

```

00301
00302 // used by fl_define_FL_IMAGE_LABEL() to avoid 'friend' declaration
00303 static Fl_Labeltype define_FL_IMAGE_LABEL();
00304
00305 // set RGB image scaling method
00306 static void RGB_scaling(Fl_RGB_Scaling);
00307 // get RGB image scaling method
00308 static Fl_RGB_Scaling RGB_scaling();
00309
00310 // set the image drawing size
00311 virtual void scale(int width, int height, int proportional = 1, int can_expand = 0);
00312 static void scaling_algorithm(Fl_RGB_Scaling algorithm) {scaling_algorithm_ = algorithm; }
00322 static Fl_RGB_Scaling scaling_algorithm() {return scaling_algorithm_;}
00323 static bool register_images_done;
00324 };
00325
00326 class Fl_SVG_Image;
00327
00339 class FL_EXPORT Fl_RGB_Image : public Fl_Image {
00340     friend class Fl_Graphics_Driver;
00341     static size_t max_size_;
00342 public:
00343
00347     const uchar *array;
00350     int alloc_array;
00351
00352 private:
00353     // These two variables are used to cache the image and mask for the main display graphics driver
00354     fl_uintptr_t id_;
00355     fl_uintptr_t mask_;
00356     int cache_w_, cache_h_; // size of image when cached
00357 public:
00358
00359     Fl_RGB_Image(const uchar *bits, int W, int H, int D=3, int LD=0);
00360     Fl_RGB_Image(const uchar *bits, int bits_length, int W, int H, int D, int LD);
00361     Fl_RGB_Image(const Fl_Pixmap *pixmap, Fl_Color bg=FL_GRAY);
00362     ~Fl_RGB_Image() FL_OVERRIDE;
00363     Fl_Image *copy(int W, int H) const FL_OVERRIDE;
00364     Fl_Image *copy() const { return Fl_Image::copy(); }
00365     void color_average(Fl_Color c, float i) FL_OVERRIDE;
00366     void desaturate() FL_OVERRIDE;
00367     void draw(int X, int Y, int W, int H, int cx=0, int cy=0) FL_OVERRIDE;
00368     void draw(int X, int Y) {draw(X, Y, w(), h(), 0, 0);}
00369     void label(Fl_Widget*w) FL_OVERRIDE;
00370     void label(Fl_Menu_Item*m) FL_OVERRIDE;
00371     void uncache() FL_OVERRIDE;
00372     int cache_w() {return cache_w_;}
00373     int cache_h() {return cache_h_;}
00383     static void max_size(size_t size) { max_size_ = size;}
00388     static size_t max_size() {return max_size_;}
00391     virtual Fl_SVG_Image *as_svg_image() { return NULL; }
00394     virtual void normalize() {}
00395 };
00396
00397 #endif // !Fl_Image_H

```

12.72 Fl_Image_Surface.H

```

00001 //
00002 // Draw-to-image code for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2016 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Image_Surface_H
00018 #define Fl_Image_Surface_H
00019
00020 #include <FL/Fl_Widget_Surface.H>
00021 #include <FL/Fl_Image.H>
00022 #include <FL/Fl_Shared_Image.H>
00023 #include <FL/platform_types.h> // for Fl_Offscreen
00024
00025
00065 class FL_EXPORT Fl_Image_Surface : public Fl_Widget_Surface {
00066     friend class Fl_Graphics_Driver;

```

```

00067 private:
00068     class Fl_Image_Surface_Driver *platform_surface;
00069     Fl_Offscreen get_offscreen_before_delete();
00070 protected:
00071     void translate(int x, int y) FL_OVERRIDE;
00072     void untranslate() FL_OVERRIDE;
00073 public:
00074     Fl_Image_Surface(int w, int h, int high_res = 0, Fl_Offscreen off = 0);
00075     ~Fl_Image_Surface();
00076     void set_current() FL_OVERRIDE;
00077     bool is_current() FL_OVERRIDE;
00078     Fl_RGB_Image *image();
00079     Fl_Shared_Image *highres_image();
00080     void origin(int *x, int *y) FL_OVERRIDE;
00081     void origin(int x, int y) FL_OVERRIDE;
00082     int printable_rect(int *w, int *h) FL_OVERRIDE;
00083     Fl_Offscreen offscreen();
00084     void rescale();
00085     void mask(const Fl_RGB_Image *);
00086 };
00087
00088
00100 class Fl_Image_Surface_Driver : public Fl_Widget_Surface {
00101     friend class Fl_Image_Surface;
00102 private:
00103     Fl_Image_Surface *image_surface_;
00104 protected:
00105     int width;
00106     int height;
00107     Fl_Offscreen offscreen;
00108     int external_offscreen;
00109     Fl_Image_Surface_Driver(int w, int h, int /*high_res*/, Fl_Offscreen off) : Fl_Widget_Surface(NULL),
width(w), height(h), offscreen(off) {external_offscreen = (off != 0);}
00110     virtual ~Fl_Image_Surface_Driver() {}
00111     static void copy_with_mask(Fl_RGB_Image* mask, uchar *dib_dst, uchar *dib_src,
                                int line_size, bool bottom_to_top);
00112     static Fl_RGB_Image *RGB3_to_RGB1(const Fl_RGB_Image *rgb3, int W, int H);
00113     void set_current() FL_OVERRIDE = 0;
00114     void translate(int x, int y) FL_OVERRIDE = 0;
00115     void untranslate() FL_OVERRIDE = 0;
00116     int printable_rect(int *w, int *h) FL_OVERRIDE;
00117     virtual Fl_RGB_Image *image() = 0;
00118     virtual void mask(const Fl_RGB_Image *) {}
00119     static Fl_Image_Surface_Driver *newImageSurfaceDriver(int w, int h, int high_res, Fl_Offscreen off);
00120 public:
00121     Fl_Image_Surface *image_surface() { return image_surface_; }
00122 };
00123
00135 #endif // Fl_Image_Surface_H

```

12.73 Fl_Input.H

```

00001 //
00002 // Input header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00017 /* \file
00018     Fl_Input widget . */
00019
00020 #ifndef Fl_Input_H
00021 #define Fl_Input_H
00022
00023 #include "Fl_Input_.H"
00024
00025 class FL_EXPORT Fl_Input : public Fl_Input_ {
00026     friend class Fl_Screen_Driver;
00027     friend class Fl_Cocoa_Screen_Driver; // Not ideal, but probably no other platform will use it
00028     int shift_position(int p);
00029     int shift_up_down_position(int p);
00030     void handle_mouse(int keepmark=0);
00031
00032     // Private keyboard functions
00033     int kf_lines_up(int repeat_num);

```



```

00229     int kf_lines_down(int repeat_num);
00230     int kf_page_up();
00231     int kf_page_down();
00232     int kf_insert_toggle();
00233     int kf_delete_word_right();
00234     int kf_delete_word_left();
00235     int kf_delete_sol();
00236     int kf_delete_eol();
00237     int kf_delete_char_right();
00238     int kf_delete_char_left();
00239     int kf_move_sol();
00240     int kf_move_eol();
00241     int kf_clear_eol();
00242     int kf_move_char_left();
00243     int kf_move_char_right();
00244     int kf_move_word_left();
00245     int kf_move_word_right();
00246     int kf_move_up_and_sol();
00247     int kf_move_down_and_eol();
00248     int kf_top();
00249     int kf_bottom();
00250     int kf_select_all();
00251     int kf_undo();
00252     int kf_redo();
00253     int kf_copy();
00254     int kf_paste();
00255     int kf_copy_cut();
00256
00257 protected:
00258     void draw() FL_OVERRIDE;
00259     int handle_key();
00260     int handle_rmb();
00261
00262 public:
00263     int handle(int) FL_OVERRIDE;
00264     Fl_Input(int,int,int,int,const char * = 0);
00265     static const char *cut_menu_text;
00266     static const char *copy_menu_text;
00267     static const char *paste_menu_text;
00268 };
00269
00270 #endif

```

12.74 Fl_Input.H

```

00001 //
00002 // Input base class header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2015 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Input_ widget . */
00019
00020 #ifndef Fl_Input__H
00021 #define Fl_Input__H
00022
00023 #ifndef Fl_Widget_H
00024 #include "Fl_Widget.H"
00025 #endif
00026
00027 #define FL_NORMAL_INPUT          0
00028 #define FL_FLOAT_INPUT           1
00029 #define FL_INT_INPUT             2
00030 #define FL_HIDDEN_INPUT          3
00031 #define FL_MULTILINE_INPUT       4
00032 #define FL_SECRET_INPUT          5
00033 #define FL_INPUT_TYPE            7
00034 #define FL_INPUT_READONLY        8
00035 #define FL_NORMAL_OUTPUT         (FL_NORMAL_INPUT | FL_INPUT_READONLY)
00036 #define FL_MULTILINE_OUTPUT      (FL_MULTILINE_INPUT | FL_INPUT_READONLY)
00037 #define FL_INPUT_WRAP            16
00038 #define FL_MULTILINE_INPUT_WRAP (FL_MULTILINE_INPUT | FL_INPUT_WRAP)
00039 #define FL_MULTILINE_OUTPUT_WRAP (FL_MULTILINE_INPUT | FL_INPUT_READONLY | FL_INPUT_WRAP)
00040

```

```

00041 class Fl_Input_Undo_Action;
00042 class Fl_Input_Undo_Action_List;
00043
00096 class FL_EXPORT Fl_Input_ : public Fl_Widget {
00097
00099     const char* value_;
00100
00102     char* buffer;
00103
00105     int size_;
00106
00108     int bufsize;
00109
00111     int position_;
00112
00115     int mark_;
00116
00120     int tab_nav_;
00121
00123     int xscroll_, yscroll_;
00124
00127     int mu_p;
00128
00130     int maximum_size_;
00131
00133     int shortcut_;
00134
00136     uchar erase_cursor_only;
00137
00139     Fl_Font textfont_;
00140
00142     Fl_Fonsize textsize_;
00143
00145     Fl_Color textcolor_;
00146
00148     Fl_Color cursor_color_;
00149
00151     Fl_Input_Undo_Action* undo_;
00152     Fl_Input_Undo_Action_List* undo_list_;
00153     Fl_Input_Undo_Action_List* redo_list_;
00154
00156     static double up_down_pos;
00157
00159     static int was_up_down;
00160
00161     /* Convert a given text segment into the text that will be rendered on screen. */
00162     const char* expand(const char*, char*) const;
00163
00164     /* Calculates the width in pixels of part of a text buffer. */
00165     double expandpos(const char*, const char*, const char*, int*) const;
00166
00167     /* Mark a range of characters for update. */
00168     void minimal_update(int, int);
00169
00170     /* Mark a range of characters for update. */
00171     void minimal_update(int p);
00172
00173     /* Copy the value from a possibly static entry into the internal buffer. */
00174     void put_in_buffer(int newsz);
00175
00176     /* Set the current font and font size. */
00177     void setfont() const;
00178
00179 protected:
00180
00181     /* Find the start of a word. */
00182     int word_start(int i) const;
00183
00184     /* Find the end of a word. */
00185     int word_end(int i) const;
00186
00187     /* Find the start of a line. */
00188     int line_start(int i) const;
00189
00190     /* Find the end of a line. */
00191     int line_end(int i) const;
00192
00193     /* Draw the text in the passed bounding box. */
00194     void drawtext(int, int, int, int);
00195
00196     /* Draw the text in the passed bounding box. */
00197     void drawtext(int, int, int, int, bool draw_active);
00198
00199     /* Move the cursor to the column given by up_down_pos. */
00200     int up_down_position(int, int keepmark=0);
00201
00202     /* Handle mouse clicks and mouse moves. */

```

```

00203 void handle_mouse(int, int, int, int, int keepmark=0);
00204
00205 /* Handle all kinds of text field related events. */
00206 int handle_text(int e, int, int, int, int);
00207
00208 /* Check the when() field and do a callback if indicated. */
00209 void maybe_do_callback(Fl_Callback_Reason reason = FL_REASON_UNKNOWN);
00210
00212 int xscroll() const {return xscroll_;}
00213
00215 int yscroll() const {return yscroll_;}
00216 void yscroll(int yOffset) { yscroll_ = yOffset; damage(FL_DAMAGE_EXPOSE);}
00217
00218 /* Return the number of lines displayed on a single page. */
00219 int linesPerPage();
00220
00221 /* Apply the current undo/redo operation, called from undo() or redo() */
00222 int apply_undo();
00223
00224 public:
00225
00226 /* Change the size of the widget. */
00227 void resize(int, int, int, int) FL_OVERRIDE;
00228
00229 /* Constructor */
00230 Fl_Input_(int, int, int, int, const char* = 0);
00231
00232 /* Destructor */
00233 ~Fl_Input_();
00234
00235 /* Changes the widget text. */
00236 int value(const char*);
00237
00238 /* Changes the widget text. */
00239 int value(const char*, int);
00240
00241 /* Changes the widget text. */
00242 int value(int value);
00243
00244 /* Changes the widget text. */
00245 int value(double value);
00246
00247 /* Changes the widget text. */
00248 int static_value(const char*);
00249
00250 /* Changes the widget text. */
00251 int static_value(const char*, int);
00252
00263 const char* value() const {return value_;}
00264
00265 int ivalue() const;
00266
00267 double dvalue() const;
00268
00269 /* Returns the Unicode character at index \p i. */
00270 unsigned int index(int i) const;
00271
00280 int size() const {return size_;}
00281
00285 void size(int W, int H) { Fl_Widget::size(W, H); }
00286
00289 int maximum_size() const {return maximum_size_;}
00290
00300 void maximum_size(int m) {maximum_size_ = m;}
00301
00306 int insert_position() const { return position_; }
00307 FL_DEPRECATED("since 1.4.0 - use insert_position() instead",
00308 int position() const ) { return insert_position(); }
00309
00312 int mark() const {return mark_;}
00313
00314 /* Sets the index for the cursor and mark. */
00315 int insert_position(int p, int m);
00316 FL_DEPRECATED("since 1.4.0 - use insert_position(p, m) or Fl_Widget::position(x, y) instead",
00317 int position(int p, int m) ) { return insert_position(p, m); }
00318
00325 int insert_position(int p) { return insert_position(p, p); }
00326 FL_DEPRECATED("since 1.4.0 - use insert_position(p) instead",
00327 int position(int p) ) { return insert_position(p); }
00328
00334 int mark(int m) {return insert_position(insert_position(), m);}
00335
00336 /* Deletes text from \p b to \p e and inserts the new string \p text. */
00337 int replace(int b, int e, const char *text, int len=0);
00338
00349 int cut() {return replace(insert_position(), mark(), 0);}
00350

```

```

00363 int cut(int n) {return replace(insert_position(), insert_position()+n, 0);}
00364
00376 int cut(int a, int b) {return replace(a, b, 0);}
00377
00389 int insert(const char* t, int l=0){return replace(position_, mark_, t, 1);}
00390
00391 /* Append text at the end. */
00392 int append(const char* t, int l=0, char keep_selection=0);
00393
00394 /* Put the current selection into the clipboard. */
00395 int copy(int clipboard);
00396
00397 /* Undo previous changes to the text buffer. */
00398 int undo();
00399
00400 /* Return true if the last operation can be undone. */
00401 bool can_undo() const;
00402
00403 /* Redo previous undo operations. */
00404 int redo();
00405
00406 /* Return true if there is a redo action in the list. */
00407 bool can_redo() const;
00408
00409 /* Copy the yank buffer to the clipboard. */
00410 int copy_cuts();
00411
00415 int shortcut() const {return shortcut_;}
00416
00423 void shortcut(int s) {shortcut_ = s;}
00424
00427 Fl_Font textfont() const {return textfont_;}
00428
00432 void textfont(Fl_Font s) {textfont_ = s;}
00433
00436 Fl_Fonsize textsize() const {return textsize_;}
00437
00441 void textsize(Fl_Fonsize s) {textsize_ = s;}
00442
00446 Fl_Color textcolor() const {return textcolor_;}
00447
00452 void textcolor(Fl_Color n) {textcolor_ = n;}
00453
00456 Fl_Color cursor_color() const {return cursor_color_;}
00457
00461 void cursor_color(Fl_Color n) {cursor_color_ = n;}
00462
00465 int input_type() const {return type() & FL_INPUT_TYPE; }
00466
00470 void input_type(int t) { type((uchar)(t | readonly())); }
00471
00474 int readonly() const { return type() & FL_INPUT_READONLY; }
00475
00478 void readonly(int b) { if (b) type((uchar)(type() | FL_INPUT_READONLY));
00479                      else type((uchar)(type() & ~FL_INPUT_READONLY)); }
00480
00485 int wrap() const { return type() & FL_INPUT_WRAP; }
00486
00491 void wrap(int b) { if (b) type((uchar)(type() | FL_INPUT_WRAP));
00492                  else type((uchar)(type() & ~FL_INPUT_WRAP)); }
00493
00517 void tab_nav(int val) {
00518     tab_nav_ = val;
00519 }
00520
00531 int tab_nav() const {
00532     return tab_nav_;
00533 }
00534 };
00535
00536 #endif

```

12.75 Fl_Input_Choice.H

```

00001 //
00002 // An input/chooser widget.
00003 //
00004 //      |-----| | |
00005 //      | input area | | \ |
00006 //      |-----| | |
00007 //
00008 // Copyright 2004 by Greg Ercolano.
00009 // Copyright 1998-2024 by Bill Spitzak and others.
00010 //
00011 // This library is free software. Distribution and use rights are outlined in

```

```

00012 // the file "COPYING" which should have been included with this file.  If this
00013 // file is missing or damaged, see the license at:
00014 //
00015 //     https://www.fltk.org/COPYING.php
00016 //
00017 // Please see the following page on how to report bugs and issues:
00018 //
00019 //     https://www.fltk.org/bugs.php
00020 //
00021
00022 /* \file
00023     Fl_Input_Choice widget . */
00024
00025 #ifndef Fl_Input_Choice_H
00026 #define Fl_Input_Choice_H
00027
00028 #include <FL/Fl.H>
00029 #include <FL/Fl_Group.H>
00030 #include <FL/Fl_Input.H>
00031 #include <FL/Fl_Menu_Button.H>
00032
00033 /*
00034     A combination of the input widget and a menu button.
00035
00036     The user can either type into the input area, or use the
00037     menu button chooser on the right to choose an item which loads
00038     the input area with the selected text.
00039
00040     Note: doxygen docs in src/Fl_Input_Choice.cxx
00041 */
00042
00043 class FL_EXPORT Fl_Input_Choice : public Fl_Group {
00044
00045     // Private class to handle slightly 'special' behavior of menu button
00046     class InputMenuButton : public Fl_Menu_Button {
00047     public:
00048         void draw() FL_OVERRIDE;
00049         const Fl_Menu_Item* popup();
00050     public:
00051         InputMenuButton(int X, int Y, int W, int H, const char *L=0);
00052         int handle(int e) FL_OVERRIDE;
00053     };
00054
00055     Fl_Input *inp_;
00056     InputMenuButton *menu_;
00057
00058     // note: this is used by the Fl_Input_Choice ctor.
00059     static void menu_cb(Fl_Widget*, void *data);
00060
00061     // note: this is used by the Fl_Input_Choice ctor.
00062     static void inp_cb(Fl_Widget*, void *data);
00063 protected:
00064     // Custom resize behavior -- input stretches, menu button doesn't
00065
00066     virtual int inp_x() const { return(x() + Fl::box_dx(box())); }
00067     virtual int inp_y() const { return(y() + Fl::box_dy(box())); }
00068     virtual int inp_w() const { return(w() - Fl::box_dw(box()) - menu_w()); }
00069     virtual int inp_h() const { return(h() - Fl::box_dh(box())); }
00070
00071     virtual int menu_x() const { return x() + w() - menu_w() - Fl::box_dx(box()); }
00072     virtual int menu_y() const { return y() + Fl::box_dy(box()); }
00073     virtual int menu_w() const { return 20; }
00074     virtual int menu_h() const { return h() - Fl::box_dh(box()); }
00075
00076     void draw() FL_OVERRIDE;
00077
00078 public:
00079     Fl_Input_Choice(int X, int Y, int W, int H, const char *L=0);
00080
00081     void resize(int X, int Y, int W, int H) FL_OVERRIDE;
00082
00083     void add(const char *s) { menu_->add(s); }
00084
00085     int changed() const { return inp_->changed() | Fl_Widget::changed(); }
00086
00087     // Clears the changed() state of both input and menu button widgets.
00088     void clear_changed();
00089
00090     // Sets the changed() state of both input and menu button widgets.
00091     void set_changed();
00092
00093     void clear() { menu_->clear(); }
00094
00095     Fl_Boxtype down_box() const { return (menu_->down_box()); }
00096
00097     void down_box(Fl_Boxtype b) { menu_->down_box(b); }

```

```

00133
00135     const Fl_Menu_Item *menu() { return (menu_>menu()); }
00136
00138     void menu(const Fl_Menu_Item *m) { menu_>menu(m); }
00139
00141     Fl_Color textcolor() const { return (inp_>textcolor()); }
00142
00144     void textcolor(Fl_Color c) { inp_>textcolor(c); }
00145
00147     Fl_Font textfont() const { return (inp_>textfont()); }
00148
00150     void textfont(Fl_Font f) { inp_>textfont(f); }
00151
00153     Fl_Fonsize textsize() const { return (inp_>textsize()); }
00154
00156     void textsize(Fl_Fonsize s) { inp_>textsize(s); }
00157
00159     const char* value() const { return (inp_>value()); }
00160
00171     void value(const char *val) { inp_>value(val); }
00172
00173     /* Chooses item# \p val in the menu, and sets the Fl_Input text field
00174        to that value. Any previous text is cleared. */
00175     void value(int val);
00176
00177     int update_menubutton();
00178
00191     Fl_Menu_Button *menubutton() { return menu_; }
00192
00196     Fl_Input *input() { return inp_; }
00197 };
00198
00199 #endif // !Fl_Input_Choice_H

```

12.76 Fl_Int_Input.H

```

00001 //
00002 // Integer input header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Int_Input widget . */
00019
00020 #ifndef Fl_Int_Input_H
00021 #define Fl_Int_Input_H
00022
00023 #include "Fl_Input.H"
00024
00029 class FL_EXPORT Fl_Int_Input : public Fl_Input {
00030 public:
00037     Fl_Int_Input(int X,int Y,int W,int H,const char *l = 0);
00038 };
00039
00040 #endif

```

12.77 Fl_JPEG_Image.H

```

00001 //
00002 // JPEG image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //

```

```

00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_JPEG_Image class . */
00019
00020 #ifndef Fl_JPEG_Image_H
00021 #define Fl_JPEG_Image_H
00022 # include "Fl_Image.H"
00023
00030 class FL_EXPORT Fl_JPEG_Image : public Fl_RGB_Image {
00031
00032 public:
00033
00034     Fl_JPEG_Image(const char *filename);
00035     Fl_JPEG_Image(const char *name, const unsigned char *data, int data_length=-1);
00036
00037 protected:
00038
00039     void load_jpg(const char *filename, const char *sharename, const unsigned char *data, int
data_length=-1);
00040
00041 };
00042
00043 #endif

```

12.78 Fl_Light_Button.H

```

00001 //
00002 // Lighted button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Light_Button widget . */
00019
00020 #ifndef Fl_Light_Button_H
00021 #define Fl_Light_Button_H
00022
00023 #include "Fl_Button.H"
00024
00036 class FL_EXPORT Fl_Light_Button : public Fl_Button {
00037 protected:
00038     void draw() FL_OVERRIDE;
00039 public:
00040     int handle(int) FL_OVERRIDE;
00041     Fl_Light_Button(int x,int y,int w,int h,const char *l = 0);
00042 };
00043
00044 #endif

```

12.79 Fl_Line_Dial.H

```

00001 //
00002 // Line dial header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Line_Dial widget . */

```

```

00019
00020 #ifndef Fl_Line_Dial_H
00021 #define Fl_Line_Dial_H
00022
00023 #include "Fl_Dial.H"
00024
00025 class FL_EXPORT Fl_Line_Dial : public Fl_Dial {
00026 public:
00027     Fl_Line_Dial(int X,int Y,int W,int H, const char *L = 0);
00028 };
00029
00030 #endif

```

12.80 Fl_Menu.H

```

00001 //
00002 // Old menu header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 // this include file is for back compatibility only
00018 #include "Fl_Menu_Item.H"

```

12.81 Fl_Menu_.H

```

00001 //
00002 // Menu base class header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Menu_ widget . */
00019
00020 #ifndef Fl_Menu__H
00021 #define Fl_Menu__H
00022
00023 #ifndef Fl_Widget_H
00024 #include "Fl_Widget.H"
00025 #endif
00026 #include "Fl_Menu_Item.H"
00027
00028 class FL_EXPORT Fl_Menu_ : public Fl_Widget {
00029 public:
00030     Fl_Menu_Item *menu_;
00031     const Fl_Menu_Item *value_;
00032     const Fl_Menu_Item *prev_value_;
00033 protected:
00034     uchar alloc; // flag indicates if menu_ is a dynamic copy (=1) or not (=0)
00035     uchar down_box_;
00036     Fl_Boxtype menu_box_;
00037     Fl_Font textfont_;
00038     Fl_Fonsize textsize_;
00039     Fl_Color textcolor_;
00040     int item_pathname_(char *name, int namelen, const Fl_Menu_Item *finditem,
00041                       const Fl_Menu_Item *menu=0) const;
00042 public:
00043     Fl_Menu_(int,int,int,int,const char * =0);

```



```

00072 ~Fl_Menu_();
00073
00074 int item_pathname(char *name, int namelen, const Fl_Menu_Item *finditem=0) const;
00075 const Fl_Menu_Item* picked(const Fl_Menu_Item*);
00076 const Fl_Menu_Item* find_item(const char *name);
00077 const Fl_Menu_Item* find_item(Fl_Callback*);
00078 const Fl_Menu_Item* find_item_with_user_data(void*);
00079 const Fl_Menu_Item* find_item_with_argument(long);
00080 int find_index(const char *name) const;
00081 int find_index(const Fl_Menu_Item *item) const;
00082 int find_index(Fl_Callback *cb) const;
00083
00095 const Fl_Menu_Item* test_shortcut() {return picked(menu()->test_shortcut());}
00096 void global();
00097
00133 const Fl_Menu_Item *menu() const {return menu_;}
00134 const Fl_Menu_Item *menu_end(); // in src/Fl_Menu_add.cxx
00135 void menu(const Fl_Menu_Item *m);
00136 void copy(const Fl_Menu_Item *m, void* user_data = 0);
00137 int insert(int index, const char*, int shortcut, Fl_Callback*, void* = 0, int = 0);
00138 int add(const char*, int shortcut, Fl_Callback*, void* = 0, int = 0); // see src/Fl_Menu_add.cxx
00140 int add(const char* a, const char* b, Fl_Callback* c, void* d = 0, int e = 0) {
00141     return add(a,fl_old_shortcut(b),c,d,e);
00142 }
00144 int insert(int index, const char* a, const char* b, Fl_Callback* c, void* d = 0, int e = 0) {
00145     return insert(index,a,fl_old_shortcut(b),c,d,e);
00146 }
00147 int add(const char *);
00148 int size() const;
00149 void size(int W, int H) { Fl_Widget::size(W, H); }
00150 void clear();
00151 int clear_submenu(int index);
00152 void replace(int,const char *);
00153 void remove(int);
00155 void shortcut(int i, int s) {menu_[i].shortcut(s);}
00157 void mode(int i,int fl) {menu_[i].flags = fl;}
00159 int mode(int i) const {return menu_[i].flags;}
00160
00162 const Fl_Menu_Item *mvalue() const {return value_;}
00163
00167 const Fl_Menu_Item *prev_mvalue() const {return prev_value_;}
00168 // Return the index into the menu() of the last item chosen by the user or -1.
00169 int value() const;
00170 // Set the internal value_ of the menu to the given Fl_Menu_Item.
00171 int value(const Fl_Menu_Item*);
00205 int value(int i) {
00206     if (!menu_ || i < 0 || i >= size())
00207         return 0;
00208     return value(menu_ + i);
00209 }
00210
00212 const char *text() const {return value_ ? value_->text : 0;}
00214 const char *text(int i) const {return menu_[i].text;}
00215
00217 Fl_Font textfont() const {return textfont_;}
00219 void textfont(Fl_Font c) {textfont_=c;}
00221 Fl_Fonsize textsize() const {return textsize_;}
00223 void textsize(Fl_Fonsize c) {textsize_=c;}
00225 Fl_Color textcolor() const {return textcolor_;}
00227 void textcolor(Fl_Color c) {textcolor_=c;}
00228
00233 Fl_Boxtype down_box() const {return (Fl_Boxtype)down_box_;}
00235 void down_box(Fl_Boxtype b) {down_box_ = b;}
00236
00240 Fl_Boxtype menu_box() const { return menu_box_; }
00245 void menu_box(Fl_Boxtype b) { menu_box_ = b; }
00246
00248 Fl_Color down_color() const {return selection_color();}
00250 void down_color(unsigned c) {selection_color(c);}
00251 void setonly(Fl_Menu_Item* item);
00252 };
00253
00254 #endif

```

12.82 Fl_Menu_Bar.H

```

00001 //
00002 // Menu bar header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2017 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //

```

```

00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Menu_Bar widget . */
00019
00020 #ifndef Fl_Menu_Bar_H
00021 #define Fl_Menu_Bar_H
00022
00023 #include "Fl_Menu_.H"
00024
00065 class FL_EXPORT Fl_Menu_Bar : public Fl_Menu_ {
00066     friend class Fl_Sys_Menu_Bar_Driver;
00067 protected:
00068     void draw() FL_OVERRIDE;
00069 public:
00070     int handle(int) FL_OVERRIDE;
00089     Fl_Menu_Bar(int X, int Y, int W, int H, const char *l=0);
00093     virtual void update() {}
00098     virtual void play_menu(const Fl_Menu_Item *item);
00099 };
00100
00101 #endif

```

12.83 Fl_Menu_Button.H

```

00001 //
00002 // Menu button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Menu_Button widget . */
00019
00020 #ifndef Fl_Menu_Button_H
00021 #define Fl_Menu_Button_H
00022
00023 #include "Fl_Menu_.H"
00024
00056 class FL_EXPORT Fl_Menu_Button : public Fl_Menu_ {
00057 protected:
00058     void draw() FL_OVERRIDE;
00059     static Fl_Menu_Button* pressed_menu_button_;
00060 public:
00067     enum popup_buttons {POPUP1 = 1,
00068         POPUP2,
00069         POPUP12,
00070         POPUP3,
00071         POPUP13,
00072         POPUP23,
00073         POPUP123
00074     };
00075     int handle(int) FL_OVERRIDE;
00076     const Fl_Menu_Item* popup();
00077     Fl_Menu_Button(int,int,int,int,const char * =0);
00078 };
00079
00080 #endif

```

12.84 Fl_Menu_Item.H File Reference

```

#include <FL/Fl_Widget.H>
#include <FL/Fl_Image.H>
#include <FL/Fl_Multi_Label.H>
#include <FL/platform_types.h>

```

Classes

- struct [FI_Menu_Item](#)

The [FI_Menu_Item](#) structure defines a single menu item that is used by the [FI_Menu](#) class.

Typedefs

- typedef [FI_Menu_Item](#) **FI_Menu**

Enumerations

- enum {
[FL_MENU_INACTIVE](#) = 1 , [FL_MENU_TOGGLE](#) = 2 , [FL_MENU_VALUE](#) = 4 , [FL_MENU_RADIO](#) = 8 ,
[FL_MENU_INVISIBLE](#) = 0x10 , [FL_SUBMENU_POINTER](#) = 0x20 , [FL_SUBMENU](#) = 0x40 , [FL_MENU_DIVIDER](#)
= 0x80 ,
[FL_MENU_HORIZONTAL](#) = 0x100 }
- enum {
FL_PUP_NONE = 0 , **FL_PUP_GREY** = [FL_MENU_INACTIVE](#) , **FL_PUP_GRAY** = [FL_MENU_INACTIVE](#) ,
FL_MENU_BOX = [FL_MENU_TOGGLE](#) ,
FL_PUP_BOX = [FL_MENU_TOGGLE](#) , **FL_MENU_CHECK** = [FL_MENU_VALUE](#) , **FL_PUP_CHECK** = [FL_MENU_VALUE](#) , **FL_PUP_RADIO** = [FL_MENU_RADIO](#) ,
FL_PUP_INVISIBLE = [FL_MENU_INVISIBLE](#) , **FL_PUP_SUBMENU** = [FL_SUBMENU_POINTER](#) }

Functions

- [FI_Shortcut fl_old_shortcut](#) (const char *)

Emulation of XForms named shortcuts.

12.84.1 Enumeration Type Documentation

12.84.1.1 anonymous enum

anonymous enum

Enumerator

FL_MENU_INACTIVE	Deactivate menu item (gray out)
FL_MENU_TOGGLE	Item is a checkbox toggle (shows checkbox for on/off state)
FL_MENU_VALUE	The on/off state for checkbox/radio buttons (if set, state is 'on')
FL_MENU_RADIO	Item is a radio button (one checkbox of many can be on)
FL_MENU_INVISIBLE	Item will not show up (shortcut will work)
FL_SUBMENU_POINTER	Indicates user_data() is a pointer to another menu array.
FL_SUBMENU	Item is a submenu to other items.
FL_MENU_DIVIDER	Creates divider line below this item. Also ends a group of radio buttons.
FL_MENU_HORIZONTAL	??? – reserved, internal (do not use) Note: ALL other bits in flags are reserved: do not use them for your own purposes!

12.85 FI_Menu_Item.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Menu item header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
```

```

00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Menu_Item_H
00018 #define Fl_Menu_Item_H
00019
00020 #include <FL/Fl_Widget.H>
00021 #include <FL/Fl_Image.H>
00022 #include <FL/Fl_Multi_Label.H>
00023 #include <FL/platform_types.h> // for FL_COMMAND and FL_CONTROL
00024
00025 // doxygen needs the following line to enable e.g. ::FL_MENU_TOGGLE to link to the enums
00026
00027 enum { // values for flags:
00028     FL_MENU_INACTIVE   = 1,
00029     FL_MENU_TOGGLE     = 2,
00030     FL_MENU_VALUE      = 4,
00031     FL_MENU_RADIO      = 8,
00032     FL_MENU_INVISIBLE  = 0x10,
00033     FL_SUBMENU_POINTER = 0x20,
00034     FL_SUBMENU         = 0x40,
00035     FL_MENU_DIVIDER    = 0x80,
00036     FL_MENU_HORIZONTAL = 0x100
00037 };
00038
00039 extern FL_EXPORT Fl_Shortcut fl_old_shortcut(const char*);
00040
00041 class Fl_Menu;
00042
00043 struct FL_EXPORT Fl_Menu_Item {
00044     const char *text;
00045     int shortcut;
00046     Fl_Callback *callback;
00047     void *user_data;
00048     int flags;
00049     uchar labeltype;
00050     Fl_Font labelfont;
00051     Fl_Fonsize labelsize;
00052     Fl_Color labelcolor;
00053
00054     // advance N items, skipping submenus:
00055     const Fl_Menu_Item *next(int=1) const;
00056
00057     Fl_Menu_Item *next(int i=1) {
00058         return (Fl_Menu_Item*)((const Fl_Menu_Item*)this)->next(i); }
00059
00060     const Fl_Menu_Item *first() const { return next(0); }
00061
00062     Fl_Menu_Item *first() { return next(0); }
00063
00064     // methods on menu items:
00065     const char* label() const { return text; }
00066
00067     void label(const char* a) { text = a; }
00068
00069     void label(Fl_Labeltype a, const char* b) {
00070         labeltype_ = a;
00071         text = b;
00072     }
00073
00074     void multi_label(const Fl_Multi_Label *ml) {
00075         label(FL_MULTI_LABEL, (const char *)ml);
00076     }
00077
00078     void image_label(const Fl_Image *image) {
00079         label(FL_IMAGE_LABEL, (const char *)image);
00080     }
00081
00082     Fl_Labeltype labeltype() const {return (Fl_Labeltype)labeltype_;}
00083
00084     void labeltype(Fl_Labeltype a) {labeltype_ = a;}
00085
00086     Fl_Color labelcolor() const {return labelcolor_;}
00087
00088     void labelcolor(Fl_Color a) {labelcolor_ = a;}
00089     Fl_Font labelfont() const {return labelfont_;}
00090
00091     void labelfont(Fl_Font a) {labelfont_ = a;}
00092

```

```

00298 Fl_Fontsize labelsz() const {return labelsz_;}
00299
00301 void labelsz(Fl_Fontsize a) {labelsz_ = a;}
00302
00310 Fl_Callback_p callback() const {return callback_;}
00311
00316 void callback(Fl_Callback* c, void* p) {callback_=c; user_data_=p;}
00317
00323 void callback(Fl_Callback* c) {callback_=c;}
00324
00330 void callback(Fl_Callback0 *c) {
00331     callback_ = (Fl_Callback *) (void *)c;
00332 }
00333
00340 void callback(Fl_Callback1 *c, long p = 0) {
00341     callback_ = (Fl_Callback *) (void *)c;
00342     user_data_ = (void *) (fl_intptr_t)p;
00343 }
00344
00348 void* user_data() const {return user_data_;}
00352 void user_data(void* v) {user_data_ = v;}
00359 long argument() const {return (long) (fl_intptr_t)user_data_;}
00367 void argument(long v) {user_data_ = (void*) (fl_intptr_t)v;}
00368
00370 int shortcut() const {return shortcut_;}
00371
00387 void shortcut(int s) {shortcut_ = s;}
00395 int submenu() const {return flags&(FL_SUBMENU|FL_SUBMENU_POINTER);}
00400 int checkbox() const {return flags&FL_MENU_TOGGLE;}
00407 int radio() const {return flags&FL_MENU_RADIO;}
00415 int value() const {return (flags & FL_MENU_VALUE) ? 1 : 0;}
00416
00418 void value(int v) { v ? set() : clear(); }
00419
00424 void set() {flags |= FL_MENU_VALUE;}
00425
00427 void clear() {flags &= ~FL_MENU_VALUE;}
00428
00429 void setonly(Fl_Menu_Item const* first = NULL);
00430
00432 int visible() const {return !(flags&FL_MENU_INVISIBLE);}
00433
00435 void show() {flags &= ~FL_MENU_INVISIBLE;}
00436
00438 void hide() {flags |= FL_MENU_INVISIBLE;}
00439
00441 int active() const {return !(flags&FL_MENU_INACTIVE);}
00442
00444 void activate() {flags &= ~FL_MENU_INACTIVE;}
00449 void deactivate() {flags |= FL_MENU_INACTIVE;}
00451 int activevisible() const {return !(flags & (FL_MENU_INACTIVE|FL_MENU_INVISIBLE));}
00452
00453 // compatibility for FLUID so it can set the image of a menu item...
00454
00460 void image(Fl_Image* image) {image->label(this);}
00461
00467 void image(Fl_Image& image) {image.label(this);}
00468
00469 // used by menubar:
00470 int measure(int* h, const Fl_Menu_*) const;
00471 void draw(int x, int y, int w, int h, const Fl_Menu_*, int t=0) const;
00472
00473 // popup menus without using an Fl_Menu_ widget:
00474 const Fl_Menu_Item* popup(
00475     int X, int Y,
00476     const char *title = 0,
00477     const Fl_Menu_Item* picked=0,
00478     const Fl_Menu_* = 0) const;
00479 const Fl_Menu_Item* pulldown(
00480     int X, int Y, int W, int H,
00481     const Fl_Menu_Item* picked = 0,
00482     const Fl_Menu_* = 0,
00483     const Fl_Menu_Item* title = 0,
00484     int menubar=0) const;
00485 const Fl_Menu_Item* test_shortcut() const;
00486 const Fl_Menu_Item* find_shortcut(int *ip=0, const bool require_alt = false) const;
00487
00493 void do_callback(Fl_Widget* o) const {Fl::callback_reason_=FL_REASON_SELECTED; callback_(o,
user_data_);}
00494
00500 void do_callback(Fl_Widget* o,void* arg) const {Fl::callback_reason_=FL_REASON_SELECTED;
callback_(o, arg);}
00501
00509 void do_callback(Fl_Widget* o,long arg) const {Fl::callback_reason_=FL_REASON_SELECTED; callback_(o,
(void*) (fl_intptr_t)arg);}
00510
00517 inline int checked() const {return value();}

```

```

00518
00525     inline void check() {set();}
00526
00533     inline void uncheck() {clear();}
00534
00535     int insert(int,const char*,int,Fl_Callback*,void* =0, int =0);
00536     int add(const char*, int shortcut, Fl_Callback*, void* =0, int = 0);
00537
00539     int add(const char*a, const char* b, Fl_Callback* c,
00540             void* d = 0, int e = 0) {
00541         return add(a,fl_old_shortcut(b),c,d,e);}
00542
00543     int size() const ;
00544 };
00545
00546 typedef Fl_Menu_Item Fl_Menu; // back compatibility
00547
00548 enum { // back-compatibility enum:
00549     FL_PUP_NONE      = 0,
00550     FL_PUP_GREY      = FL_MENU_INACTIVE,
00551     FL_PUP_GRAY      = FL_MENU_INACTIVE,
00552     FL_MENU_BOX      = FL_MENU_TOGGLE,
00553     FL_PUP_BOX       = FL_MENU_TOGGLE,
00554     FL_MENU_CHECK    = FL_MENU_VALUE,
00555     FL_PUP_CHECK     = FL_MENU_VALUE,
00556     FL_PUP_RADIO     = FL_MENU_RADIO,
00557     FL_PUP_INVISIBLE = FL_MENU_INVISIBLE,
00558     FL_PUP_SUBMENU   = FL_SUBMENU_POINTER
00559 };
00560
00561 #endif

```

12.86 Fl_Menu_Window.H

```

00001 //
00002 // Menu window header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Menu_Window widget . */
00019
00020 #ifndef Fl_Menu_Window_H
00021 #define Fl_Menu_Window_H
00022
00023 #include "Fl_Single_Window.H"
00024
00031 class FL_EXPORT Fl_Menu_Window : public Fl_Single_Window {
00032 public:
00033     ~Fl_Menu_Window();
00035     Fl_Menu_Window(int W, int H, const char *l = 0);
00037     Fl_Menu_Window(int X, int Y, int W, int H, const char *l = 0);
00038 };
00039
00040 #endif

```

12.87 fl_message.H

```

00001 //
00002 // Standard message header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //

```

```

00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef _FL_fl_message_H_
00018 #define _FL_fl_message_H_
00019
00020 #include "fl_ask.H"
00021
00022 #endif

```

12.88 Fl_Multi_Browser.H

```

00001 //
00002 // Multi browser header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Multi_Browser widget . */
00019
00020 #ifndef Fl_Multi_Browser_H
00021 #define Fl_Multi_Browser_H
00022
00023 #include "Fl_Browser.H"
00024
00041 class FL_EXPORT Fl_Multi_Browser : public Fl_Browser {
00042 public:
00049     Fl_Multi_Browser(int X,int Y,int W,int H,const char *L=0);
00050 };
00051
00052 #endif

```

12.89 Fl_Multi_Label.H

```

00001 //
00002 // Multi-label header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Multi_Label_H
00018 #define Fl_Multi_Label_H
00019
00020 class Fl_Widget;
00021 struct Fl_Menu_Item;
00022
00076 struct FL_EXPORT Fl_Multi_Label {
00080     const char* labela;
00084     const char* labelb;
00089     uchar typea;
00094     uchar typeb;
00095
00096     // This method is used to associate a Fl_Multi_Label with a Fl_Widget.
00097     void label(Fl_Widget*);
00098
00099     // This method is used to associate a Fl_Multi_Label with a Fl_Menu_Item.
00100     void label(Fl_Menu_Item*);
00101 };
00102
00103 #endif

```

12.90 Fl_Multiline_Input.H

```

00001 //
00002 // Multiline input header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2011 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Multiline_Input widget . */
00019
00020 #ifndef Fl_Multiline_Input_H
00021 #define Fl_Multiline_Input_H
00022
00023 #include "Fl_Input.H"
00024
00043 class FL_EXPORT Fl_Multiline_Input : public Fl_Input {
00044 public:
00051     Fl_Multiline_Input(int X,int Y,int W,int H,const char *l = 0);
00052 };
00053
00054 #endif

```

12.91 Fl_Multiline_Output.H

```

00001 //
00002 // Multi line output header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2011 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Multiline_Output widget . */
00019
00020 #ifndef Fl_Multiline_Output_H
00021 #define Fl_Multiline_Output_H
00022
00023 #include "Fl_Output.H"
00024
00047 class FL_EXPORT Fl_Multiline_Output : public Fl_Output {
00048 public:
00049
00056     Fl_Multiline_Output(int X,int Y,int W,int H,const char *l = 0);
00057 };
00058
00059 #endif

```

12.92 Fl_Native_File_Chooser.H File Reference

[Fl_Native_File_Chooser](#) widget.

```
#include <FL/Fl_Export.H>
```

```
#include <FL/Fl_File_Chooser.H>
```

Classes

- class [Fl_Native_File_Chooser](#)

This class lets an FLTK application easily and consistently access the operating system's native file chooser.

12.92.1 Detailed Description

[Fl_Native_File_Chooser](#) widget.

12.93 Fl_Native_File_Chooser.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // FLTK native OS file chooser widget
00003 //
00004 // Copyright 2004 Greg Ercolano.
00005 // Copyright 2005-2024 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00021 /* Implementation note:
00022
00023 class Fl_Native_File_Chooser <== public API used by applications
00024
00025 class Fl_Native_File_Chooser_Driver <== virtual API that a platform may implement
00026                                     this API has a do-nothing default implementation
00027
00028 class Fl_Native_File_Chooser_FLTK_Driver <== this API implementation is the default FLTK
file chooser
00029 class Fl_GTK_Native_File_Chooser_Driver <== this API implementation runs a GTK file
chooser
00030 class Fl_Kdialog_Native_File_Chooser_Driver <== this API implementation runs a KDE
file chooser
00031                                     it is determined at run-time if the GTK dynamic libraries are
available
00032                                     and the KDE file chooser runs under the KDE desktop
00033
00034 class Fl_Quartz_Native_File_Chooser_Driver <== this API implementation runs a Mac OS X file
chooser
00035
00036 class Fl_WinAPI_Native_File_Chooser_Driver <== this API implementation runs a Windows file
chooser
00037
00038
00039 Each platform must implement the constructor of the Fl_Native_File_Chooser class.
00040 This particular implementation:
00041
00042 Fl_Native_File_Chooser::Fl_Native_File_Chooser(int val) {
00043     platform_fnfc = new Fl_Native_File_Chooser_FLTK_Driver(val);
00044 }
00045
00046 can be used by any platform.
00047 No more code is required. The cross-platform Fl_Native_File_Chooser_FLTK.cxx file must be compiled in
libfltk,
00048 and the default FLTK file chooser will be used.
00049
00050 This other implementation:
00051 Fl_Native_File_Chooser::Fl_Native_File_Chooser(int val) {
00052     platform_fnfc = 0;
00053 }
00054 can be used by a platform that needs no file chooser.
00055 */
00056
00057 #ifndef FL_NATIVE_FILE_CHOOSER_H
00058 #define FL_NATIVE_FILE_CHOOSER_H
00059
00060 #include <FL/Fl_Export.H>
00061 #include <FL/Fl_File_Chooser.H>
00062
00063 class Fl_Native_File_Chooser_Driver;
00064
00130 class FL_EXPORT Fl_Native_File_Chooser {
00131 private:
00132     Fl_Native_File_Chooser_Driver *platform_fnfc;
00133 public:
00134     enum Type {

```

```

00135     BROWSE_FILE = 0,
00136     BROWSE_DIRECTORY,
00137     BROWSE_MULTI_FILE,
00138     BROWSE_MULTI_DIRECTORY,
00139     BROWSE_SAVE_FILE,
00140     BROWSE_SAVE_DIRECTORY
00141 };
00142 enum Option {
00143     NO_OPTIONS      = 0x0000,
00144     SAVEAS_CONFIRM  = 0x0001,
00145     NEW_FOLDER      = 0x0002,
00146     PREVIEW         = 0x0004,
00147     USE_FILTER_EXT  = 0x0008
00148 };
00149 static const char *file_exists_message;
00150
00151 Fl_Native_File_Chooser(int val = BROWSE_FILE); // each platform implements it
00152 ~Fl_Native_File_Chooser();
00153 void type(int t);
00154 int type() const;
00155 void options(int o);
00156 int options() const;
00157 int count() const;
00158 const char *filename() const;
00159 const char *filename(int i) const;
00160 void directory(const char *val);
00161 const char *directory() const;
00162 void title(const char *t);
00163 const char* title() const;
00164 const char *filter() const;
00165 void filter(const char *f);
00166 int filters() const;
00167 void filter_value(int i);
00168 int filter_value() const;
00169 void preset_file(const char*f);
00170 const char* preset_file() const;
00171 const char *errmsg() const;
00172 int show();
00173 };
00174
00175 class Fl_Native_File_Chooser_Driver {
00176 protected:
00177     static void chrcat(char *s, char c);
00178     static char *strapp(char *s, const char *val);
00179     static char *strfree(char *val);
00180     static char *strnew(const char *val);
00181 public:
00182     Fl_Native_File_Chooser_Driver(int) {}
00183     virtual ~Fl_Native_File_Chooser_Driver() {}
00184     virtual void type(int) {}
00185     virtual int type() const {return 0;}
00186     virtual void options(int) {}
00187     virtual int options() const {return 0;}
00188     virtual int count() const {return 0;}
00189     virtual const char *filename() const {return 0;}
00190     virtual const char *filename(int) const {return 0;}
00191     virtual void directory(const char *) {}
00192     virtual const char *directory() const {return 0;}
00193     virtual void title(const char *) {}
00194     virtual const char* title() const {return 0;}
00195     virtual const char *filter() const {return 0;}
00196     virtual void filter(const char *) {}
00197     virtual int filters() const {return 0;}
00198     virtual void filter_value(int) {}
00199     virtual int filter_value() const {return 0;}
00200     virtual void preset_file(const char*) {}
00201     virtual const char* preset_file() const {return 0;}
00202     virtual const char *errmsg() const {return 0;}
00203     virtual int show() {return 1;}
00204 };
00205
00206 class Fl_Native_File_Chooser_FLTK_Driver : public Fl_Native_File_Chooser_Driver {
00207 private:
00208     void errmsg(const char *msg);
00209     int type_fl_file(int val);
00210     int exist_dialog();
00211     void parse_filter();
00212 protected:
00213     int _btype; // kind-of browser to show()
00214     int _options; // general options
00215     int _nfilters;
00216     char *_filter; // user supplied filter
00217     char *_parsedfilt; // parsed filter
00218     int _filtvalue; // selected filter
00219     char *_preset_file;
00220     char *_prevvalue; // Returned filename
00221     char *_directory;

```

```

00250     char *_errmsg;                // error message
00251     Fl_File_Chooser *_file_chooser;
00252 public:
00253     Fl_Native_File_Chooser_Fltk_Driver(int val);
00254     virtual ~Fl_Native_File_Chooser_Fltk_Driver();
00255     void type(int t) FL_OVERRIDE;
00256     int type() const FL_OVERRIDE;
00257     void options(int o) FL_OVERRIDE;
00258     int options() const FL_OVERRIDE;
00259     int count() const FL_OVERRIDE;
00260     const char *filename() const FL_OVERRIDE;
00261     const char *filename(int i) const FL_OVERRIDE;
00262     void directory(const char *val) FL_OVERRIDE;
00263     const char *directory() const FL_OVERRIDE;
00264     void title(const char *t) FL_OVERRIDE;
00265     const char* title() const FL_OVERRIDE;
00266     const char *filter() const FL_OVERRIDE;
00267     void filter(const char *f) FL_OVERRIDE;
00268     int filters() const FL_OVERRIDE;
00269     void filter_value(int i) FL_OVERRIDE;
00270     int filter_value() const FL_OVERRIDE;
00271     void preset_file(const char*f) FL_OVERRIDE;
00272     const char* preset_file() const FL_OVERRIDE;
00273     const char *errmsg() const FL_OVERRIDE;
00274     int show() FL_OVERRIDE;
00275 };
00276
00283 #endif /*FL_NATIVE_FILE_CHOOSER_H*/

```

12.94 Fl_Nice_Slider.H

```

00001 //
00002 // "Nice" slider header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Nice_Slider widget . */
00019
00020 #ifndef Fl_Nice_Slider_H
00021 #define Fl_Nice_Slider_H
00022
00023 #include "Fl_Slider.H"
00024
00025 class FL_EXPORT Fl_Nice_Slider : public Fl_Slider {
00026 public:
00027     Fl_Nice_Slider(int X,int Y,int W,int H,const char *L=0);
00028 };
00029
00030 #endif

```

12.95 Fl_Object.H

```

00001 //
00002 // Old Fl_Object header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 // This file is provided for back compatibility only. Please use Fl_Widget
00018 #ifndef Fl_Object

```

```
00019 #define Fl_Object Fl_Widget
00020 #endif
00021 #include "Fl_Widget.H"
```

12.96 Fl_Output.H

```
00001 //
00002 // Output header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Output widget . */
00019
00020 #ifndef Fl_Output_H
00021 #define Fl_Output_H
00022
00023 #include "Fl_Input.H"
00045 class FL_EXPORT Fl_Output : public Fl_Input {
00046 public:
00054     Fl_Output(int X,int Y,int W,int H, const char *l = 0);
00055 };
00056
00057 #endif
```

12.97 Fl_Overlay_Window.H

```
00001 //
00002 // Overlay window header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Overlay_Window class . */
00019
00020 #ifndef Fl_Overlay_Window_H
00021 #define Fl_Overlay_Window_H
00022
00023 #include "Fl_Double_Window.H"
00024
00036 class FL_EXPORT Fl_Overlay_Window : public Fl_Double_Window {
00037 #ifndef FL_DOXYGEN
00038     friend class _Fl_Overlay;
00039     friend class Fl_Window_Driver;
00040 #endif
00041 public:
00048     virtual void draw_overlay() = 0;
00049 private:
00050     Fl_Window *overlay_;
00051 public:
00052     void show() FL_OVERRIDE;
00053     void hide() FL_OVERRIDE;
00054     void flush() FL_OVERRIDE;
00055     void resize(int,int,int,int) FL_OVERRIDE;
00056     ~Fl_Overlay_Window();
00058     int can_do_overlay();
00059     void redraw_overlay();
00060 protected:
00064     Fl_Overlay_Window(int W, int H, const char *l=0);
00071     Fl_Overlay_Window(int X, int Y, int W, int H, const char *l=0);
```

```

00072 public:
00074     void show(int a, char **b) {Fl_Double_Window::show(a,b);}
00075     Fl_Overlay_Window *as_overlay_window() FL_OVERRIDE {return this; }
00076 };
00077
00078 #endif

```

12.98 Fl_Pack.H

```

00001 //
00002 // Pack header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2020 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Pack widget . */
00019
00020 #ifndef Fl_Pack_H
00021 #define Fl_Pack_H
00022
00023 #include <FL/Fl_Group.H>
00024
00054 class FL_EXPORT Fl_Pack : public Fl_Group {
00055     int spacing_;
00056
00057 public:
00058     enum { // values for type(int)
00059         VERTICAL = 0,
00060         HORIZONTAL = 1
00061     };
00062
00063 protected:
00064     void draw() FL_OVERRIDE;
00065
00066 public:
00067     Fl_Pack(int X, int Y, int W, int H, const char *L = 0);
00068
00073     int spacing() const {return spacing_;}
00074
00079     void spacing(int i) {spacing_ = i;}
00080
00093     uchar horizontal() const {return type();}
00094
00095     void resize(int X, int Y, int W, int H) FL_OVERRIDE;
00098     void clear() { Fl_Group::clear(); resizable(NULL); }
00099 };
00100
00101 #endif

```

12.99 Fl_Paged_Device.H File Reference

declaration of class [Fl_Paged_Device](#).

```
#include <FL/Fl_Widget_Surface.H>
```

Classes

- class [Fl_Paged_Device](#)
Represents page-structured drawing surfaces.
- struct [Fl_Paged_Device::page_format](#)
width, height and name of a page format

Macros

- `#define NO_PAGE_FORMATS 30` /* MSVC6 compilation fix */
Number of elements in enum `Page_Format`.

12.99.1 Detailed Description

declaration of class `Fl_Paged_Device`.

12.100 Fl_Paged_Device.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Printing support for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2010-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00021 #ifndef Fl_Paged_Device_H
00022 #define Fl_Paged_Device_H
00023
00024 #include <FL/Fl_Widget_Surface.H>
00025
00026
00028 #define NO_PAGE_FORMATS 30 /* MSVC6 compilation fix */
00029
00036 class FL_EXPORT Fl_Paged_Device : public Fl_Widget_Surface {
00037 protected:
00039     Fl_Paged_Device() : Fl_Widget_Surface(NULL) {}
00040 public:
00045     enum Page_Format {
00046         A0 = 0,
00047         A1,
00048         A2,
00049         A3,
00050         A4,
00051         A5,
00052         A6,
00053         A7,
00054         A8,
00055         A9,
00056         B0,
00057         B1,
00058         B2,
00059         B3,
00060         B4,
00061         B5,
00062         B6,
00063         B7,
00064         B8,
00065         B9,
00066         B10,
00067         CSE,
00068         DLE,
00069         EXECUTIVE,
00070         FOLIO,
00071         LEDGER,
00072         LEGAL,
00073         LETTER,
00074         TABLOID,
00075         ENVELOPE,
00076         MEDIA = 0x1000
00077     };
00081     enum Page_Layout {
00082         PORTRAIT = 0,
00083         LANDSCAPE = 0x100,
00084         REVERSED = 0x200,
00085         ORIENTATION = 0x300
00086     };
00087
00090     typedef struct {
```

```

00092     int width;
00094     int height;
00096     const char *name;
00097 } page_format;
00100 static const page_format page_formats[NO_PAGE_FORMATS];
00102 virtual ~Fl_Paged_Device() {}
00103 virtual int begin_job(int pagecount = 0, int *frompage = NULL, int *topage = NULL, char
**perr_message = NULL);
00106 int start_job(int pagecount = 0, int *frompage = NULL, int *topage = NULL, char **perr_message =
NULL) {
00107     return begin_job(pagecount, frompage, topage, perr_message);
00108 }
00109 virtual int begin_page(void);
00112 int start_page() {return begin_page();}
00113 virtual void margins(int *left, int *top, int *right, int *bottom);
00114 virtual void scale(float scale_x, float scale_y = 0.);
00115 virtual void rotate(float angle);
00117 void print_widget(Fl_Widget* widget, int delta_x = 0, int delta_y = 0) {draw(widget, delta_x,
delta_y);}
00119 void print_window(Fl_Window *win, int x_off = 0, int y_off = 0) {
00120     draw_decorated_window(win, x_off, y_off);
00121 }
00122 virtual int end_page (void);
00123 virtual void end_job (void);
00124 };
00125
00126 #endif // Fl_Paged_Device_H

```

12.101 Fl_PDF_File_Surface.H

```

00001 //
00002 // Declaration of class Fl_PDF_File_Surface for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef PDF_FILE_SURFACE_H
00018 #define PDF_FILE_SURFACE_H
00019
00020 #include <FL/Fl_Paged_Device.H>
00021
00031 class FL_EXPORT Fl_PDF_File_Surface : public Fl_Paged_Device {
00032 private:
00033     const char **out_filename_;
00034     Fl_Paged_Device *platform_surface_;
00035     static Fl_Paged_Device *new_platform_pdf_surface(const char **);
00036 public:
00040     static const char * format_dialog_title;
00041     static const char * format_dialog_page_size;
00042     static const char * format_dialog_orientation;
00043     static const char * format_dialog_default;
00044     Fl_PDF_File_Surface();
00045     ~Fl_PDF_File_Surface();
00055     int begin_job(const char* defaultfilename, char **perr = NULL);
00057     int begin_job(int, int *, int *, char **) FL_OVERRIDE {return 1;}
00066     int begin_document(const char* pathname,
00067         enum Fl_Paged_Device::Page_Format format = Fl_Paged_Device::A4,
00068         enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT,
00069         char **perr = NULL);
00070     int printable_rect(int *w, int *h) FL_OVERRIDE { return platform_surface_>printable_rect(w,h); }
00071     void margins(int *left, int *top, int *right, int *bottom) FL_OVERRIDE {
00072         platform_surface_>margins(left,top,right,bottom);
00073     }
00074     void origin(int x, int y) FL_OVERRIDE {platform_surface_>origin(x, y);}
00075     void origin(int *x, int *y) FL_OVERRIDE {platform_surface_>origin(x, y);}
00076     void scale(float s_x, float s_y = 0) FL_OVERRIDE {platform_surface_>scale(s_x, s_y);}
00077     void rotate(float angle) FL_OVERRIDE {platform_surface_>rotate(angle);}
00078     void translate(int x, int y) FL_OVERRIDE {platform_surface_>translate(x, y);}
00079     void untranslate() FL_OVERRIDE {platform_surface_>untranslate();};
00080     int begin_page(void) FL_OVERRIDE {return platform_surface_>begin_page();};
00081     int end_page(void) FL_OVERRIDE {return platform_surface_>end_page();};
00082     void end_job(void) FL_OVERRIDE {return platform_surface_>end_job();};
00084     inline const char *pdf_filename() { return *out_filename_; }
00085     void set_current() FL_OVERRIDE { if (platform_surface_) platform_surface_>set_current(); }
00086     bool is_current() FL_OVERRIDE { return surface() == platform_surface_; }

```

```

00087 };
00088
00089 #endif // PDF_FILE_SURFACE_H

```

12.102 Fl_Pixmap.H

```

00001 //
00002 // Pixmap header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2017 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Pixmap widget . */
00019
00020 #ifndef Fl_Pixmap_H
00021 #define Fl_Pixmap_H
00022 #    include "Fl_Image.H"
00023
00024 class Fl_Widget;
00025 struct Fl_Menu_Item;
00026
00027 // Older C++ compilers don't support the explicit keyword... :(
00028 #    if defined(__sgi) && !defined(_COMPILER_VERSION)
00029 #        define explicit
00030 #    endif // __sgi && !_COMPILER_VERSION
00031
00032 class FL_EXPORT Fl_Pixmap : public Fl_Image {
00033     friend class Fl_Graphics_Driver;
00034     void copy_data();
00035     void delete_data();
00036     void set_data(const char * const *p);
00037
00038 protected:
00039     void measure();
00040
00041 public:
00042     int alloc_data; // Non-zero if data was allocated
00043
00044 private:
00045     // for internal use
00046     fl_uintptr_t id_;
00047     fl_uintptr_t mask_;
00048     int cache_w_, cache_h_; // size of pixmap when cached
00049
00050 public:
00051     explicit Fl_Pixmap(char * const * D) : Fl_Image(-1,0,1), alloc_data(0), id_(0), mask_(0)
00052     {set_data((const char*const*)D); measure();}
00053     explicit Fl_Pixmap(uchar* const * D) : Fl_Image(-1,0,1), alloc_data(0), id_(0), mask_(0)
00054     {set_data((const char*const*)D); measure();}
00055     explicit Fl_Pixmap(const char * const * D) : Fl_Image(-1,0,1), alloc_data(0), id_(0), mask_(0)
00056     {set_data((const char*const*)D); measure();}
00057     explicit Fl_Pixmap(const uchar* const * D) : Fl_Image(-1,0,1), alloc_data(0), id_(0), mask_(0)
00058     {set_data((const char*const*)D); measure();}
00059     virtual ~Fl_Pixmap();
00060     Fl_Image *copy(int W, int H) const FL_OVERRIDE;
00061     Fl_Image *copy() const { return Fl_Image::copy(); }
00062     void color_average(Fl_Color c, float i) FL_OVERRIDE;
00063     void desaturate() FL_OVERRIDE;
00064     void draw(int X, int Y, int W, int H, int cx=0, int cy=0) FL_OVERRIDE;
00065     void draw(int X, int Y) {draw(X, Y, w(), h(), 0, 0);}
00066     void label(Fl_Widget*w) FL_OVERRIDE;
00067     void label(Fl_Menu_Item*m) FL_OVERRIDE;
00068     void uncache() FL_OVERRIDE;
00069     int cache_w() {return cache_w_;}
00070     int cache_h() {return cache_h_;}
00071 };
00072
00073 #endif

```


12.103 Fl_Plugin.H

```

00001 //
00002 // A Plugin system for FLTK, implemented in Fl_Preferences.cxx.
00003 //
00004 // Copyright 2002-2023 by Matthias Melcher.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Plugin class . */
00019
00020 #ifndef Fl_Plugin_H
00021 # define Fl_Plugin_H
00022
00023 # include "Fl_Preferences.H"
00024
00025
00059 class FL_EXPORT Fl_Plugin {
00060     Fl_Preferences::ID id;
00061 public:
00062     Fl_Plugin(const char *klass, const char *name);
00063     virtual ~Fl_Plugin();
00064 };
00065
00066
00071 class FL_EXPORT Fl_Plugin_Manager : public Fl_Preferences {
00072 public:
00073     Fl_Plugin_Manager(const char *klass);
00074     ~Fl_Plugin_Manager();
00075
00076     int plugins() { return groups(); }
00077     Fl_Plugin *plugin(int index);
00078     Fl_Plugin *plugin(const char *name);
00079     Fl_Preferences::ID addPlugin(const char *name, Fl_Plugin *plugin);
00080
00081     static void removePlugin(Fl_Preferences::ID id);
00082     static int load(const char *filename);
00083     static int loadAll(const char *dirpath, const char *pattern=0);
00084 };
00085
00086 #endif // !Fl_Preferences_H

```

12.104 Fl_PNG_Image.H

```

00001 //
00002 // PNG image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_PNG_Image class . */
00019
00020 #ifndef Fl_PNG_Image_H
00021 #define Fl_PNG_Image_H
00022 # include "Fl_Image.H"
00023
00030 class FL_EXPORT Fl_PNG_Image : public Fl_RGB_Image {
00031     friend class Fl_ICO_Image;
00032 public:
00033     Fl_PNG_Image(const char* filename);
00034     Fl_PNG_Image (const char *name_png, const unsigned char *buffer, int datasize);

```

```

00036 private:
00037     Fl_PNG_Image(const char *filename, int offset); // used by Fl_ICO_Image
00038     void load_png(const char *name_png, int offset, const unsigned char *buffer_png, int datasize);
00039 };
00040
00041 // Support functions to write PNG image files (since 1.4.0)
00042
00043 FL_EXPORT int fl_write_png(const char *filename, Fl_RGB_Image *img);
00044 FL_EXPORT int fl_write_png(const char *filename, const char *pixels, int w, int h, int d=3, int ld=0);
00045 FL_EXPORT int fl_write_png(const char *filename, const unsigned char *pixels, int w, int h, int d=3,
                           int ld=0);
00046
00047 #endif

```

12.105 Fl_PNM_Image.H

```

00001 //
00002 // PNM image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_PNM_Image class . */
00019
00020 #ifndef Fl_PNM_Image_H
00021 #define Fl_PNM_Image_H
00022 # include "Fl_Image.H"
00023
00024 class FL_EXPORT Fl_PNM_Image : public Fl_RGB_Image {
00025 public:
00026     Fl_PNM_Image(const char* filename);
00027 };
00028
00029 #endif

```

12.106 Fl_Positioner.H

```

00001 //
00002 // Positioner header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Positioner widget . */
00019
00020 #ifndef Fl_Positioner_H
00021 #define Fl_Positioner_H
00022
00023 #ifndef Fl_Widget_H
00024 #include "Fl_Widget.H"
00025 #endif
00026
00027 class FL_EXPORT Fl_Positioner : public Fl_Widget {
00028 public:
00029     double xmin, ymin;
00030     double xmax, ymax;
00031     double xvalue_, yvalue_;
00032     double xstep_, ystep_;
00033 };

```

```

00041
00042 protected:
00043
00044 // these allow subclasses to put the dial in a smaller area:
00045 void draw(int, int, int, int);
00046 int handle(int, int, int, int, int);
00047 void draw() FL_OVERRIDE;
00048
00049 public:
00050
00051 int handle(int) FL_OVERRIDE;
00052 FL_Positioner(int x,int y,int w,int h, const char *l=0);
00053 double xvalue() const {return xvalue_;}
00054 double yvalue() const {return yvalue_;}
00055 int xvalue(double);
00056 int yvalue(double);
00057 int value(double,double);
00058 void xbounds(double, double);
00059 double xminimum() const {return xmin;}
00060 void xminimum(double a) {xbounds(a, xmax);}
00061 double xmaximum() const {return xmax;}
00062 void xmaximum(double a) {xbounds(xmin, a);}
00063 void ybounds(double, double);
00064 double yminimum() const {return ymin;}
00065 void yminimum(double a) {ybounds(a, ymax);}
00066 double ymaximum() const {return ymax;}
00067 void ymaximum(double a) {ybounds(ymin, a);}
00068 void xstep(double a) {xstep_ = a;}
00069 void ystep(double a) {ystep_ = a;}
00070 };
00071
00072 #endif

```

12.107 FI_PostScript.H File Reference

declaration of classes [FI_PostScript_File_Device](#) and [FI_EPS_File_Surface](#).

```

#include <FL/Fl_Paged_Device.H>
#include <FL/fl_draw.H>
#include <stdarg.h>

```

Classes

- class [FI_EPS_File_Surface](#)
Encapsulated PostScript drawing surface.
- class [FI_PostScript_File_Device](#)
To send graphical output to a PostScript file.

Typedefs

- typedef int(* [FI_PostScript_Close_Command](#)) (FILE *)
Signature of functions FLTK may use to close FILE variables after PostScript/EPS output.

12.107.1 Detailed Description

declaration of classes [FI_PostScript_File_Device](#) and [FI_EPS_File_Surface](#).

12.107.2 Typedef Documentation

12.107.2.1 FI_PostScript_Close_Command

```
typedef int(* Fl_PostScript_Close_Command) (FILE *)
```

Signature of functions FLTK may use to close FILE variables after PostScript/EPS output.

A non-null return value indicates output error.

See also

[FI_PostScript_File_Device::close_command\(\)](#) and [FI_EPS_File_Surface::FI_EPS_File_Surface\(\)](#).

12.108 Fl_PostScript.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Support for graphics output to PostScript file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2010-2020 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00021 #ifndef Fl_PostScript_H
00022 #define Fl_PostScript_H
00023
00024 #include <FL/Fl_Paged_Device.H>
00025 #include <FL/fl_draw.H>
00026 #include <stdarg.h>
00027
00032 extern "C" {
00033     typedef int (*Fl_PostScript_Close_Command)(FILE *);
00034 }
00035
00036 class Fl_PostScript_Graphics_Driver;
00037
00078 class FL_EXPORT Fl_PostScript_File_Device : public Fl_Paged_Device {
00079 private:
00080     // memorize the display's current font to restore it when the object ceases being current
00081     Fl_Font display_font_;
00082     Fl_Fontsize display_size_;
00083 protected:
00087     inline Fl_PostScript_Graphics_Driver *driver() { return
        (Fl_PostScript_Graphics_Driver*)Fl_Surface_Device::driver(); }
00088 public:
00090     Fl_PostScript_File_Device();
00092     ~Fl_PostScript_File_Device();
00094     int begin_job(int pagecount, int* from, int* to, char **perr_message) FL_OVERRIDE;
00104     int begin_job(int pagecount = 0, enum Fl_Paged_Device::Page_Format format = Fl_Paged_Device::A4,
00105                 enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT);
00108     int start_job(int pagecount = 0, enum Fl_Paged_Device::Page_Format format = Fl_Paged_Device::A4,
00109                 enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT) {
00110         return begin_job(pagecount, format, layout);
00111     }
00122     int begin_job(FILE *ps_output, int pagecount = 0, enum Fl_Paged_Device::Page_Format format =
        Fl_Paged_Device::A4,
00123                 enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT);
00126     int start_job(FILE *ps_output, int pagecount = 0, enum Fl_Paged_Device::Page_Format format =
        Fl_Paged_Device::A4,
00127                 enum Fl_Paged_Device::Page_Layout layout = Fl_Paged_Device::PORTRAIT) {
00128         return begin_job(ps_output, pagecount, format, layout);
00129     }
00130
00131     int begin_page (void) FL_OVERRIDE;
00132     int printable_rect(int *w, int *h) FL_OVERRIDE;
00133     void margins(int *left, int *top, int *right, int *bottom) FL_OVERRIDE;
00134     void origin(int *x, int *y) FL_OVERRIDE;
00135     void origin(int x, int y) FL_OVERRIDE;
00136     void scale (float scale_x, float scale_y = 0.) FL_OVERRIDE;
00137     void rotate(float angle) FL_OVERRIDE;
00138     void translate(int x, int y) FL_OVERRIDE;
00139     void untranslate(void) FL_OVERRIDE;
00140     int end_page (void) FL_OVERRIDE;
00144     void end_job(void) FL_OVERRIDE;
00146     static const char *file_chooser_title;
00148     FILE *file();
00150     void close_command(Fl_PostScript_Close_Command cmd);
00151     void set_current() FL_OVERRIDE;
00152     void end_current() FL_OVERRIDE;
00153 };
00154
00172 class FL_EXPORT Fl_EPS_File_Surface : public Fl_Widget_Surface {
00173 protected:
00175     inline Fl_PostScript_Graphics_Driver *driver() { return
        (Fl_PostScript_Graphics_Driver*)Fl_Surface_Device::driver(); }
00176 public:
00191     Fl_EPS_File_Surface(int width, int height, FILE *eps_output,
00192                       Fl_Color background = FL_WHITE, Fl_PostScript_Close_Command closef = NULL);
00200     ~Fl_EPS_File_Surface();
00201     int printable_rect(int *w, int *h) FL_OVERRIDE;

```

```

00203 FILE *file();
00204 void origin(int x, int y) FL_OVERRIDE;
00205 void origin(int *px, int *py) FL_OVERRIDE;
00206 void translate(int x, int y) FL_OVERRIDE;
00207 void untranslate() FL_OVERRIDE;
00211 int close();
00212 };
00213
00214 #endif // Fl_PostScript_H

```

12.109 Fl_Preferences.H

```

00001 //
00002 // Preferences implementation for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2002-2023 by Matthias Melcher.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Preferences class . */
00019
00020 #ifndef Fl_Preferences_H
00021 # define Fl_Preferences_H
00022
00023 # include <stdio.h>
00024 # include "Fl_Export.H"
00025 # include "fl_attr.h"
00026
00027 #include <string>
00028
00122 class FL_EXPORT Fl_Preferences {
00123
00124 public:
00128     enum Root {
00129         UNKNOWN_ROOT_TYPE = -1,
00130         SYSTEM              = 0,
00131         USER,
00132         MEMORY,
00133         ROOT_MASK           = 0x00FF,
00134         CORE                 = 0x0100,
00135         C_LOCALE             = 0x1000,
00137         CLEAR                = 0x2000,
00138         SYSTEM_L             = SYSTEM | C_LOCALE,
00139         USER_L              = USER | C_LOCALE,
00140         CORE_SYSTEM_L       = CORE | SYSTEM_L,
00141         CORE_USER_L         = CORE | USER_L,
00142         CORE_SYSTEM         = CORE | SYSTEM,
00143         CORE_USER           = CORE | USER
00144     };
00145
00153     typedef void *ID;
00154
00155     static const char *new_UUID();
00156
00161     static const unsigned int NONE = 0x0000;
00163     static const unsigned int USER_READ_OK = 0x0001;
00165     static const unsigned int USER_WRITE_OK = 0x0002;
00167     static const unsigned int USER_OK = USER_READ_OK | USER_WRITE_OK;
00169     static const unsigned int SYSTEM_READ_OK = 0x0004;
00171     static const unsigned int SYSTEM_WRITE_OK = 0x0008;
00173     static const unsigned int SYSTEM_OK = SYSTEM_READ_OK | SYSTEM_WRITE_OK;
00175     static const unsigned int APP_OK = SYSTEM_OK | USER_OK;
00177     static const unsigned int CORE_READ_OK = 0x0010;
00179     static const unsigned int CORE_WRITE_OK = 0x0020;
00181     static const unsigned int CORE_OK = CORE_READ_OK | CORE_WRITE_OK;
00183     static const unsigned int ALL_READ_OK = USER_READ_OK | SYSTEM_READ_OK | CORE_READ_OK;
00185     static const unsigned int ALL_WRITE_OK = USER_WRITE_OK | SYSTEM_WRITE_OK | CORE_WRITE_OK;
00187     static const unsigned int ALL = ALL_READ_OK | ALL_WRITE_OK;
00188
00189     static void file_access(unsigned int flags);
00190     static unsigned int file_access();
00191     static Root filename( char *buffer, size_t buffer_size, Root root, const char *vendor, const char
00192 *application );
00193     Fl_Preferences( Root root, const char *vendor, const char *application );

```

```

00194 Fl_Preferences( const char *path, const char *vendor, const char *application, Root flags );
00195 Fl_Preferences( Fl_Preferences &parent, const char *group );
00196 Fl_Preferences( Fl_Preferences *parent, const char *group );
00197 Fl_Preferences( Fl_Preferences &parent, int groupIndex );
00198 Fl_Preferences( Fl_Preferences *parent, int groupIndex );
00199 Fl_Preferences(const Fl_Preferences&);
00200 Fl_Preferences( ID id );
00201 virtual ~Fl_Preferences();
00202
00203 FL_DEPRECATED("since 1.4.0 - use Fl_Preferences(path, vendor, application, flags) instead",
00204               Fl_Preferences( const char *path, const char *vendor, const char *application ) );
00205
00206 Root filename( char *buffer, size_t buffer_size);
00207
00210 ID id() { return (ID)node; }
00211
00214 static char remove(ID id_) { return ((Node*)id_)->remove(); }
00215
00218 const char *name() { return node->name(); }
00219
00222 const char *path() { return node->path(); }
00223
00224 int groups();
00225 const char *group( int num_group );
00226 char group_exists( const char *key );
00227 char delete_group( const char *group );
00228 char delete_all_groups();
00229
00230 int entries();
00231 const char *entry( int index );
00232 char entry_exists( const char *key );
00233 char delete_entry( const char *entry );
00234 char delete_all_entries();
00235
00236 char clear();
00237
00238 char set( const char *entry, int value );
00239 char set( const char *entry, float value );
00240 char set( const char *entry, float value, int precision );
00241 char set( const char *entry, double value );
00242 char set( const char *entry, double value, int precision );
00243 char set( const char *entry, const char *value );
00244 char set( const char *entry, const void *value, int size );
00245 char set( const char *entry, const std::string &value );
00246
00247 char get( const char *entry, int &value, int defaultValue );
00248 char get( const char *entry, float &value, float defaultValue );
00249 char get( const char *entry, double &value, double defaultValue );
00250 char get( const char *entry, char *&value, const char *defaultValue );
00251 char get( const char *entry, char *value, const char *defaultValue, int maxSize );
00252 char get( const char *entry, void *&value, const void *defaultValue, int defaultSize );
00253 char get( const char *entry, void *value, const void *defaultValue, int defaultSize, int maxSize
);
00254 char get( const char *entry, void *value, const void *defaultValue, int defaultSize, int *size );
00255 char get( const char *entry, std::string &value, const std::string &defaultValue );
00256
00257 int size( const char *entry );
00258
00259 char get_userdata_path( char *path, int pathlen );
00260
00261 int flush();
00262
00263 int dirty();
00264
00266 static const char *newUUID() { return new_UUID(); }
00267 char groupExists( const char *key ) { return group_exists(key); }
00268 char deleteGroup( const char *group ) { return delete_group(group); }
00269 char deleteAllGroups() { return delete_all_groups(); }
00270 char entryExists( const char *key ) { return entry_exists(key); }
00271 char deleteEntry( const char *entry ) { return delete_entry(entry); }
00272 char deleteAllEntries() { return delete_all_entries(); }
00273 char getUserdataPath( char *path, int pathlen ) { return get_userdata_path(path, pathlen); }
00288 class FL_EXPORT Name {
00289     char *data_;
00290
00291 public:
00292     Name( unsigned int n );
00293     Name( const char *format, ... );
00294
00295     operator const char *() { return data_; }
00301     ~Name();
00302 };
00303
00305 struct Entry {
00306     char *name, *value;
00307 };

```

```

00308
00309 private:
00310     Fl_Preferences() : node(0), rootNode(0) { }
00311     Fl_Preferences &operator=(const Fl_Preferences&);
00312
00313     static char nameBuffer[128];
00314     static char uuidBuffer[40];
00315     static Fl_Preferences *runtimePrefs;
00316     static unsigned int fileAccess_;
00317
00318 public: // older Sun compilers need this (public definition of the following classes)
00319     class RootNode;
00320
00321     class FL_EXPORT Node {           // a node contains a list to all its entries
00322                                     // and all means to manage the tree structure
00323     public:
00324         Node *first_child_, *next_;
00325         union {                     // these two are mutually exclusive
00326             Node *parent_;          // top_ bit clear
00327             RootNode *root_node_;   // top_ bit set
00328         };
00329         char *path_;
00330         Entry *entry_;
00331         int nEntry_, NEntry_;
00332         unsigned char dirty_:1;
00333         unsigned char top_:1;
00334         unsigned char indexed_:1;
00335         // indexing routines
00336         Node **index_;
00337         int nIndex_, NIndex_;
00338         void createIndex();
00339         void updateIndex();
00340         void deleteIndex();
00341     public:
00342         static int lastEntrySet;
00343     public:
00344         Node( const char *path );
00345         ~Node();
00346         // node methods
00347         int write( FILE *f );
00348         const char *name();
00349         const char *path() { return path_; }
00350         Node *find( const char *path );
00351         Node *search( const char *path, int offset=0 );
00352         Node *childNode( int ix );
00353         Node *addChild( const char *path );
00354         void setParent( Node *parent );
00355         Node *parent() { return top_?0L:parent_; }
00356         void setRoot( RootNode *r ) { root_node_ = r; top_ = 1; }
00357         RootNode *findRoot();
00358         char remove();
00359         char dirty();
00360         void clearDirtyFlags();
00361         void deleteAllChildren();
00362         // entry methods
00363         int nChildren();
00364         const char *child( int ix );
00365         void set( const char *name, const char *value );
00366         void set( const char *line );
00367         void add( const char *line );
00368         const char *get( const char *name );
00369         int getEntry( const char *name );
00370         char deleteEntry( const char *name );
00371         void deleteAllEntries();
00372         int nEntry() { return nEntry_; }
00373         Entry &entry(int i) { return entry_[i]; }
00374     };
00375     friend class Node;
00376
00377     class FL_EXPORT RootNode {       // the root node manages file paths and basic reading and writing
00378     public:
00379         Fl_Preferences *prefs_;
00380         char *filename_;
00381         char *vendor_, *application_;
00382         Root root_type_;
00383     public:
00384         RootNode( Fl_Preferences *, Root root, const char *vendor, const char *application );
00385         RootNode( Fl_Preferences *, const char *path, const char *vendor, const char *application, Root
00386         flags );
00387         RootNode( Fl_Preferences * );
00388         ~RootNode();
00389         int read();
00390         int write();
00391         char getPath( char *path, int pathlen );
00392         char *filename() { return filename_; }
00393         Root root() { return root_type_; }
00394     };
00395     friend class RootNode;
00396

```

```

00394 protected:
00395     Node *node;
00396     RootNode *rootNode;
00397 };
00398
00399 #endif // !Fl_Preferences_H

```

12.110 Fl_Printer.H File Reference

declaration of class [Fl_Printer](#).

```
#include <FL/Fl_Paged_Device.H>
```

Classes

- class [Fl_Printer](#)
OS-independent print support.

12.110.1 Detailed Description

declaration of class [Fl_Printer](#).

12.111 Fl_Printer.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Printing support for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2010-2016 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00021 #ifndef Fl_Printer_H
00022 #define Fl_Printer_H
00023
00024 #include <FL/Fl_Paged_Device.H>
00025
00101 class FL_EXPORT Fl_Printer : public Fl_Paged_Device {
00102 private:
00103     Fl_Paged_Device *printer;
00105     static Fl_Paged_Device* newPrinterDriver(void);
00106 public:
00108     Fl_Printer(void);
00109     int begin_job(int pagecount = 0, int *frompage = NULL, int *topage = NULL, char **perr_message =
NULL) FL_OVERRIDE;
00110     int begin_page(void) FL_OVERRIDE;
00111     int printable_rect(int *w, int *h) FL_OVERRIDE;
00112     void margins(int *left, int *top, int *right, int *bottom) FL_OVERRIDE;
00113     void origin(int *x, int *y) FL_OVERRIDE;
00114     void origin(int x, int y) FL_OVERRIDE;
00115     void scale(float scale_x, float scale_y = 0.) FL_OVERRIDE;
00116     void rotate(float angle) FL_OVERRIDE;
00117     void translate(int x, int y) FL_OVERRIDE;
00118     void untranslate(void) FL_OVERRIDE;
00119     int end_page(void) FL_OVERRIDE;
00120     void end_job(void) FL_OVERRIDE;
00121     void set_current(void) FL_OVERRIDE;
00122     bool is_current() FL_OVERRIDE;
00123
00127     static const char *dialog_title;
00128     static const char *dialog_printer;
00129     static const char *dialog_range;
00130     static const char *dialog_copies;
00131     static const char *dialog_all;
00132     static const char *dialog_pages;
00133     static const char *dialog_from;

```



```

00134 static const char *dialog_to;
00135 static const char *dialog_properties;
00136 static const char *dialog_copyNo;
00137 static const char *dialog_print_button;
00138 static const char *dialog_cancel_button;
00139 static const char *dialog_print_to_file;
00140 static const char *property_title;
00141 static const char *property_pagesize;
00142 static const char *property_mode;
00143 static const char *property_use;
00144 static const char *property_save;
00145 static const char *property_cancel;
00146 ~Fl_Printer(void);
00147 };
00148
00149 #endif // Fl_Printer_H

```

12.112 Fl_Progress.H

```

00001 //
00002 // Progress bar widget definitions.
00003 //
00004 // Copyright 2000-2010 by Michael Sweet.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Progress widget . */
00019
00020 #ifndef _Fl_Progress_H_
00021 # define _Fl_Progress_H_
00022
00023 //
00024 // Include necessary headers.
00025 //
00026 #include "Fl_Widget.H"
00027
00028
00029 //
00030 // Progress class...
00031 //
00032
00033 class FL_EXPORT Fl_Progress : public Fl_Widget {
00034
00035     float value_,
00036           minimum_,
00037           maximum_;
00038
00039     protected:
00040
00041     void draw() FL_OVERRIDE;
00042
00043     public:
00044
00045     Fl_Progress(int x, int y, int w, int h, const char *l = 0);
00046
00047     void maximum(float v) { maximum_ = v; redraw(); }
00048     float maximum() const { return (maximum_); }
00049
00050     void minimum(float v) { minimum_ = v; redraw(); }
00051     float minimum() const { return (minimum_); }
00052
00053     void value(float v) { value_ = v; redraw(); }
00054     float value() const { return (value_); }
00055 };
00056
00057 #endif // !_Fl_Progress_H_

```

12.113 Fl_Radio_Button.H

```

00001 //
00002 // Radio button header file for the Fast Light Tool Kit (FLTK).
00003 //

```

```
00004 // Copyright 1998-2014 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00017 /* \file
00018     Fl_Radio_Button widget . */
00019
00020 #ifndef Fl_Radio_Button_H
00021 #define Fl_Radio_Button_H
00022
00023 #include "Fl_Button.H"
00024
00025 class FL_EXPORT Fl_Radio_Button : public Fl_Button {
00026 public:
00027     Fl_Radio_Button(int X,int Y,int W,int H,const char *L=0);
00028 };
00029
00030 #endif
```

12.114 Fl_Radio_Light_Button.H

```
00001 //
00002 // Radio light button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2014 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00017 /* \file
00018     Fl_Radio_Light_Button widget . */
00019
00020 #ifndef Fl_Radio_Light_Button_H
00021 #define Fl_Radio_Light_Button_H
00022
00023 #include "Fl_Light_Button.H"
00024
00025 class FL_EXPORT Fl_Radio_Light_Button : public Fl_Light_Button {
00026 public:
00027     Fl_Radio_Light_Button(int X,int Y,int W,int H,const char *L=0);
00028 };
00029
00030 #endif
```

12.115 Fl_Radio_Round_Button.H

```
00001 //
00002 // Radio round button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2014 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00017 /* \file
00018     Fl_Radio_Round_Button widget . */
00019
```

```

00020 #ifndef Fl_Round_Button_H
00021 #define Fl_Round_Button_H
00022
00023 #include "Fl_Round_Button.H"
00024
00025 class FL_EXPORT Fl_Round_Button : public Fl_Round_Button {
00026 public:
00027     Fl_Round_Button(int X,int Y,int W,int H,const char *L=0);
00028 };
00029
00030 #endif

```

12.116 Fl_Rect.H

```

00001 //
00002 // Fl_Rect header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Rect_H
00018 #define Fl_Rect_H
00019
00020 #include <FL/Fl_Widget.H> // for c'tor based on Fl_Widget
00021
00030 class FL_EXPORT Fl_Rect {
00031
00032     int x_;
00033     int y_;
00034     int w_;
00035     int h_;
00036
00037 public:
00038
00040     Fl_Rect()
00041         : x_(0), y_(0), w_(0), h_(0) {}
00042
00045     Fl_Rect(int W, int H)
00046         : x_(0), y_(0), w_(W), h_(H) {}
00047
00050     Fl_Rect(int X, int Y, int W, int H)
00051         : x_(X), y_(Y), w_(W), h_(H) {}
00052
00059     Fl_Rect(int X, int Y, int W, int H, Fl_Boxtype bt)
00060         : x_(X), y_(Y), w_(W), h_(H) {
00061         inset(bt);
00062     }
00063
00065     Fl_Rect (const Fl_Widget& widget)
00066         : x_(widget.x()), y_(widget.y()), w_(widget.w()), h_(widget.h()) {}
00067
00069     Fl_Rect (const Fl_Widget* const widget)
00070         : x_(widget->x()), y_(widget->y()), w_(widget->w()), h_(widget->h()) {}
00071
00072     int x() const { return x_; }
00073     int y() const { return y_; }
00074     int w() const { return w_; }
00075     int h() const { return h_; }
00076
00080     int r() const { return x_ + w_; }
00084     int b() const { return y_ + h_; }
00085
00086     void x(int X) { x_ = X; }
00087     void y(int Y) { y_ = Y; }
00088     void w(int W) { w_ = W; }
00089     void h(int H) { h_ = H; }
00090
00091     void r(int R) { w_ = R - x_; }
00092     void b(int B) { h_ = B - y_; }
00093
00104     void inset(int d) {
00105         x_ += d;
00106         y_ += d;
00107         w_ -= 2 * d;
00108         h_ -= 2 * d;

```

```

00109     }
00110
00125     void inset(Fl_Boxtype bt) {
00126         x_ += Fl::box_dx(bt);
00127         y_ += Fl::box_dy(bt);
00128         w_ -= Fl::box_dw(bt);
00129         h_ -= Fl::box_dh(bt);
00130     }
00131
00142     void inset(int left, int top, int right, int bottom) {
00143         x_ += left;
00144         y_ += top;
00145         w_ -= (left + right);
00146         h_ -= (top + bottom);
00147     }
00148
00149     friend bool operator==(const Fl_Rect& lhs, const Fl_Rect& rhs) {
00150         return (lhs.x_==rhs.x_) && (lhs.y_==rhs.y_) && (lhs.w_==rhs.w_) && (lhs.h_==rhs.h_);
00151     }
00152
00153     friend bool operator!=(const Fl_Rect& lhs, const Fl_Rect& rhs) {
00154         return !(lhs==rhs);
00155     }
00156
00157 }; // class Fl_Rect
00158
00159 #endif // Fl_Rect_H

```

12.117 Fl_Repeat_Button.H

```

00001 //
00002 // Repeat button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Repeat_Button widget . */
00019
00020 #ifndef Fl_Repeat_Button_H
00021 #define Fl_Repeat_Button_H
00022 #include "Fl.H"
00023 #include "Fl_Button.H"
00024
00031 class FL_EXPORT Fl_Repeat_Button : public Fl_Button {
00032     static void repeat_callback(void *);
00033 public:
00034     int handle(int) FL_OVERRIDE;
00040     Fl_Repeat_Button(int X,int Y,int W,int H,const char *l=0);
00041
00042     void deactivate() {
00043         Fl::remove_timeout(repeat_callback,this);
00044         Fl_Button::deactivate();
00045     }
00046 };
00047
00048 #endif

```

12.118 Fl_Return_Button.H

```

00001 //
00002 // Return button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:

```

```

00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Return_Button widget . */
00019
00020 #ifndef Fl_Return_Button_H
00021 #define Fl_Return_Button_H
00022 #include "Fl_Button.H"
00023
00031 class FL_EXPORT Fl_Return_Button : public Fl_Button {
00032 protected:
00033     void draw() FL_OVERRIDE;
00034 public:
00035     int handle(int) FL_OVERRIDE;
00041     Fl_Return_Button(int X, int Y, int W, int H, const char *l=0);
00042 };
00043
00044 #endif

```

12.119 Fl_RGB_Image.H

```

00001 //
00002 // RGB Image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_RGB_Image_H
00018 # define Fl_RGB_Image_H
00019 # include "Fl_Image.H"
00020 #endif // !Fl_RGB_Image_H

```

12.120 Fl_Roller.H

```

00001 //
00002 // Roller header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Roller widget . */
00019
00020 #ifndef Fl_Roller_H
00021 #define Fl_Roller_H
00022
00023 #ifndef Fl_Valuator_H
00024 #include "Fl_Valuator.H"
00025 #endif
00026
00038 class FL_EXPORT Fl_Roller : public Fl_Valuator {
00039 protected:
00040     void draw() FL_OVERRIDE;
00041 public:
00042     int handle(int) FL_OVERRIDE;
00043     Fl_Roller(int X, int Y, int W, int H, const char* L=0);
00044 };
00045
00046 #endif

```

12.121 Fl_Round_Button.H

```

00001 //
00002 // Round button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Round_Button_H
00018 #define Fl_Round_Button_H
00019
00020 #include "Fl_Light_Button.H"
00021
00034 class FL_EXPORT Fl_Round_Button : public Fl_Light_Button {
00035 public:
00036     Fl_Round_Button(int x,int y,int w,int h,const char *l = 0);
00037 };
00038
00039 #endif

```

12.122 Fl_Round_Clock.H

```

00001 //
00002 // Round clock header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Round_Clock widget . */
00019
00020 #ifndef Fl_Round_Clock_H
00021 #define Fl_Round_Clock_H
00022
00023 #include "Fl_Clock.H"
00024
00026 class FL_EXPORT Fl_Round_Clock : public Fl_Clock {
00027 public:
00029     Fl_Round_Clock(int X,int Y,int W,int H, const char *L = 0);
00030 };
00031
00032 #endif

```

12.123 Fl_Scheme.H

```

00001 //
00002 // Scheme header for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2022-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef FL_Fl_Scheme_H_

```

```

00018 #define FL_Fl_Scheme_H_
00019
00020 #include <FL/Fl.H>
00021
00022 class Fl_Scheme {
00023
00024 private:
00025
00026     static const char **names_;           // registered scheme names
00027     static int num_schemes_;              // number of registered schemes
00028     static int alloc_size_;              // number of allocated scheme name entries
00029
00030 protected:
00031
00032     // const char *name_;                 // the scheme's name
00033
00034     // protected constructor - not yet implemented
00035     // Fl_Scheme(const char *name);
00036
00037 public:
00038
00039     // Static methods.
00040
00041     // Some of these methods will replace the scheme related methods of class Fl,
00042     // for instance Fl::scheme() and Fl::is_scheme().
00043     // Backwards compatibility must be kept though.
00044
00045     static const char **names();
00046
00052     static int num_schemes() {
00053         if (!names_) names(); // force initialization
00054         return num_schemes_;
00055     }
00056
00057     // Adding a scheme name must be a public static method in FLTK 1.4.0.
00058     // This will later be protected or replaced by another method name.
00059
00060     static int add_scheme_name(const char *name);
00061
00062 }; // class Fl_Scheme
00063
00064 #endif // FL_Fl_Scheme_H_

```

12.124 Fl_Scheme_Choice.H

```

00001 //
00002 // Scheme Choice header for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2022-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef FL_Fl_Scheme_Choice_H_
00018 #define FL_Fl_Scheme_Choice_H_
00019
00020 #include <FL/Fl.H>
00021 #include <FL/Fl_Scheme.H>
00022 #include <FL/Fl_Choice.H>
00023
00024 class FL_EXPORT Fl_Scheme_Choice : public Fl_Choice {
00025
00026 protected:
00027     static void scheme_cb_(Fl_Widget *w, void *);
00028
00029 public:
00030     Fl_Scheme_Choice(int X, int Y, int W, int H, const char *L = 0);
00031     int handle(int event) FL_OVERRIDE;
00032
00033     // set the current value according to the active scheme
00034     virtual void init_value();
00035
00036 }; // class Fl_Scheme_Choice
00037
00038 #endif // FL_Fl_Scheme_Choice_H_

```

12.125 Fl_Scroll.H

```

00001 //
00002 // Fl_Scroll header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Scroll widget . */
00019
00020 #ifndef Fl_Scroll_H
00021 #define Fl_Scroll_H
00022
00023 #include "Fl_Group.H"
00024 #include "Fl_Scrollbar.H"
00025
00098 class FL_EXPORT Fl_Scroll : public Fl_Group {
00099
00100     int xposition_, yposition_;
00101     int oldx, oldy;
00102     int scrollbar_size_;
00103     static void hscrollbar_cb(Fl_Widget*, void*);
00104     static void scrollbar_cb(Fl_Widget*, void*);
00105     static void draw_clip(void*,int,int,int,int);
00106
00107 protected:      // (STR#1895)
00108
00110     typedef struct { int x,y,w,h; } Fl_Region_XYWH;
00111
00113     typedef struct {
00114         int l;
00115         int r;
00116         int t;
00117         int b;
00118     } Fl_Region_LRTB;
00119
00121     typedef struct {
00122         int x,y,w,h;
00123         int pos;
00124         int size;
00125         int first;
00126         int total;
00127     } Fl_Scrollbar_Data;
00128
00135     typedef struct {
00136         int scrollbar;
00137         Fl_Region_XYWH innerbox;
00138         Fl_Region_XYWH innerchild;
00139         Fl_Region_LRTB child;
00140         int hneeded;
00141         int vneeded;
00142         Fl_Scrollbar_Data hscroll;
00143         Fl_Scrollbar_Data vscroll;
00144     } ScrollInfo;
00145     void recalc_scrollbars(ScrollInfo &si) const;
00146
00147 protected:
00148
00149     int on_insert(Fl_Widget*, int) FL_OVERRIDE;
00150     int on_move(int, int) FL_OVERRIDE;
00151     void fix_scrollbar_order();
00152     void bbox(int&,int&,int&,int&) const;
00153     void draw() FL_OVERRIDE;
00154
00155 public:
00156
00157     Fl_Scrollbar scrollbar;
00158     Fl_Scrollbar hscrollbar;
00159
00160     void resize(int X, int Y, int W, int H) FL_OVERRIDE;
00161     int handle(int) FL_OVERRIDE;
00162
00163     Fl_Scroll(int X, int Y, int W, int H, const char *L = 0);
00164     virtual ~Fl_Scroll();
00165
00166     enum { // values for type()

```



```

00167     HORIZONTAL = 1,
00168     VERTICAL = 2,
00169     BOTH = 3,
00170     ALWAYS_ON = 4,
00171     HORIZONTAL_ALWAYS = 5,
00172     VERTICAL_ALWAYS = 6,
00173     BOTH_ALWAYS = 7
00174 };
00175
00177 int xposition() const {return xposition_;}
00179 int yposition() const {return yposition_;}
00180 void scroll_to(int, int);
00181 void clear();
00182
00183 /* delete child n (by index) */
00184 int delete_child(int n) FL_OVERRIDE;
00185
00195 int scrollbar_size() const {
00196     return(scrollbar_size_);
00197 }
00217 void scrollbar_size(int newSize) {
00218     if ( newSize != scrollbar_size_ ) redraw();
00219     scrollbar_size_ = newSize;
00220 }
00221 };
00222
00223 #endif

```

12.126 Fl_Scrollbar.H

```

00001 //
00002 // Scroll bar header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Scrollbar widget . */
00019
00020 #ifndef Fl_Scrollbar_H
00021 #define Fl_Scrollbar_H
00022
00023 #include "Fl_Slider.H"
00024
00041 class FL_EXPORT Fl_Scrollbar : public Fl_Slider {
00042
00043     int linesize_;
00044     int pushed_;
00045     static void timeout_cb(void*);
00046     void increment_cb();
00047 protected:
00048     void draw() FL_OVERRIDE;
00049
00050 public:
00051
00052     Fl_Scrollbar(int X,int Y,int W,int H, const char *L = 0);
00053     ~Fl_Scrollbar();
00054     int handle(int) FL_OVERRIDE;
00055
00063     int value() const {return int(Fl_Slider::value());}
00064
00071     int value(int p) {return int(Fl_Slider::value((double)p));}
00072
00087     int value(int pos, int windowSize, int first_line, int total_lines) {
00088         return scrollvalue(pos, windowSize, first_line, total_lines);
00089     }
00090
00094     int linesize() const {return linesize_;}
00095
00101     void linesize(int i) {linesize_ = i;}
00102
00103 };
00104
00105 #endif

```

12.127 Fl_Secret_Input.H

```

00001 //
00002 // Secret input header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2011 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Secret_Input widget . */
00019
00020 #ifndef Fl_Secret_Input_H
00021 #define Fl_Secret_Input_H
00022
00023 #include "Fl_Input.H"
00024
00025 class FL_EXPORT Fl_Secret_Input : public Fl_Input {
00026 public:
00027     Fl_Secret_Input(int X,int Y,int W,int H,const char *l = 0);
00028     int handle(int) FL_OVERRIDE;
00029 };
00030
00031 #endif

```

12.128 Fl_Select_Browser.H

```

00001 //
00002 // Select browser header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Select_Browser widget . */
00019
00020 #ifndef Fl_Select_Browser_H
00021 #define Fl_Select_Browser_H
00022
00023 #include "Fl_Browser.H"
00024
00025 class FL_EXPORT Fl_Select_Browser : public Fl_Browser {
00026 public:
00027     Fl_Select_Browser(int X,int Y,int W,int H,const char *L=0);
00028 };
00029
00030 #endif

```

12.129 Fl_Shared_Image.H File Reference

[Fl_Shared_Image](#) class.

```
#include "Fl_Image.H"
```

Classes

- class [Fl_Shared_Image](#)

This class supports caching, loading, and drawing of image files.

Typedefs

- typedef [FI_Image](#) *(* [FI_Shared_Handler](#)) (const char *name, [uchar](#) *header, int headerlen)
Test function (typedef) for adding new shared image formats.

Functions

- void [fl_register_images](#) ()
Register the known image formats.

12.129.1 Detailed Description

[FI_Shared_Image](#) class.

12.129.2 Typedef Documentation

12.129.2.1 FI_Shared_Handler

```
typedef FI\_Image *(* FI\_Shared\_Handler) (const char *name, uchar *header, int headerlen)
```

Test function (typedef) for adding new shared image formats.

This defines the function type you can use to add a handler for unknown image formats that can be opened and loaded as an [FI_Shared_Image](#).

[fl_register_images\(\)](#) adds all image formats known to FLTK. Call [FI_Shared_Image::add_handler\(\)](#) to add your own check function to the list of known image formats.

Your function will be passed the filename (*name*), some *header* bytes already read from the image file and the size *headerlen* of the data read. The max value of size is implementation dependent. If your handler function needs to check more bytes you must open the image file yourself.

The provided buffer *header* must not be overwritten.

If your handler function can identify the file type you must open the file and return a valid [FI_Image](#) or derived type, otherwise you must return NULL. Example:

```
static FI\_Image *check_my_image(const char *name,
                                uchar *header,
                                int headerlen) {
    // (test image type using header and headerlen)
    if (known) {
        // (load image data from file \p name)
        return new FI\_RGB\_Image(data, ...);
    } else
        return 0;
}
// add your handler:
FI\_Shared\_Image::add\_handler(check_my_image);
```

Parameters

in	<i>name</i>	filename to be checked and opened if applicable
in	<i>header</i>	portion of the file that has already been read
in	<i>headerlen</i>	length of provided <i>header</i> data

Returns

valid [FI_Image](#) or NULL.

See also

[FI_Shared_Image::add_handler\(\)](#)

12.129.3 Function Documentation

12.129.3.1 fl_register_images()

```
void fl_register_images () [extern]
```

Register the known image formats.

This function is provided in the `fltk_images` library and registers all of the "extra" image file formats known to FLTK that are not part of the core FLTK library.

You may add your own image formats with `Fl_Shared_Image::add_handler()`.

12.130 Fl_Shared_Image.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Shared image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00020 #ifndef Fl_Shared_Image_H
00021 # define Fl_Shared_Image_H
00022
00023 # include "Fl_Image.H"
00024
00025 #undef SHIM_DEBUG
00026
00071 typedef Fl_Image *(*Fl_Shared_Handler)(const char *name,
00072                                         uchar *header,
00073                                         int headerlen);
00074
00094 class FL_EXPORT Fl_Shared_Image : public Fl_Image {
00095
00096     friend class Fl_JPEG_Image;
00097     friend class Fl_PNG_Image;
00098     friend class Fl_SVG_Image;
00099     friend class Fl_Graphics_Driver;
00100
00101 protected:
00102
00103     static Fl_Shared_Image **images_; // Shared images
00104     static int num_images_; // Number of shared images
00105     static int alloc_images_; // Allocated shared images
00106     static Fl_Shared_Handler *handlers_; // Additional format handlers
00107     static int num_handlers_; // Number of format handlers
00108     static int alloc_handlers_; // Allocated format handlers
00109
00110     const char *name_; // Name of image file
00111     int original_; // Original image?
00112     int refcount_; // Number of times this image has been used
00113     Fl_Image *image_; // The image that is shared
00114     int alloc_image_; // Was the image allocated?
00115
00116     static int compare(Fl_Shared_Image **i0, Fl_Shared_Image **i1);
00117
00118     // Use get() and release() to load/delete images in memory...
00119     Fl_Shared_Image();
00120     Fl_Shared_Image(const char *n, Fl_Image *img = 0);
00121     virtual ~Fl_Shared_Image();
00122     void add();
00123     void update();
00124     Fl_Shared_Image *copy_(int W, int H) const;
00125
00126 public:
00127     #ifdef SHIM_DEBUG
00128         static void print_pool();
00129     #endif
00130
00132     const char *name() { return name_; }
00133
00137     int refcount() { return refcount_; }
00138
00146     int original() { return original_; }
00147
00148     void release() FL_OVERRIDE;
00149     virtual void reload();
00150
00151     Fl_Shared_Image *as_shared_image() FL_OVERRIDE {
00152         return this;
00153     }

```

```

00154
00155     Fl_Image *copy(int W, int H) const FL_OVERRIDE;
00156     Fl_Image *copy() const;
00157     Fl_Image *copy();
00158
00159     void color_average(Fl_Color c, float i) FL_OVERRIDE;
00160     void desaturate() FL_OVERRIDE;
00161     void draw(int X, int Y, int W, int H, int cx = 0, int cy = 0) FL_OVERRIDE;
00162     void draw(int X, int Y) { draw(X, Y, w(), h(), 0, 0); }
00163     void uncache() FL_OVERRIDE;
00164
00165     static Fl_Shared_Image *find(const char *name, int W = 0, int H = 0);
00166     static Fl_Shared_Image *get(const char *name, int W = 0, int H = 0);
00167     static Fl_Shared_Image *get(Fl_RGB_Image *rgb, int own_it = 1);
00168     static Fl_Shared_Image **images();
00169     static int          num_images();
00170     static void         add_handler(Fl_Shared_Handler f);
00171     static void         remove_handler(Fl_Shared_Handler f);
00172
00194     const Fl_Image *image() const { return image_; }
00195
00196 }; // class Fl_Shared_Image
00197
00198 //
00199 // The following function is provided in the fltk_images library and
00200 // registers all of the "extra" image file formats that are not part
00201 // of the core FLTK library...
00202 //
00203
00204 FL_EXPORT extern void fl_register_images();
00205
00206 #endif // !Fl_Shared_Image_H

```

12.131 Fl_Shortcut_Button.H

```

00001 //
00002 // Shortcut Button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Shortcut_Button_H
00018 #define Fl_Shortcut_Button_H
00019
00020 #include <FL/Fl_Button.H>
00021
00022 class FL_EXPORT Fl_Shortcut_Button : public Fl_Button {
00023 private:
00024     bool hot_, pre_hot_, default_set_, handle_default_button_;
00025     Fl_Shortcut pre_esc_;
00026     Fl_Shortcut default_shortcut_;
00027 protected:
00028     Fl_Shortcut shortcut_value_;
00029     void do_end_hot_callback();
00030     int handle(int) FL_OVERRIDE;
00031     void draw() FL_OVERRIDE;
00032 public:
00033     Fl_Shortcut_Button(int X,int Y,int W,int H, const char* l = 0);
00034     void value(Fl_Shortcut shortcut);
00035     Fl_Shortcut value();
00036     #if 0
00037     // Default shortcut settings are disabled until successful review of the UI
00038     void default_value(Fl_Shortcut shortcut);
00039     Fl_Shortcut default_value();
00040     void default_clear();
00041     #endif
00042 };
00043
00044 #endif // Fl_Shortcut_Button_H
00045

```

12.132 fl_show_colormap.H File Reference

The `fl_show_colormap()` function hides the implementation classes used to provide the popup window and color selection mechanism.

Functions

- [Fl_Color fl_show_colormap \(Fl_Color oldcol\)](#)
Pops up a window to let the user pick a colormap entry.

12.132.1 Detailed Description

The `fl_show_colormap()` function hides the implementation classes used to provide the popup window and color selection mechanism.

12.133 fl_show_colormap.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Colormap picker header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00022 #ifndef fl_show_colormap_H
00023 #define fl_show_colormap_H
00024
00025 /* doxygen comment here to avoid exposing ColorMenu in fl_show_colormap.cxx
00026 */
00027
00039 FL_EXPORT Fl_Color fl_show_colormap(Fl_Color oldcol);
00040
00043 #endif
```

12.134 fl_show_input.H

```
00001 //
00002 // Standard input dialog header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #include "fl_ask.H"
```

12.135 Fl_Simple_Counter.H

```
00001 //
00002 // Simple counter header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
```

```

00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Simple_Counter widget . */
00019
00020 #ifndef Fl_Simple_Counter_H
00021 #define Fl_Simple_Counter_H
00022
00023 #include "Fl_Counter.H"
00029 class FL_EXPORT Fl_Simple_Counter : public Fl_Counter {
00030 public:
00031     Fl_Simple_Counter(int X,int Y,int W,int H, const char *L = 0);
00032 };
00033
00034 #endif

```

12.136 Fl_Single_Window.H

```

00001 //
00002 // Single-buffered window header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Single_Window class . */
00019
00020 #ifndef Fl_Single_Window_H
00021 #define Fl_Single_Window_H
00022
00023 #include "Fl_Window.H"
00024
00032 class FL_EXPORT Fl_Single_Window : public Fl_Window {
00033 public:
00034     void show() FL_OVERRIDE;
00036     void show(int argc, char **argv) { Fl_Window::show(argc, argv);}
00037
00042     Fl_Single_Window(int W, int H, const char *l=0);
00043
00048     Fl_Single_Window(int X, int Y, int W, int H, const char *l=0);
00049
00053     void make_current() { Fl_Window::make_current(); }
00054
00058     void flush() FL_OVERRIDE { Fl_Window::flush(); }
00059 };
00060
00061 #endif

```

12.137 Fl_Slider.H

```

00001 //
00002 // Slider header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php

```

```

00015 //
00016
00017 /* \file
00018     Fl_Slider widget . */
00019
00020 #ifndef Fl_Slider_H
00021 #define Fl_Slider_H
00022
00023 #ifndef Fl_Valuator_H
00024 #include "Fl_Valuator.H"
00025 #endif
00026
00027 // values for type(), lowest bit indicate horizontal:
00028 #define FL_VERT_SLIDER      0
00029 #define FL_HOR_SLIDER      1
00030 #define FL_VERT_FILL_SLIDER 2
00031 #define FL_HOR_FILL_SLIDER 3
00032 #define FL_VERT_NICE_SLIDER 4
00033 #define FL_HOR_NICE_SLIDER 5
00034
00059 class FL_EXPORT Fl_Slider : public Fl_Valuator {
00060
00061     float slider_size_;
00062     uchar slider_;
00063     void _Fl_Slider();
00064     void draw_bg(int, int, int, int);
00065
00066 protected:
00067
00068     // these allow subclasses to put the slider in a smaller area:
00069     void draw(int, int, int, int);
00070     int handle(int, int, int, int, int);
00071     void draw() FL_OVERRIDE;
00072
00073 public:
00074
00075     int handle(int) FL_OVERRIDE;
00076     Fl_Slider(int X,int Y,int W,int H, const char *L = 0);
00077     Fl_Slider(uchar t,int X,int Y,int W,int H, const char *L);
00078
00079     int scrollvalue(int pos,int size,int first,int total);
00080     void bounds(double a, double b);
00081
00085     float slider_size() const {return slider_size_;}
00086
00096     void slider_size(double v);
00097
00099     Fl_Boxtype slider() const {return (Fl_Boxtype)slider_;}
00100
00102     void slider(Fl_Boxtype c) {slider_ = c;}
00103 };
00104
00105 #endif

```

12.138 Fl_Spinner.H

```

00001 //
00002 // Spinner widget for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Spinner widget . */
00019
00020 #ifndef Fl_Spinner_H
00021 #define Fl_Spinner_H
00022
00023 #include <FL/Enumerations.H>
00024 #include <FL/Fl_Group.H>
00025 #include <FL/Fl_Input.H>
00026 #include <FL/Fl_Repeat_Button.H>
00027
00037 class FL_EXPORT Fl_Spinner : public Fl_Group {
00038

```



```

00039     double        value_;           // Current value
00040     double        minimum_;         // Minimum value
00041     double        maximum_;         // Maximum value
00042     double        step_;            // Amount to add/subtract for up/down
00043     const char    *format_;         // Format string for input field
00044     int           wrap_;            // wrap around at bounds (1/0)
00045
00046 private:
00047
00048     static void sb_cb(Fl_Widget *w, Fl_Spinner *sb); // internal callback
00049     void update(); // update input field
00050
00051 protected:
00052
00053     // This class works like Fl_Input but ignores FL_Up and FL_Down key
00054     // presses so they are handled by its parent, the Fl_Spinner widget.
00055     // See STR #2989.
00056
00057     class FL_EXPORT Fl_Spinner_Input : public Fl_Input {
00058     public:
00059         Fl_Spinner_Input(int X, int Y, int W, int H)
00060             : Fl_Input(X, Y, W, H) {}
00061         int handle(int event) FL_OVERRIDE; // implemented in src/Fl_Spinner.cxx
00062     };
00063
00064     Fl_Spinner_Input input_;         // Input field for the value
00065     Fl_Repeat_Button
00066         up_button_,                 // Up button
00067         down_button_;               // Down button
00068
00069     void draw() FL_OVERRIDE;
00070
00071 public:
00072
00073     // Constructor
00074     Fl_Spinner(int X, int Y, int W, int H, const char *L = 0);
00075     // Event handling
00076     int handle(int event) FL_OVERRIDE;
00077     // Resize group and subwidgets
00078     void resize(int X, int Y, int W, int H) FL_OVERRIDE;
00079
00080     const char *format() const { return (format_); }
00081
00082     void format(const char *f) { format_ = f; update(); }
00083
00084     double maximum() const { return (maximum_); }
00085
00086     void maximum(double m) { maximum_ = m; }
00087
00088     double minimum() const { return (minimum_); }
00089
00090     void minimum(double m) { minimum_ = m; }
00091
00092     void range(double a, double b) { minimum_ = a; maximum_ = b; }
00093
00094     // Sets the amount to change the value when the user clicks a button.
00095     // Docs in src/Fl_Spinner.cxx
00096     void step(double s);
00097
00098     double step() const { return (step_); }
00099
00100     void wrap(int set) { wrap_ = set ? 1 : 0; }
00101
00102     int wrap() const { return wrap_; }
00103
00104     Fl_Color textcolor() const { return (input_.textcolor()); }
00105
00106     void textcolor(Fl_Color c) { input_.textcolor(c); }
00107
00108     Fl_Font textfont() const { return (input_.textfont()); }
00109
00110     void textfont(Fl_Font f) { input_.textfont(f); }
00111
00112     Fl_Fonsize textsize() const { return (input_.textsize()); }
00113
00114     void textsize(Fl_Fonsize s) { input_.textsize(s); }
00115
00116     // Sets the numeric representation in the input field.
00117     // Docs see src/Fl_Spinner.cxx
00118     void type(uchar v);
00119
00120     uchar type() const { return (input_.type()); }
00121
00122     double value() const { return (value_); }
00123
00124     void value(double v) { value_ = v; update(); }
00125

```

```

00181 void color(Fl_Color v) { input_.color(v); }
00182
00186 Fl_Color color() const { return(input_.color()); }
00187
00191 void selection_color(Fl_Color val) { input_.selection_color(val); }
00192
00196 Fl_Color selection_color() const { return input_.selection_color(); }
00197
00201 void maximum_size(int m) { if (m > 0) input_.maximum_size(m); }
00202
00206 int maximum_size() const { return input_.maximum_size(); }
00207 };
00208
00209 #endif // !Fl_Spinner_H

```

12.139 fl_string_functions.h File Reference

Public header for FLTK's platform-agnostic string handling.

```

#include "Fl_Export.H"
#include <stddef.h>

```

Functions

- char * [fl_strdup](#) (const char *s)
Cross platform interface to POSIX function strdup().
- size_t [fl_strlcpy](#) (char *, const char *, size_t)

12.139.1 Detailed Description

Public header for FLTK's platform-agnostic string handling.

12.140 fl_string_functions.h

[Go to the documentation of this file.](#)

```

00001 /*
00002  * Platform agnostic string portability functions for the Fast Light Tool Kit (FLTK).
00003  *
00004  * Copyright 2020-2022 by Bill Spitzak and others.
00005  *
00006  * This library is free software. Distribution and use rights are outlined in
00007  * the file "COPYING" which should have been included with this file. If this
00008  * file is missing or damaged, see the license at:
00009  *
00010  *     https://www.fltk.org/COPYING.php
00011  *
00012  * Please see the following page on how to report bugs and issues:
00013  *
00014  *     https://www.fltk.org/bugs.php
00015  */
00016
00022 #ifndef _FL_fl_string_functions_h_
00023 #define _FL_fl_string_functions_h_
00024
00025 #include "Fl_Export.H"
00026
00027 #ifdef __cplusplus
00028 extern "C" {
00029 #endif
00030
00031 #include <stddef.h> // size_t
00032
00038 FL_EXPORT char* fl_strdup(const char *s);
00039
00040 FL_EXPORT size_t fl_strlcpy(char *, const char *, size_t);
00041
00044 #ifdef __cplusplus
00045 }
00046 #endif /* __cplusplus */
00047
00048
00049 #endif /* _FL_fl_string_functions_h_ */

```

12.141 Fl_SVG_File_Surface.H

```

00001 //
00002 // Declaration of Fl_SVG_File_Surface in the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2020 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_SVG_File_Surface_H
00018 #define Fl_SVG_File_Surface_H
00019
00020 #include <FL/Fl_Widget_Surface.H>
00021 #include <stdio.h>
00022
00047 class FL_EXPORT Fl_SVG_File_Surface : public Fl_Widget_Surface {
00048     int width_, height_;
00049     int (*closef_)(FILE*);
00050 public:
00060     Fl_SVG_File_Surface(int width, int height, FILE *svg, int (*closef)(FILE*) = NULL);
00065     ~Fl_SVG_File_Surface();
00067     FILE *file();
00068     void origin(int x, int y) FL_OVERRIDE;
00069     void origin(int *x, int *y) FL_OVERRIDE;
00070     void translate(int x, int y) FL_OVERRIDE;
00071     void untranslate() FL_OVERRIDE;
00072     int printable_rect(int *w, int *h) FL_OVERRIDE;
00077     int close();
00078 };
00079
00080 #endif /* Fl_SVG_File_Surface_H */

```

12.142 Fl_SVG_Image.H

```

00001 //
00002 // SVG Image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2017-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef FL_SVG_IMAGE_H
00018 #define FL_SVG_IMAGE_H
00019
00020 #include <FL/Fl_Image.H>
00021
00022 struct NSVGImage;
00023
00135 class FL_EXPORT Fl_SVG_Image : public Fl_RGB_Image {
00136 private:
00137     typedef struct {
00138         NSVGImage* svg_image;
00139         int ref_count;
00140     } counted_NSVGImage;
00141     counted_NSVGImage* counted_svg_image_;
00142     bool rasterized_;
00143     int raster_w_, raster_h_;
00144     bool to_desaturate_;
00145     Fl_Color average_color_;
00146     float average_weight_;
00147     float svg_scaling_(int W, int H);
00148     void rasterize_(int W, int H);
00149     void cache_size_(int &width, int &height) FL_OVERRIDE;
00150     void init_(const char *name, const unsigned char *filedata, size_t length);
00151     Fl_SVG_Image(const Fl_SVG_Image *source);
00152 public:

```

```

00155     bool proportional;
00156     Fl_SVG_Image(const char *filename);
00157     Fl_SVG_Image(const char *sharedname, const char *svg_data);
00158     Fl_SVG_Image(const char *sharedname, const unsigned char *svg_data, size_t length);
00159     virtual ~Fl_SVG_Image();
00160     Fl_Image *copy(int W, int H) const FL_OVERRIDE;
00161     Fl_Image *copy() const {
00162         return Fl_Image::copy();
00163     }
00164     void resize(int width, int height);
00165     void desaturate() FL_OVERRIDE;
00166     void color_average(Fl_Color c, float i) FL_OVERRIDE;
00167     void draw(int X, int Y, int W, int H, int cx = 0, int cy = 0) FL_OVERRIDE;
00168     void draw(int X, int Y) { draw(X, Y, w(), h(), 0, 0); }
00169     Fl_SVG_Image *as_svg_image() FL_OVERRIDE { return this; }
00170     void normalize() FL_OVERRIDE;
00171 };
00172
00173 #endif // FL_SVG_IMAGE_H

```

12.143 Fl_Sys_Menu_Bar.H File Reference

Definition of class [Fl_Sys_Menu_Bar](#).

```
#include <FL/Fl_Menu_Bar.H>
```

Classes

- class [Fl_Sys_Menu_Bar](#)

A class to create and modify menus that appear on macOS in the menu bar at the top of the screen.

Variables

- [Fl_Sys_Menu_Bar](#) * `fl_sys_menu_bar`

The system menu bar.

12.143.1 Detailed Description

Definition of class [Fl_Sys_Menu_Bar](#).

12.144 Fl_Sys_Menu_Bar.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // MacOS system menu bar header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2017 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00022 #ifndef Fl_Sys_Menu_Bar_H
00023 #define Fl_Sys_Menu_Bar_H
00024
00025 #include <FL/Fl_Menu_Bar.H>
00026
00027 class Fl_Sys_Menu_Bar_Driver;
00028
00095 class FL_EXPORT Fl_Sys_Menu_Bar : public Fl_Menu_Bar {
00096     static Fl_Sys_Menu_Bar_Driver *driver();
00097 protected:
00098     void draw() FL_OVERRIDE;
00099 public:
00101     typedef enum {
00102         no_window_menu = 0,

```

```

00103     tabbing_mode_none,
00104     tabbing_mode_automatic,
00105     tabbing_mode_preferred
00106 } window_menu_style_enum;
00107 Fl_Sys_Menu_Bar(int x,int y,int w,int h,const char *l=0);
00108 virtual ~Fl_Sys_Menu_Bar();
00111 const Fl_Menu_Item *menu() const {return Fl_Menu_::menu();}
00112 void menu(const Fl_Menu_Item *m);
00113 void update() FL_OVERRIDE;
00114 void play_menu(const Fl_Menu_Item *) FL_OVERRIDE;
00115 int add(const char* label, int shortcut, Fl_Callback*, void *user_data=0, int flags=0);
00119 int add(const char* label, const char* shortcut, Fl_Callback* cb, void *user_data=0, int flags=0) {
00120     return add(label, fl_old_shortcut(shortcut), cb, user_data, flags);
00121 }
00122 int add(const char* str);
00123 int insert(int index, const char* label, int shortcut, Fl_Callback *cb, void *user_data=0, int
00127 flags=0);
00127 int insert(int index, const char* label, const char* shortcut, Fl_Callback *cb, void *user_data=0,
00128 int flags=0) {
00128     return insert(index, label, fl_old_shortcut(shortcut), cb, user_data, flags);
00129 }
00130 void remove(int n);
00131 void replace(int index, const char *name);
00132 void clear();
00133 int clear_submenu(int index);
00134 void mode (int i, int fl);
00137 int mode(int i) const { return Fl_Menu_::mode(i); }
00138 void shortcut (int i, int s);
00139 void setonly (Fl_Menu_Item *item);
00140 static void about(Fl_Callback *cb, void *data);
00141
00142 static window_menu_style_enum window_menu_style();
00143 static void window_menu_style(window_menu_style_enum style);
00144 static void create_window_menu();
00145 };
00146
00149 extern Fl_Sys_Menu_Bar *fl_sys_menu_bar;
00150
00151 #endif // Fl_Sys_Menu_Bar_H

```

12.145 Fl_Table.H

```

00001 //
00002 // Fl_Table -- A table widget for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2002 by Greg Ercolano.
00005 // Copyright (c) 2004 O'ksi'D
00006 // Copyright 2023 by Bill Spitzak and others.
00007 //
00008 // This library is free software. Distribution and use rights are outlined in
00009 // the file "COPYING" which should have been included with this file. If this
00010 // file is missing or damaged, see the license at:
00011 //
00012 //     https://www.fltk.org/COPYING.php
00013 //
00014 // Please see the following page on how to report bugs and issues:
00015 //
00016 //     https://www.fltk.org/bugs.php
00017 //
00018
00019 #ifndef _FL_TABLE_H
00020 #define _FL_TABLE_H
00021
00022 #include <FL/Fl_Group.H>
00023 #include <FL/Fl_Scroll.H>
00024
00025 // EXPERIMENTAL
00026 // We use either std::vector or the private class Fl_Int_Vector
00027 // depending on the build option FLTK_OPTION_STD or --enable-use_std.
00028 // This option allows to use std::string and std::vector in FLTK 1.4.x
00029
00030 #if (FLTK_USE_STD)
00031 #include <vector>
00032 typedef std::vector<int> Fl_Int_Vector;
00033 #else
00034 class Fl_Int_Vector; // private class declared in src/Fl_Int_Vector.H
00035 #endif
00036
00121 class FL_EXPORT Fl_Table : public Fl_Group {
00122 public:
00129     enum TableContext {
00130         CONTEXT_NONE           = 0,
00131         CONTEXT_STARTPAGE      = 0x01,
00132         CONTEXT_ENDPAGE        = 0x02,
00133         CONTEXT_ROW_HEADER     = 0x04,

```

```

00134     CONTEXT_COL_HEADER = 0x08,
00135     CONTEXT_CELL       = 0x10,
00136     CONTEXT_TABLE      = 0x20,
00137     CONTEXT_RC_RESIZE  = 0x40
00138 };
00139
00140 private:
00141     int _rows, _cols;          // total rows/cols
00142     int _row_header_w;        // width of row header
00143     int _col_header_h;        // height of column header
00144     int _row_position;        // last row_position set (not necessarily == toprow!)
00145     int _col_position;        // last col_position set (not necessarily == leftcol!)
00146
00147     char _row_header;         // row header enabled?
00148     char _col_header;         // col header enabled?
00149     char _row_resize;         // row resizing enabled?
00150     char _col_resize;         // col resizing enabled?
00151     int _row_resize_min;      // row minimum resizing height (default=1)
00152     int _col_resize_min;      // col minimum resizing width (default=1)
00153
00154     // OPTIMIZATION: partial row/column redraw variables
00155     int _redraw_toprow;
00156     int _redraw_botrow;
00157     int _redraw_leftcol;
00158     int _redraw_rightcol;
00159     Fl_Color _row_header_color;
00160     Fl_Color _col_header_color;
00161
00162     int _auto_drag;
00163     int _selecting;
00164     int _scrollbar_size;
00165     enum {
00166         TABCELLNAV = 1<<0
00167     };
00168     unsigned int flags_;
00169
00170     Fl_Int_Vector *_colwidths;    // column widths in pixels
00171     Fl_Int_Vector *_rowheights;   // row heights in pixels
00172
00173     // number of columns and rows == size of corresponding vectors
00174     int col_size();               // size of the column widths vector
00175     int row_size();               // size of the row heights vector
00176
00177     Fl_Cursor _last_cursor;       // last mouse cursor before changed to 'resize' cursor
00178
00179     // EVENT CALLBACK DATA
00180     TableContext _callback_context; // event context
00181     int _callback_row, _callback_col; // event row/col
00182
00183     // handle() state variables.
00184     // Put here instead of local statics in handle(), so more
00185     // than one Fl_Table can exist without crosstalk between them.
00186     //
00187     int _resizing_col;             // column being dragged
00188     int _resizing_row;            // row being dragged
00189     int _dragging_x;              // starting x position for horiz drag
00190     int _dragging_y;              // starting y position for vert drag
00191     int _last_row;                // last row we FL_PUSH'ed
00192
00193     // Redraw single cell
00194     void _redraw_cell(TableContext context, int R, int C);
00195
00196     void _start_auto_drag();
00197     void _stop_auto_drag();
00198     void _auto_drag_cb();
00199     static void _auto_drag_cb2(void *d);
00200
00201 protected:
00202     enum ResizeFlag {
00203         RESIZE_NONE      = 0,
00204         RESIZE_COL_LEFT  = 1,
00205         RESIZE_COL_RIGHT = 2,
00206         RESIZE_ROW_ABOVE = 3,
00207         RESIZE_ROW_BELOW = 4
00208     };
00209
00210     int table_w;
00211     int table_h;
00212     int toprow;
00213     int botrow;
00214     int leftcol;
00215     int rightcol;
00216
00217     // selection
00218     int current_row;
00219     int current_col;
00220     int select_row;

```

```

00221 int select_col;
00222
00223 // OPTIMIZATION: Precomputed scroll positions for the toprow/leftcol
00224 int toprow_scrollpos;
00225 int leftcol_scrollpos;
00226
00227 // Data table's inner dimension
00228 int tix;
00229 int tiy;
00230 int tiw;
00231 int tih;
00232
00233 // Data table's outer dimension
00234 int tox;
00235 int toy;
00236 int tow;
00237 int toh;
00238
00239 // Table widget's inner dimension
00240 int wix;
00241 int wiy;
00242 int wiw;
00243 int wih;
00244
00245 Fl_Scroll *table;
00246 Fl_Scrollbar *vscrollbar;
00247 Fl_Scrollbar *hscrollbar;
00248
00249 // Fltk
00250 int handle(int e) FL_OVERRIDE; // fltk handle() FL_OVERRIDE
00251
00252 // Class maintenance
00253 void recalc_dimensions();
00254 void table_resized(); // table resized; recalc
00255 void table_scrolled(); // table scrolled; recalc
00256 void get_bounds(TableContext context, // return x/y/w/h bounds for context
00257                int &X, int &Y, int &W, int &H);
00258 void change_cursor(Fl_Cursor newcursor); // change mouse cursor to some other shape
00259 TableContext cursor2rowcol(int &R, int &C, ResizeFlag &resizeflag);
00260 int find_cell(TableContext context, // find cell's x/y/w/h given r/c
00261              int R, int C, int &X, int &Y, int &W, int &H);
00262 int row_col_clamp(TableContext context, int &R, int &C);
00263 // clamp r/c to known universe
00264
00375 virtual void draw_cell(TableContext context, int R=0, int C=0,
00376                      int X=0, int Y=0, int W=0, int H=0)
00377 { (void)context; (void)R; (void)C; (void)X; (void)Y; (void)W; (void)H; }
// overridden by deriving class
00378
00379 long row_scroll_position(int row); // find scroll position of row (in pixels)
00380 long col_scroll_position(int col); // find scroll position of col (in pixels)
00381
00382 int is_fltk_container() { // does table contain fltk widgets?
00383     return( Fl_Group::children() > 3 ); // (ie. more than box and 2 scrollbars?)
00384 }
00385
00386 static void scroll_cb(Fl_Widget*,void*); // h/v scrollbar callback
00387
00388 void damage_zone(int r1, int c1, int r2, int c2, int r3 = 0, int c3 = 0);
00389
00390 void redraw_range(int topRow, int botRow, int leftCol, int rightCol) {
00391     if ( _redraw_toprow == -1 ) {
00392         // Initialize redraw range
00393         _redraw_toprow = topRow;
00394         _redraw_botrow = botRow;
00395         _redraw_leftcol = leftCol;
00396         _redraw_rightcol = rightCol;
00397     } else {
00398         // Extend redraw range
00399         if ( topRow < _redraw_toprow ) _redraw_toprow = topRow;
00400         if ( botRow > _redraw_botrow ) _redraw_botrow = botRow;
00401         if ( leftCol < _redraw_leftcol ) _redraw_leftcol = leftCol;
00402         if ( rightCol > _redraw_rightcol ) _redraw_rightcol = rightCol;
00403     }
00404     // Indicate partial redraw needed of some cells
00405     damage(FL_DAMAGE_CHILD);
00406 }
00407
00408 // draw() has to be protected per FLTK convention (was public in 1.3.x)
00409 void draw() FL_OVERRIDE;
00410
00411 public:
00412 Fl_Table(int X, int Y, int W, int H, const char *l=0);
00413 ~Fl_Table();
00414
00415 virtual void clear() {
00416     rows(0);
00417 }

```

```

00429     cols(0);
00430     table->clear();
00431 }
00432
00433 // \todo: add topline(), middleline(), bottomline()
00434
00440 inline void table_box(Fl_Boxtype val) {
00441     table->box(val);
00442     table_resized();
00443 }
00444
00448 inline Fl_Boxtype table_box( void ) {
00449     return(table->box());
00450 }
00451
00452 virtual void rows(int val); // set number of rows
00453
00457 inline int rows() {
00458     return(_rows);
00459 }
00460
00461 virtual void cols(int val); // set number of columns
00462
00466 inline int cols() {
00467     return(_cols);
00468 }
00469
00498 inline void visible_cells(int& r1, int& r2, int& c1, int& c2) {
00499     r1 = toprow;
00500     r2 = botrow;
00501     c1 = leftcol;
00502     c2 = rightcol;
00503 }
00504
00509 int is_interactive_resize() {
00510     return(_resizing_row != -1 || _resizing_col != -1);
00511 }
00512
00516 inline int row_resize() {
00517     return(_row_resize);
00518 }
00519
00526 void row_resize(int flag) { // enable row resizing
00527     _row_resize = flag;
00528 }
00529
00533 inline int col_resize() {
00534     return(_col_resize);
00535 }
00536
00543 void col_resize(int flag) { // enable col resizing
00544     _col_resize = flag;
00545 }
00546
00550 inline int col_resize_min() { // column minimum resizing width
00551     return(_col_resize_min);
00552 }
00553
00559 void col_resize_min(int val) {
00560     _col_resize_min = ( val < 1 ) ? 1 : val;
00561 }
00562
00566 inline int row_resize_min() { // column minimum resizing width
00567     return(_row_resize_min);
00568 }
00569
00575 void row_resize_min(int val) {
00576     _row_resize_min = ( val < 1 ) ? 1 : val;
00577 }
00578
00582 inline int row_header() { // set/get row header enable flag
00583     return(_row_header);
00584 }
00585
00590 void row_header(int flag) {
00591     _row_header = flag;
00592     table_resized();
00593     redraw();
00594 }
00595
00599 inline int col_header() { // set/get col header enable flag
00600     return(_col_header);
00601 }
00602
00607 void col_header(int flag) {
00608     _col_header = flag;
00609     table_resized();

```



```

00610     redraw();
00611 }
00612
00616 inline void col_header_height(int height) { // set/get col header height
00617     _col_header_h = height;
00618     table_resized();
00619     redraw();
00620 }
00621
00625 inline int col_header_height() {
00626     return(_col_header_h);
00627 }
00628
00632 inline void row_header_width(int width) { // set/get row header width
00633     _row_header_w = width;
00634     table_resized();
00635     redraw();
00636 }
00637
00641 inline int row_header_width() {
00642     return(_row_header_w);
00643 }
00644
00648 inline void row_header_color(Fl_Color val) { // set/get row header color
00649     _row_header_color = val;
00650     redraw();
00651 }
00652
00656 inline Fl_Color row_header_color() {
00657     return(_row_header_color);
00658 }
00659
00663 inline void col_header_color(Fl_Color val) { // set/get col header color
00664     _col_header_color = val;
00665     redraw();
00666 }
00667
00671 inline Fl_Color col_header_color() {
00672     return(_col_header_color);
00673 }
00674
00675 void row_height(int row, int height); // set row height in pixels
00676
00677 // Returns the current height of the specified row as a value in pixels.
00678 int row_height(int row);
00679
00680 void col_width(int col, int width); // set a column's width in pixels
00681
00682 // Returns the current width of the specified column in pixels.
00683 int col_width(int col);
00684
00689 void row_height_all(int height) { // set all row/col heights
00690     for ( int r=0; r<rows(); r++ ) {
00691         row_height(r, height);
00692     }
00693 }
00694
00699 void col_width_all(int width) {
00700     for ( int c=0; c<cols(); c++ ) {
00701         col_width(c, width);
00702     }
00703 }
00704
00705 void row_position(int row); // set/get table's current scroll position
00706 void col_position(int col);
00707
00711 int row_position() {
00712     return(_row_position);
00713 }
00714
00718 int col_position() {
00719     return(_col_position);
00720 }
00721
00727 inline void top_row(int row) { // set/get top row (deprecated)
00728     row_position(row);
00729 }
00730
00735 inline int top_row() {
00736     return(row_position());
00737 }
00738
00738 int is_selected(int r, int c); // selected cell
00739 void get_selection(int &row_top, int &col_left, int &row_bot, int &col_right);
00740 void set_selection(int row_top, int col_left, int row_bot, int col_right);
00741 int move_cursor(int R, int C, int shiftselect);
00742 int move_cursor(int R, int C);
00743 void resize(int X, int Y, int W, int H) FL_OVERRIDE; // fltk resize() FL_OVERRIDE

```

```

00744
00745 // This crashes sortapp() during init.
00746 // void box(Fl_Boxtype val) {
00747 //     Fl_Group::box(val);
00748 //     if ( table ) {
00749 //         resize(x(), y(), w(), h());
00750 //     }
00751 // }
00752 // Fl_Boxtype box(void) const {
00753 //     return(Fl_Group::box());
00754 // }
00755
00756 // Child group management
00757
00762 void init_sizes() {
00763     table->init_sizes();
00764     table->redraw();
00765 }
00766
00771 void add(Fl_Widget& wgt) {
00772     table->add(wgt);
00773     if ( table->children() > 2 ) {
00774         table->show();
00775     } else {
00776         table->hide();
00777     }
00778 }
00779
00784 void add(Fl_Widget* wgt) {
00785     add(*wgt);
00786 }
00787
00792 void insert(Fl_Widget& wgt, int n) {
00793     table->insert(wgt,n);
00794 }
00795
00801 void insert(Fl_Widget& wgt, Fl_Widget* w2) {
00802     table->insert(wgt,w2);
00803 }
00804
00808 void remove(Fl_Widget& wgt) {
00809     table->remove(wgt);
00810 }
00811
00812 // (doxygen will substitute Fl_Group's docs here)
00813 void begin() {
00814     table->begin();
00815 }
00816
00817 // (doxygen will substitute Fl_Group's docs here)
00818 void end() {
00819     table->end();
00820     // HACK: Avoid showing Fl_Scroll; seems to erase screen
00821     //         causing unnecessary flicker, even if its box() is FL_NO_BOX.
00822     //
00823     if ( table->children() > 2 ) {
00824         table->show();
00825     } else {
00826         table->hide();
00827     }
00828     Fl_Group::current(Fl_Group::parent());
00829 }
00830
00835 Fl_Widget*const* array() {
00836     return(table->array());
00837 }
00838
00853 Fl_Widget *child(int n) const {
00854     return(table->child(n));
00855 }
00856
00865 int children() const {
00866     return(table->children()-2);    // -2: skip Fl_Scroll's h/v scrollbar widgets
00867 }
00868
00869 // (doxygen will substitute Fl_Group's docs here)
00870 int find(const Fl_Widget *wgt) const {
00871     return(table->find(wgt));
00872 }
00873
00874 // (doxygen will substitute Fl_Group's docs here)
00875 int find(const Fl_Widget &wgt) const {
00876     return(table->find(wgt));
00877 }
00878
00879 // CALLBACKS
00880

```

```

00886 int callback_row() {
00887     return(_callback_row);
00888 }
00889
00895 int callback_col() {
00896     return(_callback_col);
00897 }
00898
00904 TableContext callback_context() {
00905     return(_callback_context);
00906 }
00907
00915 void do_callback(TableContext context, int row, int col) {
00916     _callback_context = context;
00917     _callback_row = row;
00918     _callback_col = col;
00919     Fl_Widget::do_callback();
00920 }
00921
00922 #ifdef FL_DOXYGEN
00951 void when(Fl_When flags);
00952 #endif
00953
00954 #ifdef FL_DOXYGEN
01032 void callback(Fl_Widget*, void*);
01033 #endif
01034
01044 int scrollbar_size() const {
01045     return(_scrollbar_size);
01046 }
01047
01066 void scrollbar_size(int newSize) {
01067     if ( newSize != _scrollbar_size ) redraw();
01068     _scrollbar_size = newSize;
01069 }
01070
01084 void tab_cell_nav(int val) {
01085     if ( val ) flags_ |= TABCELLNAV;
01086     else      flags_ &= ~TABCELLNAV;
01087 }
01088
01096 int tab_cell_nav() const {
01097     return(flags_ & TABCELLNAV ? 1 : 0);
01098 }
01099 };
01100
01101 #endif /*_FL_TABLE_H*/

```

12.146 Fl_Table_Row.H

```

00001 //
00002
00003 #ifndef _FL_TABLE_ROW_H
00004 #define _FL_TABLE_ROW_H
00005
00006 //
00007 // Fl_Table_Row -- A row oriented table widget for the Fast Light Tool Kit (FLTK).
00008 //
00009 //     A class specializing in a table of rows.
00010 //     Handles row-specific selection behavior.
00011 //
00012 // Copyright 2002 by Greg Ercolano.
00013 //
00014 // This library is free software. Distribution and use rights are outlined in
00015 // the file "COPYING" which should have been included with this file.  If this
00016 // file is missing or damaged, see the license at:
00017 //
00018 //     https://www.fltk.org/COPYING.php
00019 //
00020 // Please see the following page on how to report bugs and issues:
00021 //
00022 //     https://www.fltk.org/bugs.php
00023 //
00024
00025 #include <FL/Fl_Table.H>
00026
00044 class FL_EXPORT Fl_Table_Row : public Fl_Table {
00045 public:
00046     enum TableRowSelectMode {
00047         SELECT_NONE,           // no selection allowed
00048         SELECT_SINGLE,         // single row selection
00049         SELECT_MULTIPLE        // multiple row selection (default)
00050     };
00051 private:
00052     // An STL-ish vector without templates

```

```

00053 class FL_EXPORT CharVector {
00054     char *arr;
00055     int _size;
00056     void init() {
00057         arr = 0;
00058         _size = 0;
00059     }
00060     void copy(char *newarr, int newsz);
00061 public:
00062     CharVector() { // CTOR
00063         init();
00064     }
00065     ~CharVector(); // DTOR
00066     CharVector(CharVector&o) { // COPY CTOR
00067         init();
00068         copy(o.arr, o._size);
00069     }
00070     CharVector& operator=(CharVector&o) { // ASSIGN
00071         init();
00072         copy(o.arr, o._size);
00073         return(*this);
00074     }
00075     char operator[](int x) const {
00076         return(arr[x]);
00077     }
00078     char& operator[](int x) {
00079         return(arr[x]);
00080     }
00081     int size() {
00082         return(_size);
00083     }
00084     void size(int count);
00085     char pop_back() {
00086         char tmp = arr[_size-1];
00087         _size--;
00088         return(tmp);
00089     }
00090     void push_back(char val) {
00091         int x = _size;
00092         size(_size+1);
00093         arr[x] = val;
00094     }
00095     char back() {
00096         return(arr[_size-1]);
00097     }
00098 };
00099
00100 CharVector _rowselect; // selection flag for each row
00101
00102 // handle() state variables.
00103 // Put here instead of local statics in handle(), so more
00104 // than one instance can exist without crosstalk between.
00105 //
00106 int _dragging_select; // dragging out a selection?
00107 int _last_row;
00108 int _last_y; // last event's Y position
00109 int _last_push_x; // last PUSH event's X position
00110 int _last_push_y; // last PUSH event's Y position
00111
00112 TableRowSelectMode _selectmode;
00113
00114 protected:
00115     int handle(int event) FL_OVERRIDE;
00116     int find_cell(TableContext context, // find cell's x/y/w/h given r/c
00117         int R, int C, int &X, int &Y, int &W, int &H) {
00118         return(Fl_Table::find_cell(context, R, C, X, Y, W, H));
00119     }
00120
00121 public:
00122     Fl_Table_Row(int X, int Y, int W, int H, const char *l=0) : Fl_Table(X,Y,W,H,l) {
00123         _dragging_select = 0;
00124         _last_row = -1;
00125         _last_y = -1;
00126         _last_push_x = -1;
00127         _last_push_y = -1;
00128         _selectmode = SELECT_MULTII;
00129     }
00130
00131     ~Fl_Table_Row() { }
00132
00133     void rows(int val) FL_OVERRIDE; // set number of rows
00134     int rows() { // get number of rows
00135         return(Fl_Table::rows());
00136     }
00137
00138     void type(TableRowSelectMode val); // set selection mode
00139
00140
00141

```

```

00156 TableRowSelectionMode type() const { // get selection mode
00157     return(_selectmode);
00158 }
00159
00160 // Checks to see if 'row' is selected. Returns 1 if selected, 0 if not.
00161 int row_selected(int row);
00162
00163 // Changes the selection state for 'row', depending on the value of 'flag'.
00164 int select_row(int row, int flag = 1);
00165
00170 void select_all_rows(int flag=1); // all rows to a known state
00171
00172 void clear() FL_OVERRIDE {
00173     rows(0); // implies clearing selection
00174     cols(0);
00175     Fl_Table::clear(); // clear the table
00176 }
00177 };
00178
00179 #endif /*_FL_TABLE_ROW_H*/

```

12.147 Fl_Tabs.H

```

00001 //
00002 // Tab header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Tabs widget . */
00019
00020 #ifndef Fl_Tabs_H
00021 #define Fl_Tabs_H
00022
00023 #include "Fl_Group.H"
00024
00025 struct Fl_Menu_Item;
00026
00028 class FL_EXPORT Fl_Tabs : public Fl_Group {
00029
00030     Fl_Widget *push_;
00031
00032 protected:
00033
00034     int overflow_type;
00035     int tab_offset;
00036     int *tab_pos;
00037     int *tab_width;
00038     int *tab_flags;
00039     int tab_count;
00040     Fl_Align tab_align;
00041     int has_overflow_menu;
00042
00043     void take_focus(Fl_Widget *o);
00044     int maybe_do_callback(Fl_Widget *o);
00045     void check_overflow_menu();
00046     void handle_overflow_menu();
00047     void draw_overflow_menu_button();
00048
00049     int on_insert(Fl_Widget*, int) FL_OVERRIDE;
00050     int on_move(int, int) FL_OVERRIDE;
00051     void on_remove(int) FL_OVERRIDE;
00052
00053     virtual void redraw_tabs();
00054     virtual int tab_positions(); // allocate and calculate tab positions
00055     virtual void clear_tab_positions();
00056     virtual void draw_tab(int x1, int x2, int W, int H, Fl_Widget* o, int flags, int sel);
00057     virtual int tab_height();
00058     virtual int hit_close(Fl_Widget *o, int event_x, int event_y);
00059     virtual int hit_overflow_menu(int event_x, int event_y);
00060     virtual int hit_tabs_area(int event_x, int event_y);
00061
00062     void draw() FL_OVERRIDE;
00063
00064 };

```

```

00284 public:
00285
00286     Fl_Tabs(int X, int Y, int W, int H, const char *L = 0);
00287     ~Fl_Tabs() FL_OVERRIDE;
00288
00289     void resize(int, int, int, int) FL_OVERRIDE;
00290     void show() FL_OVERRIDE;
00291
00292     int handle(int) FL_OVERRIDE;
00293     Fl_Widget *value();
00294     int value(Fl_Widget *);
00295
00306     Fl_Widget *push() const { return push_; }
00307     int push(Fl_Widget *);
00308
00309     virtual Fl_Widget *which(int event_x, int event_y);
00310     void client_area(int &rx, int &ry, int &rw, int &rh, int tabh=0);
00311
00323     void tab_align(Fl_Align a) { tab_align_ = a; }
00324
00330     Fl_Align tab_align() const { return tab_align_; }
00331
00332     enum {
00333         OVERFLOW_COMPRESS = 0,
00334         OVERFLOW_CLIP,
00335         OVERFLOW_PULLDOWN,
00336         OVERFLOW_DRAG
00337     };
00338
00339     void handle_overflow(int ov);
00340
00341 };
00342
00343 #endif

```

12.148 Fl_Terminal.H File Reference

[Fl_Terminal](#) widget.

```

#include <FL/Fl.H>
#include <FL/Fl_Window.H>
#include <FL/Fl_Group.H>
#include <FL/Fl_Scrollbar.H>
#include <FL/Fl_Rect.H>
#include <stdarg.h>

```

Classes

- class [Fl_Terminal::CharStyle](#)
- class [Fl_Terminal::Cursor](#)
- class [Fl_Terminal::EscapeSeq](#)
- class [Fl_Terminal](#)

Terminal widget supporting Unicode/utf-8, ANSI/xterm escape codes with full RGB color control.

- class [Fl_Terminal::Margin](#)
- class [Fl_Terminal::PartialUtf8Buf](#)
- class [Fl_Terminal::RingBuffer](#)
- class [Fl_Terminal::Selection](#)
- class [Fl_Terminal::Utf8Char](#)

12.148.1 Detailed Description

[Fl_Terminal](#) widget.

12.149 Fl_Terminal.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Fl_Terminal - A terminal widget for Fast Light Tool Kit (FLTK).

```

```

00003 //
00004 // Copyright 2022 by Greg Ercolano.
00005 // Copyright 2024 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //      https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //      https://www.fltk.org/bugs.php
00016 //
00017
00022 #ifndef Fl_Terminal_H
00023 #define Fl_Terminal_H
00024
00025 #include <FL/Fl.H>
00026 #include <FL/Fl_Window.H>
00027 #include <FL/Fl_Group.H>
00028 #include <FL/Fl_Scrollbar.H>
00029 #include <FL/Fl_Rect.H>
00030
00031 #include <stdarg.h>          // va_list (MinGW)
00032
00032 class FL_EXPORT Fl_Terminal : public Fl_Group {
00032 public:
00032     enum RedrawStyle {
00033         NO_REDRAW=0,
00034         RATE_LIMITED,
00035         PER_WRITE
00036     };
00037
00036     enum Attrib {
00037         NORMAL      = 0x00,
00038         BOLD        = 0x01,
00039         DIM         = 0x02,
00040         ITALIC      = 0x04,
00041         UNDERLINE   = 0x08,
00042         _RESERVED_1 = 0x10,
00043         INVERSE     = 0x20,
00044         _RESERVED_2 = 0x40,
00045         STRIKEOUT   = 0x80
00046     };
00047
00046     enum CharFlags {
00047         FG_XTERM    = 0x01,
00048         BG_XTERM    = 0x02,
00049         EOL         = 0x04,
00050         RESV_A      = 0x08,
00051         RESV_B      = 0x10,
00052         RESV_C      = 0x20,
00053         RESV_D      = 0x40,
00054         RESV_E      = 0x80,
00055         COLORMASK   = (FG_XTERM | BG_XTERM)
00056     };
00057
00056     enum OutFlags {
00057         OFF          = 0x00,
00058         CR_TO_LF     = 0x01,
00059         LF_TO_CR     = 0x02,
00060         LF_TO_CRLF   = 0x04
00061     };
00062
00061     enum ScrollbarStyle {
00062         SCROLLBAR_OFF = 0x00,
00063         SCROLLBAR_AUTO = 0x01,
00064         SCROLLBAR_ON  = 0x02
00065     };
00066
00066 protected:
00067 // Margin Class //////////////////////////////////////
00068 //
00069 // Class to manage the terminal's margins
00070 //
00071 class FL_EXPORT Margin {
00072     int left_, right_, top_, bottom_;
00073 public:
00074     Margin(void) { left_ = right_ = top_ = bottom_ = 3; }
00075     int left(void) const { return left_; }
00076     int right(void) const { return right_; }
00077     int top(void) const { return top_; }
00078     int bottom(void) const { return bottom_; }
00079     void left(int val) { left_ = val; }
00080     void right(int val) { right_ = val; }
00081     void top(int val) { top_ = val; }
00082     void bottom(int val) { bottom_ = val; }

```

```

00416     void bottom(int val)    { bottom_ = val;  }
00417 };
00418
00419 // CharStyle Class //////////////////////////////////////
00420 //
00421 // Class to manage the terminal's character style
00422 // This includes the font, color, and some cached internal
00423 // info for optimized drawing speed.
00424 //
00425 class FL_EXPORT CharStyle {
00426     uchar      attrib_;      // bold, underline..
00427     uchar      charflags_;   // CharFlags (xterm color management)
00428     Fl_Color    fgcolor_;    // foreground color for text
00429     Fl_Color    bgcolor_;    // background color for text
00430     Fl_Color    defaultfgcolor_; // default fg color used by ESC[0m
00431     Fl_Color    defaultbgcolor_; // default bg color used by ESC[0m
00432     Fl_Font     fontface_;   // font face
00433     Fl_Fontsize fontsize_;   // font size
00434     int         fontheight_; // font height (in pixels)
00435     int         fontdescent_; // font descent (pixels below font baseline)
00436     int         charwidth_;  // width of a fixed width ASCII character
00437 public:
00438     CharStyle(bool fontsize_defer);
00439     uchar attrib(void) const { return attrib_; }
00440     uchar charflags(void) const { return charflags_; }
00441     // Colors - All access to colors are by Fl_Color only.
00442     //       There are three ways to SET colors: Fl_Color, rgb, xterm(uchar)
00443     //
00444     Fl_Color fltk_fg_color(uchar ci);
00445     Fl_Color fltk_bg_color(uchar ci);
00446     Fl_Color fgcolor(void) const;
00447     Fl_Color bgcolor(void) const;
00448     Fl_Color defaultfgcolor(void) const { return defaultfgcolor_; }
00449     Fl_Color defaultbgcolor(void) const { return defaultbgcolor_; }
00450     Fl_Font fontface(void) const { return fontface_; }
00451     Fl_Fontsize fontsize(void) const { return fontsize_; }
00452     int fontheight(void) const { return fontheight_; }
00453     int fontdescent(void) const { return fontdescent_; }
00454     int charwidth(void) const { return charwidth_; }
00455     uchar colorbits_only(uchar inflags) const;
00456     void attrib(uchar val) { attrib_ = val; }
00457     void charflags(uchar val) { charflags_ = val; }
00458     void set_charflag(uchar val) { charflags_ |= val; }
00459     void clr_charflag(uchar val) { charflags_ &= ~val; }
00460     // Non-xterm colors
00461     void fgcolor(int r,int g,int b) { fgcolor_ = (r<<24) | (g<<16) | (b<<8); clr_charflag(FG_XTERM); }
00462     void bgcolor(int r,int g,int b) { bgcolor_ = (r<<24) | (g<<16) | (b<<8); clr_charflag(BG_XTERM); }
00463     void fgcolor(Fl_Color val) { fgcolor_ = val; }
00464     void bgcolor(Fl_Color val) { bgcolor_ = val; }
00465     // Xterm colors
00466     void fgcolor_xterm(Fl_Color val) { fgcolor_ = val; }
00467     void bgcolor_xterm(Fl_Color val) { bgcolor_ = val; }
00468     void fgcolor_xterm(uchar val);
00469     void bgcolor_xterm(uchar val);
00470     //
00471     void defaultfgcolor(Fl_Color val) { defaultfgcolor_ = val; }
00472     void defaultbgcolor(Fl_Color val) { defaultbgcolor_ = val; }
00473     void fontface(Fl_Font val) { fontface_ = val; update(); }
00474     void fontsize(Fl_Fontsize val) { fontsize_ = val; update(); }
00475     void update(void);
00476     void update_fake(void);
00477     // SGR MODES: Set Graphics Rendition
00478     void sgr_reset(void) { // e.g. ESC[0m
00479         attrib(Fl_Terminal::NORMAL);
00480         if (charflags() & FG_XTERM) fgcolor_xterm(defaultfgcolor_);
00481         else fgcolor(defaultfgcolor_);
00482         if (charflags() & BG_XTERM) bgcolor_xterm(defaultbgcolor_);
00483         else bgcolor(defaultbgcolor_);
00484     }
00485     int onoff(bool flag, Attrib a) { return (flag ? (attrib_ | a) : (attrib_ & ~a)); }
00486     void sgr_bold(bool val) { attrib_ = onoff(val, Fl_Terminal::BOLD); } // e.g. ESC[1m
00487     void sgr_dim(bool val) { attrib_ = onoff(val, Fl_Terminal::DIM); } // e.g. ESC[2m
00488     void sgr_italic(bool val) { attrib_ = onoff(val, Fl_Terminal::ITALIC); } // e.g. ESC[3m
00489     void sgr_underline(bool val) { attrib_ = onoff(val, Fl_Terminal::UNDERLINE); } // e.g. ESC[3m
00490     void sgr_dbf_under(bool val) { attrib_ = onoff(val, Fl_Terminal::UNDERLINE); } // e.g. ESC[21m
00491     (TODO!)
00492     void sgr_blink(bool val) { (void)val; /* NOT IMPLEMENTED */ } // e.g. ESC[5m
00493     void sgr_inverse(bool val) { attrib_ = onoff(val, Fl_Terminal::INVERSE); } // e.g. ESC[7m
00494     void sgr_strike(bool val) { attrib_ = onoff(val, Fl_Terminal::STRIKEOUT); } // e.g. ESC[9m
00495 };

```



```

00496 protected:
00497 // Cursor Class //////////////////////////////////////
00498 //
00499 // Class to manage the terminal's cursor position, color, etc.
00500 //
00501 class FL_EXPORT Cursor {
00502     int col_; // cursor's current col (x) position on display
00503     int row_; // cursor's current row (y) position on display
00504     int h_; // cursor's height (affected by font size)
00505     Fl_Color fgcolor_; // cursor's fg color (color of text, if any)
00506     Fl_Color bgcolor_; // cursor's bg color
00507 public:
00508     Cursor(void) {
00509         col_ = 0;
00510         row_ = 0;
00511         h_ = 10;
00512         fgcolor_ = 0xfffff000; // wht
00513         bgcolor_ = 0x00d00000; // grn
00514     }
00515     int col(void) const { return col_; }
00516     int row(void) const { return row_; }
00517     int h(void) const { return h_; }
00518     Fl_Color fgcolor(void) const { return fgcolor_; }
00519     Fl_Color bgcolor(void) const { return bgcolor_; }
00520     void col(int val) { col_ = val >= 0 ? val : 0; }
00521     void row(int val) { row_ = val >= 0 ? val : 0; }
00522     void h(int val) { h_ = val; }
00523     void fgcolor(Fl_Color val) { fgcolor_ = val; }
00524     void bgcolor(Fl_Color val) { bgcolor_ = val; }
00525     int left(void) { col_ = (col_>0) ? (col_-1) : 0; return col_; }
00526     int right(void) { return ++col_; }
00527     int up(void) { row_ = (row_>0) ? (row_-1) : 0; return row_; }
00528     int down(void) { return ++row_; }
00529     bool is_rowcol(int drow,int dcol) const;
00530     void scroll(int nrows);
00531     void home(void) { row_ = 0; col_ = 0; }
00532 };
00533
00534 // Utf8Char Class //////////////////////////////////////
00535 //
00536 // Class to manage the terminal's individual UTF-8 characters.
00537 // Includes fg/bg color, attributes (BOLD, UNDERLINE..)
00538 //
00539 class FL_EXPORT Utf8Char {
00540     static const int max_utf8_ = 4; // RFC 3629 paraphrased: In UTF-8, chars are encoded with 1 to 4
octets
00541     char text_[max_utf8_]; // memory for actual ASCII or UTF-8 byte contents
00542     uchar len_; // length of bytes in text_[] buffer; 1 for ASCII, >1 for UTF-8
00543     uchar attrib_; // attribute bits for this char (bold, underline..)
00544     uchar charflags_; // CharFlags (xterm colors management)
00545     Fl_Color fgcolor_; // fltk fg color (supports 8color or 24bit color set
w/ESC[37;<cr>;<g>;<b>m)
00546     Fl_Color bgcolor_; // fltk bg color (supports 8color or 24bit color set
w/ESC[47;<cr>;<g>;<b>m)
00547 // Private methods
00548     void text_utf8(const char *text, int len);
00549     Fl_Color attr_color_(Fl_Color col, const Fl_Widget *grp) const;
00550 public:
00551 // Public methods
00552     Utf8Char(void); // ctor
00553     Utf8Char(const Utf8Char& o); // copy ctor
00554     ~Utf8Char(void); // dtor
00555     Utf8Char& operator=(const Utf8Char& o); // assignment
00556     inline int max_utf8() const { return max_utf8_; }
00557     void text_utf8(const char *text, int len, const CharStyle& style);
00558     void text_ascii(char c, const CharStyle& style);
00559     void fl_font_set(const CharStyle& style) const;
00560
00561 // Return the UTF-8 text string for this character.
00562 // Use length() to get number of bytes in string, which will be 1 for ASCII chars.
00563 //
00564     const char* text_utf8(void) const { return text_; }
00565 // Return the attribute for this char
00566     uchar attrib(void) const { return attrib_; }
00567     uchar charflags(void) const { return charflags_; }
00568     Fl_Color fgcolor(void) const;
00569     Fl_Color bgcolor(void) const;
00570 // Return the length of this character in bytes (UTF-8 can be multibyte..)
00571     int length(void) const { return int(len_); }
00572     double pwidth(void) const;
00573     int pwidth_int(void) const;
00574 // Clear the character to a 'space'
00575     void clear(const CharStyle& style) { text_utf8(" ", 1, style); charflags_ = 0; attrib_ = 0; }
00576     bool is_char(char c) const { return *text_ == c; }
00577     void show_char(void) const { ::printf("%.s", len_, text_); }
00578     void show_char_info(void) const { ::fprintf(stderr, "UTF-8('%.s', len=%d)\n", len_, text_, len_); }
}

```

```

00579     Fl_Color attr_fg_color(const Fl_Widget *grp) const;
00580     Fl_Color attr_bg_color(const Fl_Widget *grp) const;
00581 };
00582
00583 // RingBuffer Class //////////////////////////////////////
00584 //
00585 // Manages ring with indexed row/col and "history" vs. "display" concepts.
00586 //
00587 class FL_EXPORT RingBuffer {
00588     Utf8Char *ring_chars_; // the ring UTF-8 char buffer
00589     int ring_rows_;        // #rows in ring total
00590     int ring_cols_;        // #columns in ring/hist/disp
00591     int nchars_;           // #chars in ring (ring_rows*ring_cols)
00592     int hist_rows_;        // #rows in history
00593     int hist_use_;         // #rows in use by history
00594     int disp_rows_;        // #rows in display
00595     int offset_;           // index offset (used for 'scrolling')
00596
00597 private:
00598     void new_copy(int drows, int dcols, int hrows, const CharStyle& style);
00599     //DEBUG void write_row(FILE *fp, Utf8Char *u8c, int cols) const {
00600     //DEBUG     cols = (cols != 0) ? cols : ring_cols();
00601     //DEBUG     for (int col=0; col<cols; col++, u8c++) {
00602     //DEBUG         ::fprintf(fp, "%.*s", u8c->length(), u8c->text_utf8());
00603     //DEBUG     }
00604     //DEBUG }
00605 public:
00606     void clear(void);
00607     void clear_hist(void);
00608     RingBuffer(void);
00609     RingBuffer(int drows, int dcols, int hrows);
00610     ~RingBuffer(void);
00611
00612     // Methods to access ring
00613     //
00614     // The 'offset' concept allows the 'history' and 'display'
00615     // to be scrolled indefinitely. The 'offset' is applied
00616     // to all the row accesses, and are clamped to within their bounds.
00617     //
00618     // For 'raw' access to the ring (without the offset concept),
00619     // use the ring_chars() method, and walk from 0 - ring_rows().
00620     //
00621     //
00622     //      |-----| <- hist_srow() <- ring_srow()
00623     //      |   H i s t   |
00624     //      |-----|
00625     //      |-----| <- hist_erow()
00626     //      |   D i s p   | <- disp_srow()
00627     //      |-----|
00628     //      |-----| <- disp_erow() <- ring_erow()
00629     //
00630     //
00631     //      \-----/
00632     //      ring_cols()
00633     //      hist_cols()
00634     //      disp_cols()
00635     //
00636     inline int ring_rows(void) const { return ring_rows_; }
00637     inline int ring_cols(void) const { return ring_cols_; }
00638     inline int ring_srow(void) const { return 0; }
00639     inline int ring_erow(void) const { return(ring_rows_ - 1); }
00640     inline int hist_rows(void) const { return hist_rows_; }
00641     inline int hist_cols(void) const { return ring_cols_; }
00642     inline int hist_srow(void) const { return((offset_ + 0) % ring_rows_); }
00643     inline int hist_erow(void) const { return((offset_ + hist_rows_ - 1) % ring_rows_); }
00644     inline int disp_rows(void) const { return disp_rows_; }
00645     inline int disp_cols(void) const { return ring_cols_; }
00646     inline int disp_srow(void) const { return((offset_ + hist_rows_) % ring_rows_); }
00647     inline int disp_erow(void) const { return((offset_ + hist_rows_ + disp_rows_ - 1) % ring_rows_); }
00648     inline int offset(void) const { return offset_; }
00649     void offset_adjust(int rows);
00650     void hist_rows(int val) { hist_rows_ = val; }
00651     void disp_rows(int val) { disp_rows_ = val; }
00652
00653     // History use
00654     inline int hist_use(void) const { return hist_use_; }
00655     inline void hist_use(int val) { hist_use_ = val; }
00656     inline int hist_use_srow(void) const { return((offset_ + hist_rows_ - hist_use_) % ring_rows_); }
00657 }
00658
00659 inline Utf8Char *ring_chars(void) { return ring_chars_; } // access ring buffer directly
00660 inline Utf8Char *ring_chars(void) const { return ring_chars_; } // access ring buffer directly
00661
00662 bool is_hist_ring_row(int grow) const;
00663 bool is_disp_ring_row(int grow) const;
00664 //DEBUG void show_ring_info(void) const;
00665 void move_disp_row(int src_row, int dst_row);
00666 void clear_disp_rows(int sdrow, int edrow, const CharStyle& style);

```

```

00665     void scroll(int rows, const CharStyle& style);
00666
00667     const Utf8Char* u8c_ring_row(int row) const;
00668     const Utf8Char* u8c_hist_row(int hrow) const;
00669     const Utf8Char* u8c_hist_use_row(int hurow) const;
00670     const Utf8Char* u8c_disp_row(int drow) const;
00671     // Non-const versions of the above methods
00672     // Using "Effective C++" ugly-as-hell syntax technique.
00673     //
00674     Utf8Char* u8c_ring_row(int row);
00675     Utf8Char* u8c_hist_row(int hrow);
00676     Utf8Char* u8c_hist_use_row(int hurow);
00677     Utf8Char* u8c_disp_row(int drow);
00678
00679     void create(int drows, int dcols, int hrows);
00680     void resize(int drows, int dcols, int hrows, const CharStyle& style);
00681
00682     void change_disp_rows(int drows, const CharStyle& style);
00683     void change_disp_cols(int dcols, const CharStyle& style);
00684 };
00685
00686 // Selection Class //////////////////////////////////////
00687 //
00688 // Class to manage mouse selection
00689 //
00690 class FL_EXPORT Selection {
00691     Fl_Terminal *terminal_;
00692     int srow_, scol_, erow_, ecol_; // selection start/end. NOTE: start *might* be > end
00693     int push_row_, push_col_; // global row/col for last FL_PUSH
00694     bool push_char_right_;
00695     Fl_Color selectionbgcolor_;
00696     Fl_Color selectionfgcolor_;
00697     int state_; // 0=none, 1=started, 2=extended, 3=done
00698     bool is_selection_; // false: no selection
00699 public:
00700     Selection(Fl_Terminal *terminal);
00701     int srow(void) const { return srow_; }
00702     int scol(void) const { return scol_; }
00703     int erow(void) const { return erow_; }
00704     int ecol(void) const { return ecol_; }
00705     void push_clear() { push_row_ = push_col_ = -1; push_char_right_ = false; }
00706     void push_rowcol(int row, int col, bool char_right) {
00707         push_row_ = row; push_col_ = col; push_char_right_ = char_right; }
00708     void start_push() { start(push_row_, push_col_, push_char_right_); }
00709     bool dragged_off(int row, int col, bool char_right) {
00710         return (push_row_ != row) || (push_col_ + push_char_right_ != col + char_right); }
00711     void selectionfgcolor(Fl_Color val) { selectionfgcolor_ = val; }
00712     void selectionbgcolor(Fl_Color val) { selectionbgcolor_ = val; }
00713     Fl_Color selectionfgcolor(void) const { return selectionfgcolor_; }
00714     Fl_Color selectionbgcolor(void) const { return selectionbgcolor_; }
00715     bool is_selection(void) const { return is_selection_; }
00716     bool get_selection(int &srow, int &scol, int &erow, int &ecol) const; // guarantees return (start <
end)
00717     bool start(int row, int col, bool char_right);
00718     bool extend(int row, int col, bool char_right);
00719     void end(void);
00720     void select(int srow, int scol, int erow, int ecol);
00721     bool clear(void);
00722     int state(void) const { return state_; }
00723     void scroll(int nrows);
00724 };
00725
00726 // EscapeSeq Class //////////////////////////////////////
00727 //
00728 // Class to handle parsing ESC sequences
00729 //
00730 // Holds all state information for parsing esc sequences,
00731 // so sequences can span multiple block read(2) operations, etc.
00732 // Handling of parsed sequences is NOT handled in this class,
00733 // just the parsing of the sequences and managing generic integers.
00734 //
00735 class FL_EXPORT EscapeSeq {
00736 public:
00737     // EscapeSeq Constants
00738     // Maximums
00739     static const int maxbuff = 80; // character buffer
00740     static const int maxvals = 20; // integer value buffer
00741     // Return codes
00742     static const int success = 0; // operation succeeded
00743     static const int fail = -1; // operation failed
00744     static const int completed = 1; // multi-step operation completed successfully
00745 private:
00746     char esc_mode_; // escape parsing mode state
00747     char csi_; // This is an ESC[.. sequence (Ctrl Seq Introducer)
00748     char buff_[maxbuff]; // escape sequence being parsed
00749     char *bufptr_; // parsing ptr into buff[]
00750     char *buffendp_; // end of buff[] (ptr to last valid buff char)

```

```

00751     char *valbufp_;           // pointer to first char in buff of integer being parsed
00752     int  vals_[maxvals];      // value array for parsing #'s in ESC[#;#;#..
00753     int  vali_;              // parsing index into vals[], 0 if none
00754     int  save_row_, save_col_; // used by ESC[s/u for save/restore
00755
00756     int append_buff(char c);
00757     int append_val(void);
00758
00759 public:
00760     EscapeSeq(void);
00761     void reset(void);
00762     char esc_mode(void) const;
00763     void esc_mode(char val);
00764     int  total_vals(void) const;
00765     int  val(int i) const;
00766     int  defvalmax(int dval, int max) const;
00767     bool parse_in_progress(void) const;
00768     bool is_csi(void) const;
00769     int  parse(char c);
00770     void save_cursor(int row, int col);
00771     void restore_cursor(int &row, int &col);
00772 };
00773
00774 // Partial UTF-8 Buffer Class //////////////////////////////////////
00775 //
00776 // Class to manage buffering partial UTF-8 characters between write calls.
00777 //
00778 class FL_EXPORT PartialUtf8Buf {
00779     char buf_[10];           // buffer partial UTF-8 encoded char
00780     int  buflen_;           // length of buffered UTF-8 encoded char
00781     int  clen_;             // final byte length of a UTF-8 char
00782 public:
00783     void clear(void) { buflen_ = clen_ = 0; } // clear the buffer
00784     PartialUtf8Buf(void) { clear(); } // Ctor
00785     // Is byte 'c' in the middle of a UTF-8 encoded byte sequence?
00786     bool is_continuation(char c) {
00787         //           Byte 1      Byte 2      Byte 3      ..etc..
00788         // ASCII:  0xxxxxxx
00789         // UTF8(2): 110xxxxx 10xxxxxx
00790         // UTF8(3): 1110xxxx 10xxxxxx 10xxxxxx
00791         // UTF8(4): 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
00792         // UTF8(5): 111110xx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx
00793         // UTF8(6): 1111110x 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx
00794         //           \_____/ \_____/ \_____/ \_____/ \_____/ \_____/
00795         //           Start byte      Continuation bytes
00796         //           (c & 0xc0) == 0x80
00797         return ((c & 0xc0) == 0x80);
00798     }
00799     // Access buffer
00800     const char* buf(void) const { return buf_; }
00801     // Access buffer length
00802     int buflen(void) const { return buflen_; }
00803     // Append bytes of a partial UTF-8 string to the buffer.
00804     //
00805     // Returns:
00806     // - true if done OK. Use is_complete() to see if a complete char received.
00807     // - false if buffer overrun occurred, class is clear()ed.
00808     //
00809     // An appropriate response to 'false' would be to print the
00810     // "unknown character" and skip all subsequent UTF-8 continuation chars.
00811     //
00812     bool append(const char* p, int len) {
00813         if (len <= 0) return true; // ignore silly requests: say we did but dont
00814         if (buflen_ + len >= (int)sizeof(buf_)) // overrun check
00815             { clear(); return false; } // clear self, return false
00816         if (!buflen_) clen_ = fl_utf8len(*p); // first byte? save char len for later
00817         while (len>0) { buf_[buflen_++] = *p++; len--; } // append byte to buffer
00818         return true;
00819     }
00820     bool is_complete(void) const { return (buflen_ && (buflen_ == clen_)); }
00821 };
00822
00823 public:
00842     Fl_Scrollbar *scrollbar; // vertical scrollbar (value: rows above disp_chars[])
00853     Fl_Scrollbar *hscrollbar; // horizontal scrollbar
00854 private:
00855     // Special utf8 symbols
00856     const char *error_char_; // utf8 string shown for invalid utf8, bad ANSI, etc
00857     bool fontsize_defer_; // flag defers font calcs until first draw() (issue 837)
00858     int scrollbar_size_; // local preference for scrollbar size
00859     ScrollbarStyle hscrollbar_style_;
00860     CharStyle *current_style_; // current font, attrib, color..
00861     OutFlags oflags_; // output translation flags (CR_TO_LF, LF_TO_CR, LF_TO_CRLF)
00862
00863     // A ring buffer is used for the terminal's history (hist) and display (disp) buffer.
00864     // See README-Fl_Terminal.txt, section "RING BUFFER DESCRIPTION" for diagrams/info.
00865     //

```

```

00866 // Ring buffer
00867 RingBuffer ring_; // terminal history/display ring buffer
00868 Cursor cursor_; // terminal cursor (position, color, etc)
00869 Margin margin_; // terminal margins (top,left,bottom,right)
00870 Selection select_; // mouse selection
00871 EscapeSeq escseq; // Escape sequence parsing (ESC[ xterm/vt100)
00872 bool show_unknown_; // if true, show unknown chars as '?' (default off)
00873 bool ansi_; // if true, parse ansi codes (default on)
00874 char *tabstops_; // array of tab stops (0|1) \__ TODO: This should probably
00875 int tabstops_size_; // size of tabstops[] array / be a class "TabStops".
00876 Fl_Rect scrn_; // terminal screen xywh inside box(), margins, and scrollbar
00877 int autoscroll_dir_; // 0=autoscroll timer off, 3=scrolling up, 4=scrolling down
00878 int autoscroll_amt_; // #pixels above or below edge, used for autoscroll speed
00879 RedrawStyle redraw_style_; // NO_REDRAW, RATE_LIMITED, PER_WRITE
00880 float redraw_rate_; // maximum redraw rate in seconds, default=0.10
00881 bool redraw_modified_; // display modified; used by update_cb() to rate limit redraws
00882 bool redraw_timer_; // if true, redraw timer is running
00883 PartialUtf8Buf pub_; // handles Partial Utf8 Buffer (pub)
00884
00885 protected:
00886 // Ring buffer management
00887 const Utf8Char* u8c_ring_row(int grow) const;
00888 const Utf8Char* u8c_hist_row(int hrow) const;
00889 const Utf8Char* u8c_hist_use_row(int hrow) const;
00890 const Utf8Char* u8c_disp_row(int drow) const;
00891 // Non-const versions of the above.
00892 // "Effective C++" says: implement non-const method to cast away const
00893 //
00894 Utf8Char* u8c_ring_row(int grow);
00895 Utf8Char* u8c_hist_row(int hrow);
00896 Utf8Char* u8c_hist_use_row(int hurow);
00897 Utf8Char* u8c_disp_row(int drow);
00898 Utf8Char* u8c_cursor(void);
00899 private:
00900 void create_ring(int drows, int dcols, int hrows);
00901 void init_(int X,int Y,int W,int H,const char*L,int rows,int cols,int hist,bool fontsize_defer);
00902 // Tabstops
00903 void init_tabstops(int newsz);
00904 void default_tabstops(void);
00905 void clear_all_tabstops(void);
00906 void set_tabstop(void);
00907 void clear_tabstop(void);
00908 // Updates
00909 void update_screen_xywh(void);
00910 void update_screen(bool font_changed);
00911 void set_scrollbar_params(Fl_Scrollbar* scroll, int min, int max);
00912 void update_scrollbar(void);
00913 // Resize
00914 void resize_display_rows(int drows);
00915 void resize_display_columns(int dcols);
00916 void refit_disp_to_screen(void);
00917 // Callbacks
00918 static void scrollbar_cb(Fl_Widget*, void*); // scrollbar manipulation
00919 static void autoscroll_timer_cb(void*); // mouse drag autoscroll
00920 void autoscroll_timer_cb2(void);
00921 static void redraw_timer_cb(void*); // redraw rate limiting timer
00922 void redraw_timer_cb2(void);
00923
00924 // Screen management
00925 protected:
00926 CharStyle& current_style(void) const;
00927 void current_style(const CharStyle& sty);
00928 private:
00929 int x_to_glob_col(int X, int grow, int &gcol, bool &gcr) const;
00930 int xy_to_glob_rowcol(int X, int Y, int &grow, int &gcol, bool &gcr) const;
00931 protected:
00932 int w_to_col(int W) const;
00933 int h_to_row(int H) const;
00934 // API: Display clear operations
00935 void clear_sod(void);
00936 void clear_eod(void);
00937 void clear_eol(void);
00938 void clear_sol(void);
00939 void clear_line(int row);
00940 void clear_line(void);
00941 const Utf8Char* walk_selection(const Utf8Char *u8c, int &row, int &col) const;
00942 bool get_selection(int &srow,int &scol,int &erow,int &ecol) const;
00943 bool is_selection(void) const;
00944 bool is_inside_selection(int row,int col) const;
00945 private:
00946 bool is_hist_ring_row(int grow) const;
00947 bool is_disp_ring_row(int grow) const;
00948 public:
00949 int selection_text_len(void) const;
00950 const char* selection_text(void) const;
00951 protected:
00952 void clear_mouse_selection(void);

```

```

00953 bool selection_extend(int X,int Y);
00954 void select_word(int grow, int gcol);
00955 void select_line(int grow);
00956 void scroll(int rows);
00957 void insert_rows(int count);
00958 void delete_rows(int count);
00959 void insert_char_eol(char c, int drow, int dcol, int rep);
00960 void insert_char(char c, int rep);
00961 void delete_chars(int drow, int dcol, int rep);
00962 void delete_chars(int rep);
00963 public:
00964 // API: Terminal operations
00965 void clear(void);
00966 void clear(Fl_Color val);
00967 void clear_screen(bool scroll_to_hist=true); // ESC [ 2 J
00968 void clear_screen_home(bool scroll_to_hist=true); // ESC [ H ESC [ 2 J
00969 void clear_history(void); // ESC [ 3 J
00970 void reset_terminal(void); // ESC c
00971 void cursor_home(void); // ESC [ 0 H
00972 // API: Cursor
00973 void cursorfgcolor(Fl_Color val);
00974 void cursorbgcolor(Fl_Color val);
00975 Fl_Color cursorfgcolor(void) const;
00976 Fl_Color cursorbgcolor(void) const;
00977 protected:
00978 void cursor_row(int row);
00979 void cursor_col(int col);
00980 public:
00981 int cursor_row(void) const;
00982 int cursor_col(void) const;
00983 protected:
00984 void cursor_up(int count=1, bool do_scroll=false);
00985 void cursor_down(int count=1, bool do_scroll=false);
00986 void cursor_left(int count=1);
00987 void cursor_right(int count=1, bool do_scroll=false);
00988 void cursor_eol(void);
00989 void cursor_sol(void);
00990 void cursor_cr(void);
00991 void cursor_crlf(int count=1);
00992 void cursor_tab_right(int count=1);
00993 void cursor_tab_left(int count=1);
00994 void save_cursor(void);
00995 void restore_cursor(void);
00996 // Output translation
00997 public:
00998 void output_translate(Fl_Terminal::OutFlags val);
00999 Fl_Terminal::OutFlags output_translate(void) const;
01000 private:
01001 void handle_lf(void);
01002 void handle_cr(void);
01003 void handle_esc(void);
01004 // Printing
01005 void handle_ctrl(char c);
01006 bool is_printable(char c);
01007 bool is_ctrl(char c);
01008 void handle_SGR(void);
01009 void handle DECRARA(void);
01010 void handle_escseq(char c);
01011 // --
01012 void display_modified(void);
01013 void display_modified_clear(void);
01014 void clear_char_at_disp(int drow, int dcol);
01015 const Utf8Char* utf8_char_at_disp(int drow, int dcol) const;
01016 const Utf8Char* utf8_char_at_glob(int grow, int gcol) const;
01017 void repeat_char(char c, int rep);
01018 void utf8_cache_clear(void);
01019 void utf8_cache_flush(void);
01020 // API: Character display output
01021 public:
01022 void plot_char(const char *text, int len, int drow, int dcol);
01023 void plot_char(char c, int drow, int dcol);
01024 void print_char(const char *text, int len=-1);
01025 void print_char(char c);
01026 // API: String display output
01027 void append_utf8(const char *buf, int len=-1);
01028 void append_ascii(const char *s);
01029 void append(const char *s, int len=-1);
01030 protected:
01031 int handle_unknown_char(void);
01032 int handle_unknown_char(int drow, int dcol);
01033 // Drawing
01034 void draw_row_bg(int grow, int X, int Y) const;
01035 void draw_row(int grow, int Y) const;
01036 void draw_buff(int Y) const;
01037 private:
01038 void handle_selection_autoscroll(void);
01039 int handle_selection(int e);

```

```

01040 public:
01041     // FLTK: draw(), resize(), handle()
01042     void draw(void) FL_OVERRIDE;
01043     void resize(int X,int Y,int W,int H) FL_OVERRIDE;
01044     int handle(int e) FL_OVERRIDE;
01045     const char* text(bool lines_below_cursor=false) const;
01046
01047 protected:
01048     // Internal short names
01049     // Don't make these public, but allow internals and
01050     // derived classes to maintain brevity.
01051     //
01053     inline int ring_rows(void) const { return ring_.ring_rows(); }
01055     inline int ring_cols(void) const { return ring_.ring_cols(); }
01057     inline int ring_srow(void) const { return ring_.ring_srow(); }
01059     inline int ring_erow(void) const { return ring_.ring_erow(); }
01061     inline int hist_rows(void) const { return ring_.hist_rows(); }
01063     inline int hist_cols(void) const { return ring_.hist_cols(); }
01065     inline int hist_srow(void) const { return ring_.hist_srow(); }
01067     inline int hist_erow(void) const { return ring_.hist_erow(); }
01069     inline int hist_use(void) const { return ring_.hist_use(); }
01071     inline int hist_use_srow(void) const { return ring_.hist_use_srow(); }
01073     inline int disp_rows(void) const { return ring_.disp_rows(); }
01075     inline int disp_cols(void) const { return ring_.disp_cols(); }
01077     inline int disp_srow(void) const { return ring_.disp_srow(); }
01079     inline int disp_erow(void) const { return ring_.disp_erow(); }
01081     inline int offset(void) const { return ring_.offset(); }
01082
01083     // TODO: CLEAN UP WHAT'S PUBLIC, AND WHAT SHOULD BE 'PROTECTED' AND 'PRIVATE'
01084     // Some of the public stuff should, quite simply, "not be".
01085
01086     // API: Terminal features
01087 public:
01088     // API: Scrollbar
01089     int scrollbar_size(void) const;
01090     void scrollbar_size(int val);
01091     int scrollbar_actual_size(void) const;
01092     void hscrollbar_style(ScrollbarStyle val);
01093     ScrollbarStyle hscrollbar_style(void) const;
01094     // API: History
01095     int history_rows(void) const;
01096     void history_rows(int val);
01097     int history_use(void) const;
01098     // API: Display
01099     int display_rows(void) const;
01100     void display_rows(int val);
01101     int display_columns(void) const;
01102     void display_columns(int val);
01103     // API: Box
01111     void box(Fl_Boxtype val) { Fl_Group::box(val); update_screen(false); }
01113     Fl_Boxtype box(void) const { return Fl_Group::box(); }
01114     // API: Margins
01116     int margin_left(void) const { return margin_.left(); }
01118     int margin_right(void) const { return margin_.right(); }
01120     int margin_top(void) const { return margin_.top(); }
01122     int margin_bottom(void) const { return margin_.bottom(); }
01123     void margin_left(int val);
01124     void margin_right(int val);
01125     void margin_top(int val);
01126     void margin_bottom(int val);
01127     // API: Text font/size/color
01128     void textfont(Fl_Font val);
01129     void textsize(Fl_Fonsize val);
01130     void textcolor(Fl_Color val);
01131     void color(Fl_Color val);
01132     void textfgcolor(Fl_Color val);
01133     void textbgcolor(Fl_Color val);
01134     void textfgcolor_default(Fl_Color val);
01135     void textbgcolor_default(Fl_Color val);
01137     Fl_Font textfont(void) const { return current_style_>fontface(); }
01139     Fl_Fonsize textsize(void) const { return current_style_>fontsize(); }
01141     Fl_Color color(void) const { return Fl_Group::color(); }
01143     Fl_Color textcolor(void) const { return textfgcolor_default(); }
01145     Fl_Color textfgcolor(void) const { return current_style_>fgcolor(); }
01147     Fl_Color textbgcolor(void) const { return current_style_>bgcolor(); }
01149     Fl_Color textfgcolor_default(void) const { return current_style_>defaultfgcolor(); }
01151     Fl_Color textbgcolor_default(void) const { return current_style_>defaultbgcolor(); }
01152     void textfgcolor_xterm(uchar val);
01153     void textbgcolor_xterm(uchar val);
01155     void selectionfgcolor(Fl_Color val) { select_.selectionfgcolor(val); }
01157     void selectionbgcolor(Fl_Color val) { select_.selectionbgcolor(val); }
01159     Fl_Color selectionfgcolor(void) const { return select_.selectionfgcolor(); }
01161     Fl_Color selectionbgcolor(void) const { return select_.selectionbgcolor(); }
01162     // API: Text attrib
01163     void textattrib(uchar val);
01164     uchar textattrib() const;
01165     // API: Redraw style/rate

```



```

01166     RedrawStyle redraw_style(void) const;
01167     void redraw_style(RedrawStyle val);
01168 private:
01169     bool is_redraw_style(RedrawStyle val) { return redraw_style_ == val; }
01170 public:
01171     float redraw_rate(void) const;
01172     void redraw_rate(float val);
01173     // API: Show unknown/invalid utf8/ANSI sequences with an error character (z).
01174     bool show_unknown(void) const;
01175     void show_unknown(bool val);
01176     void error_char(const char* val) { error_char_ = val; }
01177     const char* error_char(void) const { return error_char_; }
01178     // API: ANSI sequences
01179     bool ansi(void) const;
01180     void ansi(bool val);
01181     // Fl_Simple_Terminal API compatibility
01182     int history_lines(void) const;
01183     void history_lines(int val);
01184     // API: printf()
01185     void printf(const char *fmt, ...);
01186     void vprintf(const char *fmt, va_list ap);
01187     // Ctor
01188     Fl_Terminal(int X,int Y,int W,int H,const char*L=0);
01189     Fl_Terminal(int X,int Y,int W,int H,const char*L,int rows,int cols,int hist);
01190     // Dtor
01191     ~Fl_Terminal(void);
01192     // Debugging features
01193     //DEBUG void show_ring_info() const { ring_.show_ring_info(); }
01194     //DEBUG void write_row(FILE *fp, Utf8Char *u8c, int cols) const;
01195     //DEBUG void show_buffers(RingBuffer *a, RingBuffer *b=0) const;
01196 };
01197 #endif

```

12.150 Fl_Text_Buffer.H

```

00001 //
00002 // Header file for Fl_Text_Buffer class.
00003 //
00004 // Copyright 2001-2023 by Bill Spitzak and others.
00005 // Original code Copyright Mark Edel. Permission to distribute under
00006 // the LGPL for the FLTK library granted by Mark Edel.
00007 //
00008 // This library is free software. Distribution and use rights are outlined in
00009 // the file "COPYING" which should have been included with this file. If this
00010 // file is missing or damaged, see the license at:
00011 //
00012 //     https://www.fltk.org/COPYING.php
00013 //
00014 // Please see the following page on how to report bugs and issues:
00015 //
00016 //     https://www.fltk.org/bugs.php
00017 //
00018 //
00019 /* \file
00020  Fl_Text_Buffer, Fl_Text_Selection widget . */
00021
00022 #ifndef FL_TEXT_BUFFER_H
00023 #define FL_TEXT_BUFFER_H
00024
00025 #include <stdarg.h>      /* va_list */
00026 #include "fl_attr.h"     /* Doxygen can't find <FL/fl_attr.h> */
00027
00028 #undef ASSERT_UTF8
00029
00030 #ifdef ASSERT_UTF8
00031 #include <assert.h>
00032 #define IS_UTF8_ALIGNED(a) if (a && *a) assert(fl_utf8len(*a)>0);
00033 #define IS_UTF8_ALIGNED2(a, b) if (b>=0 && b<a->length()) assert(fl_utf8len(a->byte_at(b))>0);
00034 #else
00035 #define IS_UTF8_ALIGNED(a)
00036 #define IS_UTF8_ALIGNED2(a, b)
00037 #endif
00038
00039
00040 /*
00041  "character size" is the size of a UTF-8 character in bytes
00042  "character width" is the width of a Unicode character in pixels
00043  "column" was originally defined as a character offset from the left margin.
00044  It was identical to the byte offset. In UTF-8, we have neither a byte offset
00045  nor truly fixed width fonts (*). Column could be a pixel value multiplied with
00046  an average character width (which is a bearable approximation).
00047
00048  * in Unicode, there are no fixed width fonts! Even if the ASCII characters may
00049  happen to be all the same width in pixels, Chinese characters surely are not.
00050  There are plenty of exceptions, like ligatures, that make special handling of

```



```

00051     "fixed" character widths a nightmare. I decided to remove all references to
00052     fixed fonts and see "columns" as a multiple of the average width of a
00053     character in the main font.
00054     - Matthias
00055 */
00056
00057
00058 /* Maximum length in characters of a tab or control character expansion
00059    of a single buffer character */
00060 #define FL_TEXT_MAX_EXP_CHAR_LEN 20
00061
00062 #include "Fl_Export.H"
00063
00064 class Fl_Text_Undo_Action_List;
00065 class Fl_Text_Undo_Action;
00066
00067 class FL_EXPORT Fl_Text_Selection {
00068     friend class Fl_Text_Buffer;
00069
00070 public:
00071     // Sets the selection range and selected().
00072     void set(int startpos, int endpos);
00073
00074     // Updates a selection after text was modified.
00075     void update(int pos, int nDeleted, int nInserted);
00076
00077     int start() const { return mSelected ? mStart : 0; }
00078
00079     int end() const { return mSelected ? mEnd : 0; }
00080
00081     bool selected() const { return mSelected; }
00082
00083     void selected(bool b) { mSelected = b; }
00084
00085     int length() const { return mSelected ? mEnd - mStart : 0; }
00086
00087     // Returns true if position \p pos is in this Fl_Text_Selection.
00088     int includes(int pos) const;
00089
00090     // Returns true if selected() and the positions of this selection.
00091     int selected(int *startpos, int *endpos) const;
00092     FL_DEPRECATED("since 1.4.0 - use selected(startpos, endpos) instead",
00093     int position(int *startpos, int *endpos) const) { return selected(startpos, endpos); }
00094
00095 protected:
00096     int mStart;
00097     int mEnd;
00098     bool mSelected;
00099 };
00100
00101 typedef void (*Fl_Text_Modify_Cb)(int pos, int nInserted, int nDeleted,
00102     int nRestyled, const char* deletedText,
00103     void* cbArg);
00104
00105 typedef void (*Fl_Text_Predelete_Cb)(int pos, int nDeleted, void* cbArg);
00106
00107 class FL_EXPORT Fl_Text_Buffer {
00108 public:
00109     Fl_Text_Buffer(int requestedSize = 0, int preferredGapSize = 1024);
00110
00111     ~Fl_Text_Buffer();
00112
00113     int length() const { return mLength; }
00114
00115     char* text() const;
00116
00117     void text(const char* text);
00118
00119     char* text_range(int start, int end) const;
00120
00121     unsigned int char_at(int pos) const;
00122
00123     char byte_at(int pos) const;
00124
00125     const char *address(int pos) const
00126     { return (pos < mGapStart) ? mBuf+pos : mBuf+pos+mGapEnd-mGapStart; }
00127
00128     char *address(int pos)
00129     { return (pos < mGapStart) ? mBuf+pos : mBuf+pos+mGapEnd-mGapStart; }
00130
00131     void insert(int pos, const char* text, int insertedLength = -1);

```

```

00290
00296 void append(const char* t, int addedLength = -1) { insert(length(), t, addedLength); }
00297
00298 void vprintf(const char *fmt, va_list ap);
00299 void printf(const char* fmt, ...);
00300
00306 void remove(int start, int end);
00307
00316 void replace(int start, int end, const char *text, int insertedLength = -1);
00317
00325 void copy(Fl_Text_Buffer* fromBuf, int fromStart, int fromEnd, int toPos);
00326
00331 int undo(int *cp=0);
00332
00337 bool can_undo() const;
00338
00342 int redo(int *cp=0);
00343
00348 bool can_redo() const;
00349
00357 void canUndo(char flag=1);
00358
00374 int insertfile(const char *file, int pos, int buflen = 128*1024);
00375
00379 int appendfile(const char *file, int buflen = 128*1024)
00380 { return insertfile(file, length(), buflen); }
00381
00385 int loadfile(const char *file, int buflen = 128*1024)
00386 { select(0, length()); remove_selection(); return appendfile(file, buflen); }
00387
00398 int outputfile(const char *file, int start, int end, int buflen = 128*1024);
00399
00410 int savefile(const char *file, int buflen = 128*1024)
00411 { return outputfile(file, 0, length(), buflen); }
00412
00419 int tab_distance() const { return mTabDist; }
00420
00425 void tab_distance(int tabDist);
00426
00430 void select(int start, int end);
00431
00435 int selected() const { return mPrimary.selected(); }
00436
00440 void unselect();
00441
00445 int selection_position(int* start, int* end);
00446
00452 char* selection_text();
00453
00457 void remove_selection();
00458
00462 void replace_selection(const char* text);
00463
00467 void secondary_select(int start, int end);
00468
00473 int secondary_selected() { return mSecondary.selected(); }
00474
00478 void secondary_unselect();
00479
00483 int secondary_selection_position(int* start, int* end);
00484
00490 char* secondary_selection_text();
00491
00496 void remove_secondary_selection();
00497
00502 void replace_secondary_selection(const char* text);
00503
00507 void highlight(int start, int end);
00508
00512 int highlight() { return mHighlight.selected(); }
00513
00517 void unhighlight();
00518
00522 int highlight_position(int* start, int* end);
00523
00529 char* highlight_text();
00530
00542 void add_modify_callback(Fl_Text_Modify_Cb bufModifiedCB, void* cbArg);
00543
00547 void remove_modify_callback(Fl_Text_Modify_Cb bufModifiedCB, void* cbArg);
00548
00553 void call_modify_callbacks() { call_modify_callbacks(0, 0, 0, 0, 0); }
00554
00558 void add_predelete_callback(Fl_Text_Predelete_Cb bufPredeleCB, void* cbArg);
00559
00564 void remove_predelete_callback(Fl_Text_Predelete_Cb predeleCB, void* cbArg);
00565

```

```

00570 void call_predelete_callbacks() { call_predelete_callbacks(0, 0); }
00571
00580 char* line_text(int pos) const;
00581
00587 int line_start(int pos) const;
00588
00596 int line_end(int pos) const;
00597
00603 int word_start(int pos) const;
00604
00610 int word_end(int pos) const;
00611
00619 int count_displayed_characters(int lineStartPos, int targetPos) const;
00620
00630 int skip_displayed_characters(int lineStartPos, int nChars);
00631
00636 int count_lines(int startPos, int endPos) const;
00637
00643 int estimate_lines(int startPos, int endPos, int lineLen) const;
00644
00649 int skip_lines(int startPos, int nLines);
00650
00657 int rewind_lines(int startPos, int nLines);
00658
00673 int findchar_forward(int startPos, unsigned searchChar, int* foundPos) const;
00674
00688 int findchar_backward(int startPos, unsigned int searchChar, int* foundPos) const;
00689
00701 int search_forward(int startPos, const char* searchString, int* foundPos,
00702                    int matchCase = 0) const;
00703
00715 int search_backward(int startPos, const char* searchString, int* foundPos,
00716                    int matchCase = 0) const;
00717
00721 const Fl_Text_Selection* primary_selection() const { return &mPrimary; }
00722
00726 Fl_Text_Selection* primary_selection() { return &mPrimary; }
00727
00731 const Fl_Text_Selection* secondary_selection() const { return &mSecondary; }
00732
00736 const Fl_Text_Selection* highlight_selection() const { return &mHighlight; }
00737
00742 int prev_char(int ix) const;
00743 int prev_char_clipped(int ix) const;
00744
00749 int next_char(int ix) const;
00750 int next_char_clipped(int ix) const;
00751
00755 int utf8_align(int) const;
00756
00760 int input_file_was_transcoded;
00761
00765 static const char* file_encoding_warning_message;
00766
00776 void (*transcoding_warning_action)(Fl_Text_Buffer*);
00777 bool is_word_separator(int pos) const;
00778
00779 protected:
00780
00785 void call_modify_callbacks(int pos, int nDeleted, int nInserted,
00786                           int nRestyled, const char* deletedText) const;
00787
00792 void call_predelete_callbacks(int pos, int nDeleted) const;
00793
00804 int insert_(int pos, const char* text, int insertedLength = -1);
00805
00812 void remove_(int start, int end);
00813
00818 void redisplay_selection(Fl_Text_Selection* oldSelection,
00819                        Fl_Text_Selection* newSelection) const;
00820
00824 void move_gap(int pos);
00825
00830 void reallocate_with_gap(int newGapStart, int newGapLen);
00831
00832 char* selection_text_(Fl_Text_Selection* sel) const;
00833
00837 void remove_selection_(Fl_Text_Selection* sel);
00838
00842 void replace_selection_(Fl_Text_Selection* sel, const char* text);
00843
00847 void update_selections(int pos, int nDeleted, int nInserted);
00848
00852 int apply_undo(Fl_Text_Undo_Action* action, int* cursorPos);
00853
00854 Fl_Text_Selection mPrimary;
00855 Fl_Text_Selection mSecondary;

```

```

00856     Fl_Text_Selection mHighlight;
00857     int mLength;
00860     char* mBuf;
00861     int mGapStart;
00862     int mGapEnd;
00863     // The hardware tab distance used by all displays for this buffer,
00864     // and used in computing offsets for rectangular selection operations.
00865     int mTabDist;
00866     int mNModifyProcs;
00867     Fl_Text_Modify_Cb *mModifyProcs;
00869     void** mCbArgs;
00870     int mNPredeleteProcs;
00871     Fl_Text_Predelete_Cb *mPredeleteProcs;
00873     void **mPredeleteCbArgs;
00874     int mCursorPosHint;
00876     char mCanUndo;
00878     int mPreferredGapSize;
00881     Fl_Text_Undo_Action* mUndo;
00882     Fl_Text_Undo_Action_List* mUndoList;
00883     Fl_Text_Undo_Action_List* mRedoList;
00884 };
00885
00886 #endif

```

12.151 Fl_Text_Display.H

```

00001 //
00002 // Header file for Fl_Text_Display class.
00003 //
00004 // Copyright 2001-2023 by Bill Spitzak and others.
00005 // Original code Copyright Mark Edel.  Permission to distribute under
00006 // the LGPL for the FLTK library granted by Mark Edel.
00007 //
00008 // This library is free software. Distribution and use rights are outlined in
00009 // the file "COPYING" which should have been included with this file.  If this
00010 // file is missing or damaged, see the license at:
00011 //
00012 //     https://www.fltk.org/COPYING.php
00013 //
00014 // Please see the following page on how to report bugs and issues:
00015 //
00016 //     https://www.fltk.org/bugs.php
00017 //
00018
00019 /* \file
00020  Fl_Text_Display widget . */
00021
00022 #ifndef FL_TEXT_DISPLAY_H
00023 #define FL_TEXT_DISPLAY_H
00024
00025 #include <FL/Fl.H> // Fl::scrollbar_size()
00026 #include "fl_draw.H"
00027 #include "Fl_Group.H"
00028 #include "Fl_Widget.H"
00029 #include "Fl_Scrollbar.H"
00030 #include "Fl_Text_Buffer.H"
00031
00078 class FL_EXPORT Fl_Text_Display: public Fl_Group {
00079
00080 public:
00081
00085     enum {
00086         NORMAL_CURSOR,
00087         CARET_CURSOR,
00088         DIM_CURSOR,
00089         BLOCK_CURSOR,
00090         HEAVY_CURSOR,
00091         SIMPLE_CURSOR
00092     };
00093
00099     enum {
00100         CURSOR_POS,
00101         CHARACTER_POS
00102     };
00103
00109     enum {
00110         DRAG_NONE = -2,
00111         DRAG_START_DND = -1,
00112         DRAG_CHAR = 0,
00113         DRAG_WORD = 1,
00114         DRAG_LINE = 2
00115     };
00116
00120     enum {
00121         WRAP_NONE,

```

```

00122     WRAP_AT_COLUMN,
00123     WRAP_AT_PIXEL,
00124     WRAP_AT_BOUNDS
00125 };
00126
00127 friend int fl_text_drag_prepare(int pos, int key, Fl_Text_Display* d);
00128 friend void fl_text_drag_me(int pos, Fl_Text_Display* d);
00129
00130 typedef void (*Unfinished_Style_Cb)(int, void *);
00131
00132 struct Style_Table_Entry {
00133     Fl_Color    color;
00134     Fl_Font     font;
00135     Fl_Fontsize size;
00136     unsigned    attr;
00137     Fl_Color    bgcolor;
00138 };
00139
00140 enum {
00141     ATTR_BGCOLOR      = 0x0001,
00142     ATTR_BGCOLOR_EXT = 0x0002,
00143     ATTR_BGCOLOR_EXT = 0x0003,
00144     ATTR_UNDERLINE    = 0x0004,
00145     ATTR_GRAMMAR       = 0x0008,
00146     ATTR_SPELLING      = 0x000C,
00147     ATTR_STRIKE_THROUGH = 0x0010,
00148     ATTR_LINES_MASK    = 0x001C
00149 };
00150
00151 Fl_Text_Display(int X, int Y, int W, int H, const char *l = 0);
00152 ~Fl_Text_Display();
00153
00154 int handle(int e) FL_OVERRIDE;
00155
00156 void buffer(Fl_Text_Buffer* buf);
00157
00158 void buffer(Fl_Text_Buffer& buf) { buffer(&buf); }
00159
00160 Fl_Text_Buffer* buffer() const { return mBuffer; }
00161 Fl_Text_Buffer* style_buffer() const { return mStyleBuffer; }
00162
00163 void redisplay_range(int start, int end);
00164 void scroll(int topLineNum, int horizOffset);
00165 void insert(const char* text);
00166 void overstrike(const char* text);
00167 void insert_position(int newPos);
00168
00169 int insert_position() const { return mCursorPos; }
00170 int position_to_xy(int pos, int* x, int* y) const;
00171
00172 int in_selection(int x, int y) const;
00173 void show_insert_position();
00174
00175 int move_right();
00176 int move_left();
00177 int move_up();
00178 int move_down();
00179 int count_lines(int start, int end, bool start_pos_is_line_start) const;
00180 int line_start(int pos) const;
00181 int line_end(int startPos, bool startPosIsLineStart) const;
00182 int skip_lines(int startPos, int nLines, bool startPosIsLineStart);
00183 int rewind_lines(int startPos, int nLines);
00184 void next_word(void);
00185 void previous_word(void);
00186
00187 void show_cursor(int b = 1);
00188
00189 void hide_cursor() { show_cursor(0); }
00190
00191 void cursor_style(int style);
00192 int cursor_style() const { return mCursorStyle; }
00193
00194 Fl_Color cursor_color() const { return mCursor_color; }
00195
00196 void cursor_color(Fl_Color n) { mCursor_color = n; }
00197
00198 int scrollbar_width() const {
00199     return scrollbar_width_ ? scrollbar_width_ : Fl::scrollbar_size();
00200 }
00201
00202 void scrollbar_width(int width) {
00203     Fl::scrollbar_size(width);
00204     scrollbar_width_ = 0;
00205 }
00206
00207 int scrollbar_size() const {
00208     return(scrollbar_width_);

```

```

00285     }
00286
00306 void scrollbar_size(int newSize) {
00307     scrollbar_width_ = newSize;
00308 }
00309
00314 Fl_Align scrollbar_align() const { return scrollbar_align_; }
00315
00320 void scrollbar_align(Fl_Align a) { scrollbar_align_ = a; }
00321
00327 int word_start(int pos) const { return buffer()->word_start(pos); }
00328
00334 int word_end(int pos) const { return buffer()->word_end(pos); }
00335
00336
00337 void highlight_data(Fl_Text_Buffer *styleBuffer,
00338                    const Style_Table_Entry *styleTable,
00339                    int nStyles, char unfinishedStyle,
00340                    Unfinished_Style_Cb unfinishedHighlightCB,
00341                    void *cbArg);
00342
00343 int position_style(int lineStartPos, int lineLen, int lineIndex) const;
00344
00350 int shortcut() const {return shortcut_;}
00351
00357 void shortcut(int s) {shortcut_ = s;}
00358
00363 Fl_Font textfont() const {return textfont_;}
00364
00369 void textfont(Fl_Font s) {textfont_ = s; mColumnScale = 0; }
00370
00375 Fl_Fontsize textsize() const {return textsize_;}
00376
00381 void textsize(Fl_Fontsize s) {textsize_ = s; mColumnScale = 0;}
00382
00387 Fl_Color textcolor() const {return textcolor_;}
00388
00393 void textcolor(Fl_Color n) {textcolor_ = n;}
00394
00399 void grammar_underline_color(Fl_Color color) { grammar_underline_color_ = color; }
00400
00405 Fl_Color grammar_underline_color() const { return grammar_underline_color_;}
00406
00411 void spelling_underline_color(Fl_Color color) { spelling_underline_color_ = color; }
00412
00417 Fl_Color spelling_underline_color() const { return spelling_underline_color_;}
00418
00423 void secondary_selection_color(Fl_Color color) { secondary_selection_color_ = color; }
00424
00429 Fl_Color secondary_selection_color() const { return secondary_selection_color_;}
00430
00431 int wrapped_column(int row, int column) const;
00432 int wrapped_row(int row) const;
00433 void wrap_mode(int wrap, int wrap_margin);
00434
00435 virtual void recalc_display();
00436 virtual void display_needs_recalc();
00437 void resize(int X, int Y, int W, int H) FL_OVERRIDE;
00438
00444 double x_to_col(double x) const;
00445
00452 double col_to_x(double col) const;
00453
00454 void linenum_width(int width);
00455 int linenum_width() const;
00456 void linenum_font(Fl_Font val);
00457 Fl_Font linenum_font() const;
00458 void linenum_size(Fl_Fontsize val);
00459 Fl_Fontsize linenum_size() const;
00460 void linenum_fgcolor(Fl_Color val);
00461 Fl_Color linenum_fgcolor() const;
00462 void linenum_bgcolor(Fl_Color val);
00463 Fl_Color linenum_bgcolor() const;
00464 void linenum_align(Fl_Align val);
00465 Fl_Align linenum_align() const;
00466 void linenum_format(const char* val);
00467 const char* linenum_format() const;
00468
00469 protected:
00470 // Most (all?) of this stuff should only be called from resize() or
00471 // draw().
00472 // Anything with "vline" indicates that it deals with currently
00473 // visible lines.
00474
00475 void draw() FL_OVERRIDE;
00476 void draw_text(int X, int Y, int W, int H);
00477 void draw_range(int start, int end);

```

```

00478 void draw_cursor(int, int);
00479
00480 void draw_string(int style, int x, int y, int toX, const char *string,
00481                 int nChars) const;
00482
00483 void draw_vline(int visLineNum, int leftClip, int rightClip,
00484                 int leftCharIndex, int rightCharIndex);
00485
00486 int find_x(const char *s, int len, int style, int x) const;
00487
00488 enum {
00489     DRAW_LINE,
00490     FIND_INDEX,
00491     FIND_INDEX_FROM_ZERO,
00492     GET_WIDTH,
00493     FIND_CURSOR_INDEX    // STR #2788
00494 };
00495
00496 int handle_vline(int mode,
00497                  int lineStart, int lineLen, int leftChar, int rightChar,
00498                  int topClip, int bottomClip,
00499                  int leftClip, int rightClip) const;
00500
00501 int handle_rmb(int readonly);
00502
00503 void draw_line_numbers(bool clearAll);
00504
00505 void clear_rect(int style, int x, int y, int width, int height) const;
00506 void display_insert();
00507
00508 void offset_line_starts(int newTopLineNum);
00509
00510 void calc_line_starts(int startLine, int endLine);
00511
00512 void update_line_starts(int pos, int charsInserted, int charsDeleted,
00513                         int linesInserted, int linesDeleted, int *scrolled);
00514
00515 void calc_last_char();
00516
00517 int position_to_line( int pos, int* lineNum ) const;
00518 double string_width(const char* string, int length, int style) const;
00519
00520 static void scroll_timer_cb(void*);
00521
00522 static void buffer_predelete_cb(int pos, int nDeleted, void* cbArg);
00523 static void buffer_modified_cb(int pos, int nInserted, int nDeleted,
00524                                int nRestyled, const char* deletedText,
00525                                void* cbArg);
00526
00527 static void h_scrollbar_cb(Fl_Scrollbar* w, Fl_Text_Display* d);
00528 static void v_scrollbar_cb( Fl_Scrollbar* w, Fl_Text_Display* d);
00529 void update_v_scrollbar();
00530 void update_h_scrollbar();
00531 int measure_vline(int visLineNum) const;
00532 int longest_vline() const;
00533 int empty_vlines() const;
00534 int vline_length(int visLineNum) const;
00535 int xy_to_position(int x, int y, int PosType = CHARACTER_POS) const;
00536
00537 void xy_to_rowcol(int x, int y, int* row, int* column,
00538                  int PosType = CHARACTER_POS) const;
00539 void maintain_absolute_top_line_number(int state);
00540 public:
00541 int get_absolute_top_line_number() const;
00542 int scroll_row() { return mTopLineNum; }
00543 int scroll_col() { return mHorizOffset; }
00544 protected:
00545 void absolute_top_line_number(int oldFirstChar);
00546 int maintaining_absolute_top_line_number() const;
00547 void reset_absolute_top_line_number();
00548 int position_to_linecol(int pos, int* lineNum, int* column) const;
00549 int scroll_(int topLineNum, int horizOffset);
00550
00551 void extend_range_for_styles(int* start, int* end);
00552
00553 void find_wrap_range(const char *deletedText, int pos, int nInserted,
00554                     int nDeleted, int *modRangeStart, int *modRangeEnd,
00555                     int *linesInserted, int *linesDeleted);
00556 void measure_deleted_lines(int pos, int nDeleted);
00557 void wrapped_line_counter(Fl_Text_Buffer *buf, int startPos, int maxPos,
00558                           int maxLines, bool startPosIsLineStart,
00559                           int styleBufOffset, int *retPos, int *retLines,
00560                           int *retLineStart, int *retLineEnd,
00561                           bool countLastLineMissingNewLine = true) const;
00562 void find_line_end(int pos, bool start_pos_is_line_start, int *lineEnd,
00563                   int *nextLineStart) const;
00564 double measure_proportional_character(const char *s, int colNum, int pos) const;

```

```

00565 int wrap_uses_character(int lineEndPos) const;
00566
00567 int damage_range1_start, damage_range1_end;
00568 int damage_range2_start, damage_range2_end;
00569 int mCursorPos;
00570 int mCursorOn;
00571 int mCursorOldY;           /* Y pos. of cursor for blanking */
00572 int mCursorToHint;        /* Tells the buffer modified callback
00573                            where to move the cursor, to reduce
00574                            the number of redraw calls */
00575 int mCursorStyle;         /* One of enum cursorStyles above */
00576 int mCursorPreferredXPos; /* Pixel position for vert. cursor movement */
00577 int mNVisibleLines;       /* # of visible (displayed) lines. This is
00578                            also the size of the mLineStarts[] array. */
00579 int mNBufferLines;        /* # of newlines in the buffer, or number of
00580                            wraps if line wrapping is enabled. Note that
00581                            partial lines at the end of the buffer are
00582                            not counted, so you may want to add 1. */
00583 Fl_Text_Buffer* mBuffer;   /* Contains text to be displayed */
00584 Fl_Text_Buffer* mStyleBuffer; /* Optional parallel buffer containing
00585                            color and font information */
00586 int mFirstChar, mLastChar; /* Buffer positions of first and last
00587                            displayed character (lastChar points
00588                            either to a newline or one character
00589                            beyond the end of the buffer) */
00590 int mContinuousWrap;      /* Wrap long lines when displaying */
00591 int mWrapMarginPix;       /* Margin in # of pixels for
00592                            wrapping in continuousWrap mode */
00593 int* mLineStarts;         /* Array of the size mNVisibleLines.
00594                            This array only keeps track of lines
00595                            within the display area. Each entry
00596                            contains the starting character offset
00597                            (from the beginning of the text buffer)
00598                            for each /visible/ line.
00599                            If wrap enabled, points to the beginning
00600                            of each wrap. So a long line wrapping
00601                            into 3 separate lines in the display
00602                            will take up 3 separate array entries. */
00603 int mTopLineNum;          /* Line number of top displayed line
00604                            of file (first line of file is 1) */
00605 int mAbsTopLineNum;       /* In continuous wrap mode, the line
00606                            number of the top line if the text
00607                            were not wrapped (note that this is
00608                            only maintained as needed). */
00609 int mNeedAbsTopLineNum;   /* Externally settable flag to continue
00610                            maintaining absTopLineNum even if
00611                            it isn't needed for line # display */
00612 int mHorizOffset;         /* Horizontal scroll pos. in pixels */
00613 int mTopLineNumHint;      /* Line number of top displayed line
00614                            of file (first line of file is 1) */
00615 int mHorizOffsetHint;     /* Horizontal scroll pos. in pixels */
00616 int mNStyles;             /* Number of entries in styleTable */
00617 const Style_Table_Entry *mStyleTable; /* Table of fonts and colors for
00618                            coloring/syntax-highlighting */
00619 char mUnfinishedStyle;    /* Style buffer entry which triggers
00620                            on-the-fly reparsing of region */
00621 Unfinished_Style_Cb mUnfinishedHighlightCB; /* Callback to parse "unfinished" */
00622 /* regions */
00623 void* mHighlightCBArg;    /* Arg to unfinishedHighlightCB */
00624
00625 int mMaxsize;
00626
00627 int mSuppressResync;      /* Suppress resynchronization of line
00628                            starts during buffer updates */
00629 int mNLinesDeleted;       /* Number of lines deleted during
00630                            buffer modification (only used
00631                            when resynchronization is suppressed) */
00632 int mModifyingTabDistance; /* Whether tab distance is being modified XXX: UNUSED */
00633
00634 mutable double mColumnScale; /* Width in pixels of an average character. This
00635                            value is calculated as needed (lazy eval); it
00636                            needs to be mutable so that it can be calculated
00637                            within a method marked as "const" */
00638
00639 bool display_needs_recalc; /* Set to true when the display needs
00640                            to be recalculated. */
00641
00642 Fl_Color mCursor_color;
00643
00644 Fl_Scrollbar* mHScrollBar;
00645 Fl_Scrollbar* mVScrollBar;
00646 int scrollbar_width;      /* size of scrollbar trough (behavior changed in 1.4)
00647 Fl_Align scrollbar_align;
00648 int dragPos, dragType, dragging;
00649 int display_insert_position_hint;
00650 struct { int x, y, w, h; } text_area;
00651

```



```

00652     int shortcut_;
00653
00654     Fl_Font textfont_;
00655     Fl_Fonsize textsize_;
00656     Fl_Color textcolor_;
00657     Fl_Color grammar_underline_color_;
00658     Fl_Color spelling_underline_color_;
00659     Fl_Color secondary_selection_color_;
00660
00661     // Line number margin and width
00662     int mLineNumLeft, mLineNumWidth;
00663
00664     // Line number font/colors
00665     Fl_Font    linenum_font_;
00666     Fl_Fonsize linenum_size_;
00667     Fl_Color    linenum_fgcolor_;
00668     Fl_Color    linenum_bgcolor_;
00669     Fl_Align    linenum_align_;
00670     const char* linenum_format_;
00671 };
00672
00673 #endif

```

12.152 Fl_Text_Editor.H

```

00001 //
00002 // Header file for Fl_Text_Editor class.
00003 //
00004 // Copyright 2001-2023 by Bill Spitzak and others.
00005 // Original code Copyright Mark Edel.  Permission to distribute under
00006 // the LGPL for the FLTK library granted by Mark Edel.
00007 //
00008 // This library is free software. Distribution and use rights are outlined in
00009 // the file "COPYING" which should have been included with this file.  If this
00010 // file is missing or damaged, see the license at:
00011 //
00012 //     https://www.fltk.org/COPYING.php
00013 //
00014 // Please see the following page on how to report bugs and issues:
00015 //
00016 //     https://www.fltk.org/bugs.php
00017 //
00018
00019 /* \file
00020    Fl_Text_Editor widget . */
00021
00022
00023 #ifndef FL_TEXT_EDITOR_H
00024 #define FL_TEXT_EDITOR_H
00025
00026 #include "Fl_Text_Display.H"
00027
00028 // key will match in any state
00029 #define FL_TEXT_EDITOR_ANY_STATE  (-1L)
00030
00031 class FL_EXPORT Fl_Text_Editor : public Fl_Text_Display {
00032 public:
00033     typedef int (*Key_Func)(int key, Fl_Text_Editor* editor);
00034
00035     struct Key_Binding {
00036         int    key;
00037         int    state;
00038         Key_Func    function;
00039         Key_Binding* next;
00040     };
00041
00042     Fl_Text_Editor(int X, int Y, int W, int H, const char* l = 0);
00043     ~Fl_Text_Editor() { remove_all_key_bindings(); }
00044     int handle(int e) FL_OVERRIDE;
00045     void insert_mode(int b) { insert_mode_ = b; }
00046     int insert_mode() { return insert_mode_; }
00047     void tab_nav(int val);
00048     int tab_nav() const;
00049     void add_key_binding(int key, int state, Key_Func f, Key_Binding** list);
00050     void add_key_binding(int key, int state, Key_Func f)
00051     { add_key_binding(key, state, f, &key_bindings); }
00052     void remove_key_binding(int key, int state, Key_Binding** list);
00053     void remove_key_binding(int key, int state)
00054     { remove_key_binding(key, state, &key_bindings); }
00055     void remove_all_key_bindings(Key_Binding** list);
00056     void remove_all_key_bindings() { remove_all_key_bindings(&key_bindings); }
00057     void add_default_key_bindings(Key_Binding** list);
00058     Key_Func bound_key_function(int key, int state, Key_Binding* list) const;
00059     Key_Func bound_key_function(int key, int state) const
00060     { return bound_key_function(key, state, key_bindings); }

```

```

00085     void default_key_function(Key_Func f) { default_key_function_ = f; }
00086
00087     // functions for the built in default bindings
00088     static int kf_default(int c, Fl_Text_Editor* e);
00089     static int kf_ignore(int c, Fl_Text_Editor* e);
00090     static int kf_backspace(int c, Fl_Text_Editor* e);
00091     static int kf_enter(int c, Fl_Text_Editor* e);
00092     static int kf_move(int c, Fl_Text_Editor* e);
00093     static int kf_shift_move(int c, Fl_Text_Editor* e);
00094     static int kf_ctrl_move(int c, Fl_Text_Editor* e);
00095     static int kf_c_s_move(int c, Fl_Text_Editor* e);
00096     static int kf_meta_move(int c, Fl_Text_Editor* e);
00097     static int kf_m_s_move(int c, Fl_Text_Editor* e);
00098     static int kf_home(int, Fl_Text_Editor* e);
00099     static int kf_end(int c, Fl_Text_Editor* e);
00100     static int kf_left(int c, Fl_Text_Editor* e);
00101     static int kf_up(int c, Fl_Text_Editor* e);
00102     static int kf_right(int c, Fl_Text_Editor* e);
00103     static int kf_down(int c, Fl_Text_Editor* e);
00104     static int kf_page_up(int c, Fl_Text_Editor* e);
00105     static int kf_page_down(int c, Fl_Text_Editor* e);
00106     static int kf_insert(int c, Fl_Text_Editor* e);
00107     static int kf_delete(int c, Fl_Text_Editor* e);
00108     static int kf_copy(int c, Fl_Text_Editor* e);
00109     static int kf_cut(int c, Fl_Text_Editor* e);
00110     static int kf_paste(int c, Fl_Text_Editor* e);
00111     static int kf_select_all(int c, Fl_Text_Editor* e);
00112     static int kf_undo(int c, Fl_Text_Editor* e);
00113     static int kf_redo(int c, Fl_Text_Editor* e);
00114
00115     protected:
00116         int handle_key();
00117         void maybe_do_callback(Fl_Callback_Reason reason = FL_REASON_CHANGED);
00118
00119     #ifndef FL_DOXYGEN
00120         int insert_mode_;
00121         Key_Binding* key_bindings;
00122     #endif
00123
00131     static Key_Binding* global_key_bindings;
00132
00133     #ifndef FL_DOXYGEN
00134         Key_Func default_key_function_;
00135     #endif
00136 };
00137
00138 #endif
00139

```

12.153 Fl_Tile.H

```

00001 //
00002 // Tile header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Tile_H
00018 #define Fl_Tile_H
00019
00020 #include "Fl_Group.H"
00021
00022 /*
00023  The Fl_Tile class lets you resize its children by dragging
00024  the border between them.
00025 */
00026
00027 class FL_EXPORT Fl_Tile : public Fl_Group {
00028 public:
00029     int handle(int event) FL_OVERRIDE;
00030     Fl_Tile(int X, int Y, int W, int H, const char *L=0);
00031     ~Fl_Tile() FL_OVERRIDE;
00032     void resize(int X, int Y, int W, int H) FL_OVERRIDE;
00033     virtual void move_intersection(int oldx, int oldy, int newx, int newy);
00034     virtual void drag_intersection(int oldx, int oldy, int newx, int newy);

```

```

00035  FL_DEPRECATED("since 1.4.0 - use move_intersection(p) instead",
00036  void position(int oldx, int oldy, int newx, int newy)) { move_intersection(oldx, oldy, newx, newy);
    }
00037  void position(int x, int y) { Fl_Group::position(x, y); }
00038  void size_range(int index, int minw, int minh, int maxw=0x7FFFFFFF, int maxh=0x7FFFFFFF);
00039  void size_range(Fl_Widget *w, int minw, int minh, int maxw=0x7FFFFFFF, int maxh=0x7FFFFFFF);
00040  void init_size_range(int default_min_w = -1, int default_min_h = -1);
00041
00042  protected:
00043  int cursor_;
00044  Fl_Cursor *cursors_;
00045
00050  Fl_Cursor cursor(int n) {
00051  return cursors_[n];
00052  }
00053
00054  void set_cursor(int n); // set one of n (0..3) cursors
00055
00056  typedef struct { int minw, minh, maxw, maxh; } Size_Range;
00057
00058  Size_Range *size_range_;
00059  int size_range_size_, size_range_capacity_;
00060  int default_min_w_, default_min_h_;
00061  void request_shrink_l(int old_l, int &new_l, Fl_Rect *final_size);
00062  void request_shrink_r(int old_r, int &new_r, Fl_Rect *final_size);
00063  void request_shrink_t(int old_t, int &new_t, Fl_Rect *final_size);
00064  void request_shrink_b(int old_b, int &new_b, Fl_Rect *final_size);
00065  void request_grow_l(int old_l, int &new_l, Fl_Rect *final_size);
00066  void request_grow_r(int old_r, int &new_r, Fl_Rect *final_size);
00067  void request_grow_t(int old_t, int &new_t, Fl_Rect *final_size);
00068  void request_grow_b(int old_b, int &new_b, Fl_Rect *final_size);
00069
00070  int on_insert(Fl_Widget*, int) FL_OVERRIDE;
00071  int on_move(int, int) FL_OVERRIDE;
00072  void on_remove(int) FL_OVERRIDE;
00073 };
00074
00075 #endif

```

12.154 Fl_Tiled_Image.H

```

00001 //
00002 // Tiled image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2015 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 // https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 // https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018  Fl_Tiled_Image widget . */
00019
00020 #ifndef Fl_Tiled_Image_H
00021 # define Fl_Tiled_Image_H
00022
00023 # include "Fl_Image.H"
00024
00025
00032 class FL_EXPORT Fl_Tiled_Image : public Fl_Image {
00033 protected:
00034
00035  Fl_Image      *image_; // The image that is tiled
00036  int           alloc_image_; // Did we allocate this image?
00037
00038 public:
00039  Fl_Tiled_Image(Fl_Image *i, int W = 0, int H = 0);
00040  virtual ~Fl_Tiled_Image();
00041
00042  Fl_Image *copy(int W, int H) const FL_OVERRIDE;
00043  Fl_Image *copy() const {
00044  return Fl_Image::copy();
00045  }
00046  void color_average(Fl_Color c, float i) FL_OVERRIDE;
00047  void desaturate() FL_OVERRIDE;
00048  void draw(int X, int Y, int W, int H, int cx = 0, int cy = 0) FL_OVERRIDE;
00049  void draw(int X, int Y) { draw(X, Y, w(), h(), 0, 0); }
00051  Fl_Image *image() { return image_; }

```

```

00052 };
00053
00054 #endif // !Fl_Tiled_Image_H

```

12.155 Fl_Timer.H

```

00001 //
00002 // Timer header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Timer widget . */
00019
00020 #ifndef Fl_Timer_H
00021 #define Fl_Timer_H
00022
00023 #ifndef Fl_Widget_H
00024 #include "Fl_Widget.H"
00025 #endif
00026
00027 // values for type():
00028 #define FL_NORMAL_TIMER      0
00029 #define FL_VALUE_TIMER      1
00030 #define FL_HIDDEN_TIMER     2
00031
00032 class FL_EXPORT Fl_Timer : public Fl_Widget {
00033     static void stepcb(void *);
00034     void step();
00035     char on, direction_;
00036     double delay, total;
00037     long lastsec, lastusec;
00038 protected:
00039     void draw() FL_OVERRIDE;
00040 public:
00041     int handle(int) FL_OVERRIDE;
00042     Fl_Timer(uchar t, int x, int y, int w, int h, const char *l);
00043     ~Fl_Timer();
00044     void value(double);
00045     double value() const {return delay>0.0?delay:0.0;}
00046     char direction() const {return direction_;}
00047     void direction(char d) {direction_ = d;}
00048     char suspended() const {return !on;}
00049     void suspended(char d);
00050 };
00051
00052 #endif
00053
00054

```

12.156 Fl_Toggle_Button.H

```

00001 //
00002 // Toggle button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Toggle_Button widget . */
00019
00020 #ifndef Fl_Toggle_Button_H

```

```
00021 #define Fl_Toggle_Button_H
00022
00023 #include "Fl_Button.H"
00024
00034 class FL_EXPORT Fl_Toggle_Button : public Fl_Button {
00035 public:
00041     Fl_Toggle_Button(int X,int Y,int W,int H,const char *l=0);
00042 };
00043
00044 #endif
```

12.157 Fl_Toggle_Light_Button.H

```
00001 //
00002 // Toggle light button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00017 // provided for back-compatibility only
00018
00019 #ifndef Fl_Toggle_Light_Button
00020 #include "Fl_Light_Button.H"
00021 #define Fl_Toggle_Light_Button Fl_Light_Button
00022 #endif
```

12.158 Fl_Toggle_Round_Button.H

```
00001 //
00002 // Toggle round button header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00017 // provided for back-compatibility only
00018
00019 #ifndef Fl_Toggle_Round_Button
00020 #include "Fl_Round_Button.H"
00021 #define Fl_Toggle_Round_Button Fl_Round_Button
00022 #endif
```

12.159 Fl_Tooltip.H

```
00001 //
00002 // Tooltip header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2011 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
```

```

00017 /* \file
00018     Fl_Tooltip widget . */
00019
00020 #ifndef FL_TOOLTIP_H
00021 #define FL_TOOLTIP_H
00022
00023 #include <FL/Fl.H>
00024 #include <FL/Fl_Widget.H>
00025
00034 class FL_EXPORT Fl_Tooltip {
00035     friend class Fl_TooltipBox;
00036 public:
00038     static float delay() { return delay_; }
00040     static void delay(float f) { delay_ = f; }
00042     static float hidedelay() { return hidedelay_; }
00044     static void hidedelay(float f) { hidedelay_ = f; }
00049     static float hoverdelay() { return hoverdelay_; }
00054     static void hoverdelay(float f) { hoverdelay_ = f; }
00056     static int enabled() { return Fl::option(FL::OPTION_SHOW_TOOLTIPS); }
00058     static void enable(int b = 1) { Fl::option(FL::OPTION_SHOW_TOOLTIPS, (b!=0)); }
00060     static void disable() { enable(0); }
00061     static void (*enter)(Fl_Widget* w);
00062     static void enter_area(Fl_Widget* w, int X, int Y, int W, int H, const char* tip);
00063     static void (*exit)(Fl_Widget *w);
00065     static Fl_Widget* current() { return widget_; }
00066     static void current(Fl_Widget*);
00067
00069     static Fl_Font font() { return font_; }
00071     static void font(Fl_Font i) { font_ = i; }
00073     static Fl_Fonsize size() { return (size_ == -1 ? FL_NORMAL_SIZE : size_); }
00075     static void size(Fl_Fonsize s) { size_ = s; }
00077     static Fl_Color color() { return color_; }
00079     static void color(Fl_Color c) { color_ = c; }
00081     static Fl_Color textcolor() { return textcolor_; }
00083     static void textcolor(Fl_Color c) { textcolor_ = c; }
00085     static int margin_width() { return margin_width_; }
00087     static void margin_width(int v) { margin_width_ = v; }
00089     static int margin_height() { return margin_height_; }
00091     static void margin_height(int v) { margin_height_ = v; }
00093     static int wrap_width() { return wrap_width_; }
00095     static void wrap_width(int v) { wrap_width_ = v; }
00097     static Fl_Window* current_window(void);
00098
00099     // These should not be public, but Fl_Widget::tooltip() needs them...
00100     // fabien: made it private with only a friend function access
00101 private:
00102     friend void Fl_Widget::tooltip(const char *);
00103     friend void Fl_Widget::copy_tooltip(const char *);
00104     static void enter_(Fl_Widget* w);
00105     static void exit_(Fl_Widget *w);
00106     static void set_enter_exit_once_();
00107
00108 private:
00109     static float delay_;
00110     static float hidedelay_;
00111     static float hoverdelay_;
00112     static Fl_Color color_;
00113     static Fl_Color textcolor_;
00114     static Fl_Font font_;
00115     static Fl_Fonsize size_;
00116     static Fl_Widget* widget_;
00117     static int margin_width_;
00118     static int margin_height_;
00119     static int wrap_width_;
00120     static const int draw_symbols_; // 1 = draw @-symbols in tooltips, 0 = no
00121 };
00122
00123 #endif

```

12.160 Fl_Tree.H File Reference

This file contains the definitions of the [Fl_Tree](#) class.

```

#include <FL/Fl.H>
#include <FL/Fl_Group.H>
#include <FL/Fl_Scrollbar.H>
#include <FL/fl_draw.H>
#include <FL/Fl_Tree_Item.H>
#include <FL/Fl_Tree_Prefs.H>

```

Classes

- class [Fl_Tree](#)
Tree widget.

Enumerations

- enum [Fl_Tree_Reason](#) {
[FL_TREE_REASON_NONE](#) = FL_REASON_UNKNOWN , [FL_TREE_REASON_SELECTED](#) = FL_REASON_SELECTED , [FL_TREE_REASON_DESELECTED](#) = FL_REASON_DESELECTED , [FL_TREE_REASON_RESELECTED](#) = FL_REASON_RESELECTED ,
[FL_TREE_REASON_OPENED](#) = FL_REASON_OPENED , [FL_TREE_REASON_CLOSED](#) = FL_REASON_CLOSED , [FL_TREE_REASON_DRAGGED](#) = FL_REASON_DRAGGED }
The reason the callback was invoked.

12.160.1 Detailed Description

This file contains the definitions of the [Fl_Tree](#) class.

12.160.2 Enumeration Type Documentation

12.160.2.1 Fl_Tree_Reason

enum [Fl_Tree_Reason](#)

The reason the callback was invoked.

Enumerator

FL_TREE_REASON_NONE	unknown reason
FL_TREE_REASON_SELECTED	an item was selected
FL_TREE_REASON_DESELECTED	an item was de-selected
FL_TREE_REASON_RESELECTED	an item was re-selected (double-clicked). See Fl_Tree_Item_Reselect_Mode to enable this.
FL_TREE_REASON_OPENED	an item was opened
FL_TREE_REASON_CLOSED	an item was closed
FL_TREE_REASON_DRAGGED	an item was dragged into a new place

12.161 Fl_Tree.H

[Go to the documentation of this file.](#)

```

00001 //
00002
00003 #ifndef FL_TREE_H
00004 #define FL_TREE_H
00005
00006 #include <FL/Fl.H>
00007 #include <FL/Fl_Group.H>
00008 #include <FL/Fl_Scrollbar.H>
00009 #include <FL/fl_draw.H>
00010
00011 #include <FL/Fl_Tree_Item.H>
00012 #include <FL/Fl_Tree_Prefs.H>
00013
00015 // FL/Fl_Tree.H
00017 //
00018 // Fl_Tree -- This file is part of the Fl_Tree widget for FLTK
00019 // Copyright (C) 2009-2010 by Greg Ercolano.
00020 //
00021 // This library is free software. Distribution and use rights are outlined in
00022 // the file "COPYING" which should have been included with this file. If this
00023 // file is missing or damaged, see the license at:
00024 //
00025 //      https://www.fltk.org/COPYING.php
00026 //

```

```

00027 // Please see the following page on how to report bugs and issues:
00028 //
00029 //     https://www.fltk.org/bugs.php
00030 //
00031
00036
00275 enum Fl_Tree_Reason {
00276     FL_TREE_REASON_NONE           = FL_REASON_UNKNOWN,
00277     FL_TREE_REASON_SELECTED       = FL_REASON_SELECTED,
00278     FL_TREE_REASON_DESELECTED     = FL_REASON_DESELECTED,
00279     FL_TREE_REASON_RESELECTED     = FL_REASON_RESELECTED,
00281     FL_TREE_REASON_OPENED         = FL_REASON_OPENED,
00282     FL_TREE_REASON_CLOSED         = FL_REASON_CLOSED,
00283     FL_TREE_REASON_DRAGGED        = FL_REASON_DRAGGED
00284 };
00285
00286 class FL_EXPORT Fl_Tree : public Fl_Group {
00287     friend class Fl_Tree_Item;
00288     Fl_Tree_Item *_root;                // can be null!
00289     Fl_Tree_Item *_item_focus;          // item that has focus box
00290     Fl_Tree_Item *_callback_item;       // item invoked during callback (can be NULL)
00291     Fl_Tree_Reason _callback_reason;    // reason for the callback
00292     Fl_Tree_Prefs _prefs;               // all the tree's settings
00293     int _scrollbar_size;                // size of scrollbar trough
00294     Fl_Tree_Item *_lastselect;          // last selected item
00295     char _lastpushed;                   // FL_PUSH occurred on: 0=nothing, 1=open/close,
                                         // 2=usericon, 3=label
00296     void fix_scrollbar_order();
00297
00298 protected:
00299     Fl_Scrollbar *_vscroll;
00300     Fl_Scrollbar *_hscroll;
00301     int _tox, _toy, _tow, _toh;
00302     int _tix, _tiy, _tiw, _tih;
00303
00305     int _tree_w;
00307     int _tree_h;
00308     void item_clicked(Fl_Tree_Item* val);
00309     void do_callback_for_item(Fl_Tree_Item* item, Fl_Tree_Reason reason);
00310
00311     // next_visible_item() and extend_selection() moved to 'public' in ABI 1.3.3
00312     // undocumented draw_tree() dropped -- draw() does all the work now
00313
00314     // draw() has to be protected per FLTK convention (was public in 1.3.x)
00315     void draw() FL_OVERRIDE;
00316
00317 public:
00318     Fl_Tree(int X, int Y, int W, int H, const char *L=0);
00319     ~Fl_Tree();
00320     int handle(int e) FL_OVERRIDE;
00321     void show_self();
00322     void resize(int,int,int,int) FL_OVERRIDE;
00323
00325     // root methods
00327     void root_label(const char *new_label);
00328     Fl_Tree_Item* root();
00329     void root(Fl_Tree_Item *newitem);
00330     const Fl_Tree_Prefs& prefs() const { return _prefs; }
00331
00333     // Item creation/removal methods
00335     Fl_Tree_Item *add(const char *path, Fl_Tree_Item *newitem=0);
00336     Fl_Tree_Item* add(Fl_Tree_Item *parent_item, const char *name);
00337     Fl_Tree_Item *insert_above(Fl_Tree_Item *above, const char *name);
00338     Fl_Tree_Item* insert(Fl_Tree_Item *item, const char *name, int pos);
00339     int remove(Fl_Tree_Item *item);
00340     void clear();
00341     void clear_children(Fl_Tree_Item *item);
00342
00344     // Item lookup methods
00346     Fl_Tree_Item *find_item(const char *path);
00347     const Fl_Tree_Item *find_item(const char *path) const;
00348     int item_pathname(char *pathname, int pathnamelen, const Fl_Tree_Item *item) const;
00349     const Fl_Tree_Item* find_clicked(int yonly=0) const;
00350     Fl_Tree_Item* find_clicked(int yonly=0);
00351     Fl_Tree_Item *item_clicked();
00352     Fl_Tree_Item *first();
00353     Fl_Tree_Item *first_visible(); // deprecated in ABI 10303
00354     Fl_Tree_Item *first_visible_item();
00355     Fl_Tree_Item *next(Fl_Tree_Item *item=0);
00356     Fl_Tree_Item *prev(Fl_Tree_Item *item=0);
00357     Fl_Tree_Item *last();
00358     Fl_Tree_Item *last_visible(); // deprecated in ABI 10303
00359     Fl_Tree_Item *last_visible_item();
00360     Fl_Tree_Item *next_visible_item(Fl_Tree_Item *start, int dir); // made public in 1.3.3 ABI
00361     Fl_Tree_Item *first_selected_item();
00362     Fl_Tree_Item *last_selected_item();
00363     Fl_Tree_Item *next_item(Fl_Tree_Item *item, int dir=FL_Down, bool visible=false);

```



```

00364 Fl_Tree_Item *next_selected_item(Fl_Tree_Item *item=0, int dir=FL_Down);
00365 int get_selected_items(Fl_Tree_Item_Array &items);
00366
00368 // Item open/close methods
00370 int open(Fl_Tree_Item *item, int docallback=1);
00371 int open(const char *path, int docallback=1);
00372 void open_toggle(Fl_Tree_Item *item, int docallback=1);
00373 int close(Fl_Tree_Item *item, int docallback=1);
00374 int close(const char *path, int docallback=1);
00375 int is_open(Fl_Tree_Item *item) const;
00376 int is_open(const char *path) const;
00377 int is_close(Fl_Tree_Item *item) const;
00378 int is_close(const char *path) const;
00379
00381 // Item selection methods
00383 int select(Fl_Tree_Item *item, int docallback=1);
00384 int select(const char *path, int docallback=1);
00385 void select_toggle(Fl_Tree_Item *item, int docallback=1);
00386 int deselect(Fl_Tree_Item *item, int docallback=1);
00387 int deselect(const char *path, int docallback=1);
00388 int deselect_all(Fl_Tree_Item *item=0, int docallback=1);
00389 int select_only(Fl_Tree_Item *selitem, int docallback=1);
00390 int select_all(Fl_Tree_Item *item=0, int docallback=1);
00391 int extend_selection_dir(Fl_Tree_Item *from,
00392                         Fl_Tree_Item *to,
00393                         int dir,
00394                         int val,
00395                         bool visible);
00396 int extend_selection(Fl_Tree_Item *from,
00397                     Fl_Tree_Item *to,
00398                     int val=1,
00399                     bool visible=false);
00400 void set_item_focus(Fl_Tree_Item *item);
00401 Fl_Tree_Item *get_item_focus() const;
00402 int is_selected(Fl_Tree_Item *item) const;
00403 int is_selected(const char *path);
00404
00406 // Item attribute related methods
00408 Fl_Font item_labelfont() const;
00409 void item_labelfont(Fl_Font val);
00410 Fl_Fonsize item_labelsize() const;
00411 void item_labelsize(Fl_Fonsize val);
00412 Fl_Color item_labelfgcolor(void) const;
00413 void item_labelfgcolor(Fl_Color val);
00414 Fl_Color item_labelbgcolor(void) const;
00415 void item_labelbgcolor(Fl_Color val);
00416 Fl_Color connectorcolor() const;
00417 void connectorcolor(Fl_Color val);
00418 int marginleft() const;
00419 void marginleft(int val);
00420 int margintop() const;
00421 void margintop(int val);
00422 int marginbottom() const;
00423 void marginbottom(int val);
00424 int linespacing() const;
00425 void linespacing(int val);
00426 int openchild_marginbottom() const;
00427 void openchild_marginbottom(int val);
00428 int usericonmarginleft() const;
00429 void usericonmarginleft(int val);
00430 int labelmarginleft() const;
00431 void labelmarginleft(int val);
00432 int widgetmarginleft() const;
00433 void widgetmarginleft(int val);
00434 int connectorwidth() const;
00435 void connectorwidth(int val);
00436 Fl_Image* usericon() const;
00437 void usericon(Fl_Image *val);
00438 Fl_Image* openicon() const;
00439 void openicon(Fl_Image *val);
00440 Fl_Image* closeicon() const;
00441 void closeicon(Fl_Image *val);
00442 int showcollapse() const;
00443 void showcollapse(int val);
00444 int showroot() const;
00445 void showroot(int val);
00446 Fl_Tree_Connector connectorstyle() const;
00447 void connectorstyle(Fl_Tree_Connector val);
00448 Fl_Tree_Sort sortorder() const;
00449 void sortorder(Fl_Tree_Sort val);
00450 Fl_Boxtype selectbox() const;
00451 void selectbox(Fl_Boxtype val);
00452 Fl_Tree_Select selectmode() const;
00453 void selectmode(Fl_Tree_Select val);
00454 Fl_Tree_Item_Reselect_Mode item_reselect_mode() const;
00455 void item_reselect_mode(Fl_Tree_Item_Reselect_Mode mode);
00456 Fl_Tree_Item_Draw_Mode item_draw_mode() const;

```

```

00457 void item_draw_mode(Fl_Tree_Item_Draw_Mode mode);
00458 void item_draw_mode(int mode);
00459 void calc_dimensions();
00460 void calc_tree();
00461 void recalc_tree();
00462 int displayed(Fl_Tree_Item *item);
00463 void show_item(Fl_Tree_Item *item, int yoff);
00464 void show_item(Fl_Tree_Item *item);
00465 void show_item_top(Fl_Tree_Item *item);
00466 void show_item_middle(Fl_Tree_Item *item);
00467 void show_item_bottom(Fl_Tree_Item *item);
00468 void display(Fl_Tree_Item *item);
00469 int vposition() const;
00470 void vposition(int pos);
00471 int hposition() const;
00472 void hposition(int pos);
00473
00474 int is_scrollbar(Fl_Widget *w);
00475 int scrollbar_size() const;
00476 void scrollbar_size(int size);
00477 int is_vscroll_visible() const;
00478 int is_hscroll_visible() const;
00479
00481 // callback related
00483 void callback_item(Fl_Tree_Item* item);
00484 Fl_Tree_Item* callback_item();
00485 void callback_reason(Fl_Tree_Reason reason);
00486 Fl_Tree_Reason callback_reason() const;
00487
00489 void load(class Fl_Preferences&);
00490 };
00491
00492 #endif /*FL_TREE_H*/

```

12.162 Fl_Tree_Item.H File Reference

This file contains the definitions for [Fl_Tree_Item](#).

```

#include <FL/Fl.H>
#include <FL/Fl_Widget.H>
#include <FL/Fl_Image.H>
#include <FL/fl_draw.H>
#include <FL/Fl_Tree_Item_Array.H>
#include <FL/Fl_Tree_Prefs.H>

```

Classes

- class [Fl_Tree_Item](#)
Tree widget item.

12.162.1 Detailed Description

This file contains the definitions for [Fl_Tree_Item](#).

12.163 Fl_Tree_Item.H

[Go to the documentation of this file.](#)

```

00001 //
00002
00003 #ifndef FL_TREE_ITEM_H
00004 #define FL_TREE_ITEM_H
00005
00006 #include <FL/Fl.H>
00007 #include <FL/Fl_Widget.H>
00008 #include <FL/Fl_Image.H>
00009 #include <FL/fl_draw.H>
00010
00011 #include <FL/Fl_Tree_Item_Array.H>
00012 #include <FL/Fl_Tree_Prefs.H>
00013
00015 // FL/Fl_Tree_Item.H
00017 //
00018 // Fl_Tree -- This file is part of the Fl_Tree widget for FLTK

```

```

00019 // Copyright (C) 2009-2010 by Greg Ercolano.
00020 //
00021 // This library is free software. Distribution and use rights are outlined in
00022 // the file "COPYING" which should have been included with this file. If this
00023 // file is missing or damaged, see the license at:
00024 //
00025 //     https://www.fltk.org/COPYING.php
00026 //
00027 // Please see the following page on how to report bugs and issues:
00028 //
00029 //     https://www.fltk.org/bugs.php
00030 //
00031
00036
00064 class Fl_Tree;
00065 class FL_EXPORT Fl_Tree_Item {
00066     Fl_Tree      *_tree;           // parent tree
00067     const char   *_label;          // label (memory managed)
00068     Fl_Font      _labelfont;       // label's font face
00069     Fl_Fontsize  _labelsize;       // label's font size
00070     Fl_Color     _labelfgcolor;    // label's fg color
00071     Fl_Color     _labelbgcolor;    // label's bg color (0xffffffff is 'transparent')
00073     enum Fl_Tree_Item_Flags {
00074         OPEN          = 1<<0,
00075         VISIBLE       = 1<<1,
00076         ACTIVE        = 1<<2,
00077         SELECTED      = 1<<3
00078     };
00079     unsigned short _flags;         // misc flags
00080     int            _xywh[4];       // xywh of this widget (if visible)
00081     int            _collapse_xywh[4]; // xywh of collapse icon (if visible)
00082     int            _label_xywh[4]; // xywh of label
00083     Fl_Widget      *_widget;       // item's label widget (optional)
00084     Fl_Image       *_usericon;     // item's user-specific icon (optional)
00085     Fl_Image       *_userdeicon;   // deactivated usericon
00086     Fl_Tree_Item_Array _children;  // array of child items
00087     Fl_Tree_Item   *_parent;       // parent item (=0 if root)
00088     void           *_userdata;     // user data that can be associated with an item
00089     Fl_Tree_Item   *_prev_sibling; // previous sibling (same level)
00090     Fl_Tree_Item   *_next_sibling; // next sibling (same level)
00091     // Protected methods
00092 protected:
00093     void _Init(const Fl_Tree_Prefs &prefs, Fl_Tree *tree);
00094     void show_widgets();
00095     void hide_widgets();
00096     void draw_vertical_connector(int x, int y1, int y2, const Fl_Tree_Prefs &prefs);
00097     void draw_horizontal_connector(int x1, int x2, int y, const Fl_Tree_Prefs &prefs);
00098     void recalc_tree();
00099     int calc_item_height(const Fl_Tree_Prefs &prefs) const;
00100     Fl_Color drawfgcolor() const;
00101     Fl_Color drawbgcolor() const;
00102
00103 public:
00104     Fl_Tree_Item(const Fl_Tree_Prefs &prefs); // CTOR -- backwards compatible
00105     Fl_Tree_Item(Fl_Tree *tree);             // CTOR -- ABI 1.3.3+
00106     virtual ~Fl_Tree_Item();                 // DTOR -- ABI 1.3.3+
00107     Fl_Tree_Item(const Fl_Tree_Item *o);      // COPY CTOR
00109     int x() const { return(_xywh[0]); }
00111     int y() const { return(_xywh[1]); }
00114     int w() const { return(_xywh[2]); }
00116     int h() const { return(_xywh[3]); }
00119     int label_x() const { return(_label_xywh[0]); }
00122     int label_y() const { return(_label_xywh[1]); }
00126     int label_w() const { return(_label_xywh[2]); }
00129     int label_h() const { return(_label_xywh[3]); }
00130     virtual int draw_item_content(int render);
00131     void draw(int X, int &Y, int W, Fl_Tree_Item *itemfocus,
00132              int &tree_item_xmax, int lastchild=1, int render=1);
00133     void show_self(const char *indent = "") const;
00134     void label(const char *val);
00135     const char *label() const;
00136
00138     inline void user_data( void* data ) { _userdata = data; }
00139
00141     inline void* user_data() const { return _userdata; }
00142
00144     void labelfont(Fl_Font val) {
00145         _labelfont = val;
00146         recalc_tree(); // may change tree geometry
00147     }
00149     Fl_Font labelfont() const {
00150         return(_labelfont);
00151     }
00153     void labelsize(Fl_Fontsize val) {
00154         _labelsize = val;
00155         recalc_tree(); // may change tree geometry
00156     }

```

```

00158 Fl_Fonsize labelsize() const {
00159     return(_labelsize);
00160 }
00162 void labelfgcolor(Fl_Color val) {
00163     _labelfgcolor = val;
00164 }
00166 Fl_Color labelfgcolor() const {
00167     return(_labelfgcolor);
00168 }
00170 void labelcolor(Fl_Color val) {
00171     labelfgcolor(val);
00172 }
00174 Fl_Color labelcolor() const {
00175     return labelfgcolor();
00176 }
00179 void labelbgcolor(Fl_Color val) {
00180     _labelbgcolor = val;
00181 }
00186 Fl_Color labelbgcolor() const {
00187     return(_labelbgcolor);
00188 }
00190 void widget(Fl_Widget *val) {
00191     _widget = val;
00192     recalc_tree();           // may change tree geometry
00193 }
00195 Fl_Widget *widget() const {
00196     return(_widget);
00197 }
00199 int children() const {
00200     return(_children.total());
00201 }
00203 Fl_Tree_Item *child(int index) {
00204     return(_children[index]);
00205 }
00207 const Fl_Tree_Item *child(int t) const;
00209 int has_children() const {
00210     return(children());
00211 }
00212 int find_child(const char *name);
00213 int find_child(Fl_Tree_Item *item);
00214 int remove_child(Fl_Tree_Item *item);
00215 int remove_child(const char *new_label);
00216 void clear_children();
00217 void swap_children(int ax, int bx);
00218 int swap_children(Fl_Tree_Item *a, Fl_Tree_Item *b);
00219 const Fl_Tree_Item *find_child_item(const char *name) const;
00220 Fl_Tree_Item *find_child_item(const char *name);
00221 const Fl_Tree_Item *find_child_item(char **arr) const;
00222 Fl_Tree_Item *find_child_item(char **arr);
00223 const Fl_Tree_Item *find_item(char **arr) const;
00224 Fl_Tree_Item *find_item(char **arr);
00226 // Adding items
00228 Fl_Tree_Item *add(const Fl_Tree_Prefs &prefs,
00229                  const char *new_label,
00230                  Fl_Tree_Item *newitem);
00231 Fl_Tree_Item *add(const Fl_Tree_Prefs &prefs,
00232                  const char *new_label);
00233 Fl_Tree_Item *add(const Fl_Tree_Prefs &prefs,
00234                  char **arr,
00235                  Fl_Tree_Item *newitem);
00236 Fl_Tree_Item *add(const Fl_Tree_Prefs &prefs,
00237                  char **arr);
00238 Fl_Tree_Item *replace(Fl_Tree_Item *new_item);
00239 Fl_Tree_Item *replace_child(Fl_Tree_Item *olditem, Fl_Tree_Item *newitem);
00240 Fl_Tree_Item *insert(const Fl_Tree_Prefs &prefs, const char *new_label, int pos=0);
00241 Fl_Tree_Item *insert_above(const Fl_Tree_Prefs &prefs, const char *new_label);
00242 Fl_Tree_Item* deparent(int index);
00243 int reparent(Fl_Tree_Item *newchild, int index);
00244 int move(int to, int from);
00245 int move(Fl_Tree_Item *item, int op=0, int pos=0);
00246 int move_above(Fl_Tree_Item *item);
00247 int move_below(Fl_Tree_Item *item);
00248 int move_into(Fl_Tree_Item *item, int pos=0);
00249 int depth() const;
00250 Fl_Tree_Item *prev();
00251 Fl_Tree_Item *next();
00252 Fl_Tree_Item *next_sibling();
00253 Fl_Tree_Item *prev_sibling();
00254 void update_prev_next(int index);
00255 Fl_Tree_Item *next_displayed(Fl_Tree_Prefs &prefs); // deprecated
00256 Fl_Tree_Item *prev_displayed(Fl_Tree_Prefs &prefs); // deprecated
00257 Fl_Tree_Item *next_visible(Fl_Tree_Prefs &prefs);
00258 Fl_Tree_Item *prev_visible(Fl_Tree_Prefs &prefs);
00259
00261 Fl_Tree_Item *parent() {
00262     return(_parent);
00263 }

```

```

00265     const Fl_Tree_Item *parent() const {
00266         return(_parent);
00267     }
00271     void parent(Fl_Tree_Item *val) {
00272         _parent = val;
00273     }
00274     const Fl_Tree_Prefs& prefs() const;
00277     const Fl_Tree *tree() const {
00278         return(_tree);
00279     }
00282     Fl_Tree *tree() {
00283         return(_tree);
00284     }
00286     // State
00288     void open();
00289     void close();
00291     int is_open() const {
00292         return(is_flag(OPEN));
00293     }
00295     int is_close() const {
00296         return(is_flag(OPEN)?0:1);
00297     }
00299     void open_toggle() {
00300         is_open()?close():open();    // handles calling recalc_tree()
00301     }
00305     void select(int val=1) {
00306         set_flag(SELECTED, val);
00307     }
00309     void select_toggle() {
00310         if ( is_selected() ) {
00311             deselect();    // deselect if selected
00312         } else {
00313             select();    // select if deselected
00314         }
00315     }
00320     int select_all() {
00321         int count = 0;
00322         if ( ! is_selected() ) {
00323             select();
00324             ++count;
00325         }
00326         for ( int t=0; t<children(); t++ ) {
00327             count += child(t)->select_all();
00328         }
00329         return(count);
00330     }
00332     void deselect() {
00333         set_flag(SELECTED, 0);
00334     }
00339     int deselect_all() {
00340         int count = 0;
00341         if ( is_selected() ) {
00342             deselect();
00343             ++count;
00344         }
00345         for ( int t=0; t<children(); t++ ) {
00346             count += child(t)->deselect_all();
00347         }
00348         return(count);
00349     }
00351     char is_selected() const {
00352         return(is_flag(SELECTED));
00353     }
00362     void activate(int val=1) {
00363         set_flag(ACTIVE, val);
00364         if ( !_widget && val != (int)_widget->active() ) {
00365             if ( val ) {
00366                 _widget->activate();
00367             } else {
00368                 _widget->deactivate();
00369             }
00370             _widget->redraw();
00371         }
00372     }
00376     void deactivate() {
00377         activate(0);
00378     }
00380     char is_activated() const {
00381         return(is_flag(ACTIVE));
00382     }
00384     char is_active() const {
00385         return(is_activated());
00386     }
00388     int visible() const {
00389         return(is_visible());
00390     }
00392     int is_visible() const {

```

```

00393     return(is_flag(VISIBLE));
00394 }
00401 int visible_r() const {
00402     return(is_visible_r());
00403 }
00404 int is_visible_r() const;
00405
00415 void usericon(Fl_Image *val) {
00416     _usericon = val;
00417     recalc_tree();           // may change tree geometry
00418 }
00420 Fl_Image *usericon() const {
00421     return(_usericon);
00422 }
00449 void userdeicon(Fl_Image* val) {
00450     _userdeicon = val;
00451 }
00454 Fl_Image* userdeicon() const {
00455     return _userdeicon;
00456 }
00458 // Events
00460 const Fl_Tree_Item* find_clicked(const Fl_Tree_Prefs &prefs, int yonly=0) const;
00461 Fl_Tree_Item* find_clicked(const Fl_Tree_Prefs &prefs, int yonly=0);
00462 int event_on_item(const Fl_Tree_Prefs &prefs) const;
00463 int event_on_collapse_icon(const Fl_Tree_Prefs &prefs) const;
00464 int event_on_user_icon(const Fl_Tree_Prefs &prefs) const;
00465 int event_on_label(const Fl_Tree_Prefs &prefs) const;
00467 int is_root() const {
00468     return(_parent==0?1:0);
00469 }
00470
00471 // Protected methods
00472 // TODO: move these to top 'protected:' section
00473 protected:
00475 inline void set_flag(unsigned short flag,int val) {
00476     if ( flag==OPEN || flag==VISIBLE ) {
00477         recalc_tree();           // may change tree geometry
00478     }
00479     if ( val ) _flags |= flag; else _flags &= ~flag;
00480 }
00482 inline int is_flag(unsigned short val) const {
00483     return(_flags & val ? 1 : 0);
00484 }
00485
00486 };
00487
00488 #endif /*FL_TREE_ITEM_H*/

```

12.164 Fl_Tree_Item_Array.H File Reference

This file defines a class that manages an array of [Fl_Tree_Item](#) pointers.

```

#include <FL/Fl.H>
#include "Fl_Export.H"

```

Classes

- class [Fl_Tree_Item_Array](#)
Manages an array of [Fl_Tree_Item](#) pointers.

12.164.1 Detailed Description

This file defines a class that manages an array of [Fl_Tree_Item](#) pointers.

12.165 Fl_Tree_Item_Array.H

[Go to the documentation of this file.](#)

```

00001 //
00002
00003 #ifndef _FL_TREE_ITEM_ARRAY_H
00004 #define _FL_TREE_ITEM_ARRAY_H
00005
00006 #include <FL/Fl.H>
00007 #include "Fl_Export.H"
00008

```

```

00009 class FL_EXPORT Fl_Tree_Item;    // forward decl must *precede* first doxygen comment block
00010                                // or doxygen will not document our class..
00011
00013 // FL/Fl_Tree_Item_Array.H
00015 //
00016 // Fl_Tree -- This file is part of the Fl_Tree widget for FLTK
00017 // Copyright (C) 2009-2010 by Greg Ercolano.
00018 //
00019 // This library is free software. Distribution and use rights are outlined in
00020 // the file "COPYING" which should have been included with this file.  If this
00021 // file is missing or damaged, see the license at:
00022 //
00023 //     https://www.fltk.org/COPYING.php
00024 //
00025 // Please see the following page on how to report bugs and issues:
00026 //
00027 //     https://www.fltk.org/bugs.php
00028 //
00029
00034
00044
00045 class FL_EXPORT Fl_Tree_Item_Array {
00046     Fl_Tree_Item **_items;        // items array
00047     int _total;                   // #items in array
00048     int _size;                    // #items *allocated* for array
00049     int _chunksize;               // #items to enlarge mem allocation
00050     enum {
00051         MANAGE_ITEM = 1
00052     };
00053     char _flags;                  // flags to control behavior
00054     void enlarge(int count);
00055 public:
00056     Fl_Tree_Item_Array(int new_chunksize = 10);        // CTOR
00057     ~Fl_Tree_Item_Array();                             // DTOR
00058     Fl_Tree_Item_Array(const Fl_Tree_Item_Array *o);   // COPY CTOR
00059     Fl_Tree_Item *operator[](int i) {
00060         return(_items[i]);
00061     }
00062     const Fl_Tree_Item *operator[](int i) const {
00063         return(_items[i]);
00064     }
00065     int total() const {
00066         return(_total);
00067     }
00068     void swap(int ax, int bx);
00069     int move(int to, int from);
00070     int deparent(int pos);
00071     int reparent(Fl_Tree_Item *item, Fl_Tree_Item *newparent, int pos);
00072     void clear();
00073     void add(Fl_Tree_Item *val);
00074     void insert(int pos, Fl_Tree_Item *new_item);
00075     void replace(int pos, Fl_Tree_Item *new_item);
00076     void remove(int index);
00077     int remove(Fl_Tree_Item *item);
00078     void manage_item_destroy(int val) {
00079         if ( val ) _flags |= MANAGE_ITEM; else _flags &= ~MANAGE_ITEM;
00080     }
00081     int manage_item_destroy() const {
00082         return _flags & MANAGE_ITEM ? 1 : 0;
00083     }
00084 };
00085
00092 #endif /*_FL_TREE_ITEM_ARRAY_H*/

```

12.166 Fl_Tree_Prefs.H File Reference

This file contains the definitions for [Fl_Tree](#)'s preferences.

```
#include <FL/Fl.H>
```

Classes

- class [Fl_Tree_Prefs](#)
Tree widget's preferences.

Typedefs

- typedef void [Fl_Tree_Item_Draw_Callback](#)([Fl_Tree_Item](#) *, void *)

Enumerations

- enum `Fl_Tree_Connector` { `FL_TREE_CONNECTOR_NONE` =0 , `FL_TREE_CONNECTOR_DOTTED` =1 , `FL_TREE_CONNECTOR_SOLID` =2 }
Defines the style of connection lines between items.
- enum `Fl_Tree_Item_Draw_Mode` { `FL_TREE_ITEM_DRAW_DEFAULT` =0 , `FL_TREE_ITEM_DRAW_LABEL_AND_WIDGET` =1 , `FL_TREE_ITEM_HEIGHT_FROM_WIDGET` =2 }
Bit flags that control how item's labels and widget(s) are drawn in the tree via `item_draw_mode()`.
- enum `Fl_Tree_Item_Reselect_Mode` { `FL_TREE_SELECTABLE_ONCE` =0 , `FL_TREE_SELECTABLE_ALWAYS` }
Defines the ways an item can be (re) selected via `item_reselect_mode()`.
- enum `Fl_Tree_Select` { `FL_TREE_SELECT_NONE` =0 , `FL_TREE_SELECT_SINGLE` =1 , `FL_TREE_SELECT_MULTI` =2 , `FL_TREE_SELECT_SINGLE_DRAGGABLE` =3 }
Tree selection style.
- enum `Fl_Tree_Sort` { `FL_TREE_SORT_NONE` =0 , `FL_TREE_SORT_ASCENDING` =1 , `FL_TREE_SORT_DESCENDING` =2 }
Sort order options for items added to the tree.

12.166.1 Detailed Description

This file contains the definitions for `Fl_Tree`'s preferences.

```
Fl_Tree_Prefs
:
:.....:
:      :
Fl_Tree :
|_____ Fl_Tree_Item
```

12.166.2 Enumeration Type Documentation

12.166.2.1 Fl_Tree_Connector

enum `Fl_Tree_Connector`

Defines the style of connection lines between items.

Enumerator

<code>FL_TREE_CONNECTOR_NONE</code>	Use no lines connecting items.
<code>FL_TREE_CONNECTOR_DOTTED</code>	Use dotted lines connecting items (default)
<code>FL_TREE_CONNECTOR_SOLID</code>	Use solid lines connecting items.

12.166.2.2 Fl_Tree_Item_Draw_Mode

enum `Fl_Tree_Item_Draw_Mode`

Bit flags that control how item's labels and widget(s) are drawn in the tree via `item_draw_mode()`.

Enumerator

<code>FL_TREE_ITEM_DRAW_DEFAULT</code>	If <code>widget()</code> defined, draw in place of label, and <code>widget()</code> tracks item height (default)
<code>FL_TREE_ITEM_DRAW_LABEL_AND_WIDGET</code>	If <code>widget()</code> defined, include label to the left of the widget.
<code>FL_TREE_ITEM_HEIGHT_FROM_WIDGET</code>	If <code>widget()</code> defined, <code>widget()</code> 's height controls item's height.

12.166.2.3 Fl_Tree_Item_Reselect_Mode

enum `Fl_Tree_Item_Reselect_Mode`

Defines the ways an item can be (re) selected via `item_reselect_mode()`.

Enumerator

FL_TREE_SELECTABLE_ONCE	Item can only be selected once (default)
FL_TREE_SELECTABLE_ALWAYS	Enables FL_TREE_REASON_RESELECTED events for callbacks.

12.166.2.4 Fl_Tree_Select

enum [Fl_Tree_Select](#)

Tree selection style.

Enumerator

FL_TREE_SELECT_NONE	Nothing selected when items are clicked.
FL_TREE_SELECT_SINGLE	Single item selected when item is clicked (default)
FL_TREE_SELECT_MULTIPLE	Multiple items can be selected by clicking with SHIFT, CTRL or mouse drags.
FL_TREE_SELECT_SINGLE_DRAGGABLE	Single items may be selected, and they may be reordered by mouse drag.

12.166.2.5 Fl_Tree_Sort

enum [Fl_Tree_Sort](#)

Sort order options for items added to the tree.

Enumerator

FL_TREE_SORT_NONE	No sorting; items are added in the order defined (default).
FL_TREE_SORT_ASCENDING	Add items in ascending sort order.
FL_TREE_SORT_DESCENDING	Add items in descending sort order.

12.167 Fl_Tree_Prefs.H

[Go to the documentation of this file.](#)

```

00001 //
00002
00003 #ifndef FL_TREE_PREFS_H
00004 #define FL_TREE_PREFS_H
00005
00006 #include <FL/Fl.H>          // needed for ABI version features (via Enumerations.H)
00007
00008 // FL/Fl_Tree_Prefs.H
00009 //
00010 //
00011 // Fl_Tree_Prefs -- This file is part of the Fl_Tree widget for FLTK
00012 // Copyright (C) 2009-2010 by Greg Ercolano.
00013 //
00014 // This library is free software. Distribution and use rights are outlined in
00015 // the file "COPYING" which should have been included with this file.  If this
00016 // file is missing or damaged, see the license at:
00017 //
00018 //      https://www.fltk.org/COPYING.php
00019 //
00020 // Please see the following page on how to report bugs and issues:
00021 //
00022 //      https://www.fltk.org/bugs.php
00023 //
00024 //
00025
00026
00027
00028
00029
00030
00031
00032
00033
00034
00035
00036
00037
00038
00039
00040
00041
00042
00043
00044
00045
00046
00047 enum Fl_Tree_Sort {
00048     FL_TREE_SORT_NONE=0,
00049     FL_TREE_SORT_ASCENDING=1,
00050     FL_TREE_SORT_DESCENDING=2
00051 };

```

```

00052
00056 enum Fl_Tree_Connector {
00057     FL_TREE_CONNECTOR_NONE=0,
00058     FL_TREE_CONNECTOR_DOTTED=1,
00059     FL_TREE_CONNECTOR_SOLID=2
00060 };
00061
00065 enum Fl_Tree_Select {
00066     FL_TREE_SELECT_NONE=0,
00067     FL_TREE_SELECT_SINGLE=1,
00068     FL_TREE_SELECT_MULTI=2,
00070     FL_TREE_SELECT_SINGLE_DRAGGABLE=3
00072 };
00073
00078 enum Fl_Tree_Item_Reselect_Mode {
00079     FL_TREE_SELECTABLE_ONCE=0,
00080     FL_TREE_SELECTABLE_ALWAYS
00081 };
00082
00087 enum Fl_Tree_Item_Draw_Mode {
00088     FL_TREE_ITEM_DRAW_DEFAULT=0,
00090     FL_TREE_ITEM_DRAW_LABEL_AND_WIDGET=1,
00091     FL_TREE_ITEM_HEIGHT_FROM_WIDGET=2
00092 };
00093
00094 class Fl_Tree_Item;
00095 typedef void (Fl_Tree_Item_Draw_Callback) (Fl_Tree_Item*, void*);
00096
00105 class FL_EXPORT Fl_Tree_Prefs {
00106     Fl_Font _labelfont;           // label's font face
00107     Fl_Fonsize _labelsize;        // label's font size
00108     int _margin_top;              // --
00109     int _margin_left;             // -- tree's controllable margins
00110     int _margin_bottom;          // --
00111     int _openchild_margin_bottom; // extra space below an open child tree
00112     int _usericonmargin_left;     // space to left of user icon (if any)
00113     int _labelmargin_left;        // space to left of label
00114     int _widgetmargin_left;       // space to left of widget
00115     int _connectorwidth;         // connector width (right of open/close icon)
00116     int _linespacing;            // vertical space between lines
00117     // Colors
00118     Fl_Color _labelfgcolor;       // label's foreground color
00119     Fl_Color _labelbgcolor;       // label's background color
00120     Fl_Color _connectorcolor;     // connector dotted line color
00121     Fl_Tree_Connector _connectorstyle; // connector line style
00122     Fl_Image *_openimage;         // the 'open' icon [+]
00123     Fl_Image *_closeimage;        // the 'close' icon [-]
00124     Fl_Image *_userimage;         // user's own icon
00125     Fl_Image *_opendeimage;       // deactivated 'open' icon
00126     Fl_Image *_closedeimage;      // deactivated 'close' icon
00127     Fl_Image *_userdeimage;       // deactivated user icon
00128     char _showcollapse;          // 1=show collapse icons, 0=don't
00129     char _showroot;              // show the root item as part of the tree
00130     Fl_Tree_Sort _sortorder;      // none, ascending, descending, etc.
00131     Fl_Boxtype _selectbox;        // selection box type
00132     Fl_Tree_Select _selectmode;   // selection mode
00133     Fl_Tree_Item_Reselect_Mode _itemreselectmode; // controls item selection callback() behavior
00134     Fl_Tree_Item_Draw_Mode _itemdrawmode; // controls how items draw label + widget()
00135     Fl_Tree_Item_Draw_Callback *_itemdrawcallback; // callback to handle drawing items (0=none)
00136     void *_itemdrawuserdata;     // data for drawing items (0=none)
00137 public:
00138     Fl_Tree_Prefs();
00139     ~Fl_Tree_Prefs();
00140
00142     // Labels
00145     inline Fl_Font item_labelfont() const { return(_labelfont); }
00147     inline void item_labelfont(Fl_Font val) { _labelfont = val; }
00149     inline Fl_Fonsize item_labelsize() const { return(_labelsize); }
00151     inline void item_labelsize(Fl_Fonsize val) { _labelsize = val; }
00153     inline Fl_Color item_labelfgcolor() const { return(_labelfgcolor); }
00155     inline void item_labelfgcolor(Fl_Color val) { _labelfgcolor = val; }
00160     inline Fl_Color item_labelbgcolor() const {
00161         return _labelbgcolor;
00162     }
00166     inline void item_labelbgcolor(Fl_Color val) {
00167         _labelbgcolor = val;
00168     }
00169
00171     // Obsolete names - for 1.3.0 backwards compat
00174     inline Fl_Font labelfont() const { return(_labelfont); }
00176     inline void labelfont(Fl_Font val) { _labelfont = val; }
00178     inline Fl_Fonsize labelsize() const { return(_labelsize); }
00180     inline void labelsize(Fl_Fonsize val) { _labelsize = val; }
00182     inline Fl_Color labelfgcolor() const { return(_labelfgcolor); }
00184     inline void labelfgcolor(Fl_Color val) { _labelfgcolor = val; }
00186     inline Fl_Color labelbgcolor() const { return(item_labelbgcolor()); }
00188     inline void labelbgcolor(Fl_Color val) { item_labelbgcolor(val); }

```

```
00189
00191 // Margins
00194 inline int marginleft() const {
00195     return(_marginleft);
00196 }
00198 inline void marginleft(int val) {
00199     _marginleft = val;
00200 }
00202 inline int margintop() const {
00203     return(_margintop);
00204 }
00206 inline void margintop(int val) {
00207     _margintop = val;
00208 }
00211 inline int marginbottom() const {
00212     return(_marginbottom);
00213 }
00216 inline void marginbottom(int val) {
00217     _marginbottom = val;
00218 }
00220 inline int openchild_marginbottom() const {
00221     return(_openchild_marginbottom);
00222 }
00224 inline void openchild_marginbottom(int val) {
00225     _openchild_marginbottom = val;
00226 }
00228 inline int usericonmarginleft() const {
00229     return(_usericonmarginleft);
00230 }
00232 inline void usericonmarginleft(int val) {
00233     _usericonmarginleft = val;
00234 }
00236 inline int labelmarginleft() const {
00237     return(_labelmarginleft);
00238 }
00240 inline void labelmarginleft(int val) {
00241     _labelmarginleft = val;
00242 }
00244 inline int widgetmarginleft() const {
00245     return(_widgetmarginleft);
00246 }
00248 inline void widgetmarginleft(int val) {
00249     _widgetmarginleft = val;
00250 }
00252 inline int linespacing() const {
00253     return(_linespacing);
00254 }
00256 inline void linespacing(int val) {
00257     _linespacing = val;
00258 }
00259
00261 // Colors and Styles
00264 inline Fl_Color connectorcolor() const {
00265     return(_connectorcolor);
00266 }
00268 inline void connectorcolor(Fl_Color val) {
00269     _connectorcolor = val;
00270 }
00272 inline Fl_Tree_Connector connectorstyle() const {
00273     return(_connectorstyle);
00274 }
00276 inline void connectorstyle(Fl_Tree_Connector val) {
00277     _connectorstyle = val;
00278 }
00280 inline void connectorstyle(int val) {
00281     _connectorstyle = Fl_Tree_Connector(val);
00282 }
00284 inline int connectorwidth() const {
00285     return(_connectorwidth);
00286 }
00288 inline void connectorwidth(int val) {
00289     _connectorwidth = val;
00290 }
00291
00293 // Icons
00298 inline Fl_Image *openicon() const {
00299     return(_openimage);
00300 }
00301 inline int openicon_w() const { return _openimage ? _openimage->w() : 11; }
00302 inline int openicon_h() const { return _openimage ? _openimage->h() : 11; }
00303 void openicon(Fl_Image *val);
00307 inline Fl_Image *closeicon() const {
00308     return(_closeimage);
00309 }
00310 inline int closeicon_w() const { return _closeimage ? _closeimage->w() : 11; }
00311 inline int closeicon_h() const { return _closeimage ? _closeimage->h() : 11; }
00312 void closeicon(Fl_Image *val);
```

```

00314 inline Fl_Image *usericon() const {
00315     return(_userimage);
00316 }
00320 inline void usericon(Fl_Image *val) {
00321     _userimage = val;
00322     // Update deactivated version of icon..
00323     if ( _userdeimage ) delete _userdeimage;
00324     if ( _userimage ) {
00325         _userdeimage = _userimage->copy();
00326         _userdeimage->inactive();
00327     } else {
00328         _userdeimage = 0;
00329     }
00330 }
00331
00334 inline Fl_Image *opendeicon() const {
00335     return _opendeimage;
00336 }
00339 inline Fl_Image *closedeicon() const {
00340     return _closedeimage;
00341 }
00344 inline Fl_Image *userdeicon() const {
00345     return _userdeimage;
00346 }
00347
00349 // Options
00352 inline char showcollapse() const {
00353     return(_showcollapse);
00354 }
00363 inline void showcollapse(int val) {
00364     _showcollapse = val;
00365 }
00367 inline Fl_Tree_Sort sortorder() const {
00368     return(_sortorder);
00369 }
00374 inline void sortorder(Fl_Tree_Sort val) {
00375     _sortorder = val;
00376 }
00378 inline Fl_Boxtype selectbox() const {
00379     return(_selectbox);
00380 }
00382 inline void selectbox(Fl_Boxtype val) {
00383     _selectbox = val;
00384 }
00386 inline int showroot() const {
00387     return(int(_showroot));
00388 }
00393 inline void showroot(int val) {
00394     _showroot = char(val);
00395 }
00397 inline Fl_Tree_Select selectmode() const {
00398     return(_selectmode);
00399 }
00405 inline void selectmode(Fl_Tree_Select val) {
00406     _selectmode = val;
00407 }
00409 Fl_Tree_Item_Reselect_Mode item_reselect_mode() const {
00410     return _itemreselectmode;
00411 }
00413 void item_reselect_mode(Fl_Tree_Item_Reselect_Mode mode) {
00414     _itemreselectmode = mode;
00415 }
00417 inline Fl_Tree_Item_Draw_Mode item_draw_mode() const {
00418     return(_itemdrawmode);
00419 }
00425 inline void item_draw_mode(Fl_Tree_Item_Draw_Mode val) {
00426     _itemdrawmode = val;
00427 }
00428 void item_draw_callback(Fl_Tree_Item_Draw_Callback *cb, void *data=0) {
00429     _itemdrawcallback = cb;
00430     _itemdrawuserdata = data;
00431 }
00432 Fl_Tree_Item_Draw_Callback* item_draw_callback() const {
00433     return(_itemdrawcallback);
00434 }
00435 void* item_draw_user_data() const {
00436     return(_itemdrawuserdata);
00437 }
00438 void do_item_draw_callback(Fl_Tree_Item *o) const {
00439     _itemdrawcallback(o, _itemdrawuserdata);
00440 }
00441 };
00442
00443 #endif /*FL_TREE_PREFS_H*/

```

12.168 fl_types.h File Reference

This file contains simple "C"-style type definitions.

```
#include "fl_attr.h"
```

Typedefs

Miscellaneous

- typedef unsigned int [Fl_Shortcut](#)
16-bit Unicode character + 8-bit indicator for keyboard flags.
- typedef unsigned char **uchar**
unsigned char
- typedef unsigned long **ulong**
unsigned long

12.168.1 Detailed Description

This file contains simple "C"-style type definitions.

12.168.2 Typedef Documentation

12.168.2.1 Fl_Shortcut

```
typedef unsigned int Fl\_Shortcut
16-bit Unicode character + 8-bit indicator for keyboard flags.
```

Note

This **should** be 24-bit Unicode character + 8-bit indicator for keyboard flags. The upper 8 bits are currently unused but reserved.

Due to compatibility issues this type and all FLTK **shortcuts** can only be used with 16-bit Unicode characters (U+0000 .. U+FFFF) and not with the full range of unicode characters (U+0000 .. U+10FFFF).

This is caused by the bit flags `FL_SHIFT`, `FL_CTRL`, `FL_ALT`, and `FL_META` being all in the range 0x010000 .. 0x400000.

Todo Discuss and decide whether we can "shift" these special keyboard flags to the upper byte to enable full 21-bit Unicode characters (U+0000 .. U+10FFFF) plus the keyboard indicator bits as this was originally intended. This would be possible if we could rely on **all** programs being coded with symbolic names and not hard coded bit values.

12.169 fl_types.h

[Go to the documentation of this file.](#)

```
00001 /*
00002  * Simple "C"-style types for the Fast Light Tool Kit (FLTK).
00003  *
00004  * Copyright 1998-2020 by Bill Spitzak and others.
00005  *
00006  * This library is free software. Distribution and use rights are outlined in
00007  * the file "COPYING" which should have been included with this file.  If this
00008  * file is missing or damaged, see the license at:
00009  *
00010  *     https://www.fltk.org/COPYING.php
00011  *
00012  * Please see the following page on how to report bugs and issues:
00013  *
00014  *     https://www.fltk.org/bugs.php
00015  */
00016
00021 #ifndef FL_TYPES_H
00022 #define FL_TYPES_H
00023
00024 #include "fl_attr.h"
00025
```

```

00027  /* group: Miscellaneous */
00028
00030  typedef unsigned char  uchar;
00032  typedef unsigned long  ulong;
00033
00055  typedef unsigned int   Fl_Shortcut;
00056
00057  /* group: Miscellaneous */
00058
00059  #endif

```

12.170 fl_utf8.h File Reference

header for Unicode and UTF-8 character handling

```

#include "Fl_Export.H"
#include "fl_types.h"
#include <stdio.h>
#include <sys/stat.h>

```

Functions

- `int fl_access` (const char *f, int mode)
Cross-platform function to test a files access() with a UTF-8 encoded name or value.
- `int fl_chdir` (const char *path)
Cross-platform function to change the current working directory, given as a UTF-8 encoded string.
- `int fl_chmod` (const char *f, int mode)
Cross-platform function to set a files mode() with a UTF-8 encoded name or value.
- `int fl_close_fd` (int fd)
Cross-platform function to close a file descriptor.
- `int fl_execvp` (const char *file, char *const *argv)
- `FILE * fl_fopen` (const char *f, const char *mode)
Cross-platform function to open files with a UTF-8 encoded name.
- `char * fl_getcwd` (char *buf, int len)
Cross-platform function to get the current working directory as a UTF-8 encoded value.
- `char * fl_getenv` (const char *v)
Cross-platform function to get environment variables with a UTF-8 encoded name or value.
- `char fl_make_path` (const char *path)
Cross-platform function to recursively create a path in the file system.
- `void fl_make_path_for_file` (const char *path)
Cross-platform function to create a path for the file in the file system.
- `int fl_mkdir` (const char *f, int mode)
Cross-platform function to create a directory with a UTF-8 encoded name.
- `unsigned int fl_nonspacing` (unsigned int ucs)
Returns true if the Unicode character ucs is non-spacing.
- `int fl_open` (const char *fname, int oflags,...)
Cross-platform function to open files with a UTF-8 encoded name.
- `int fl_open_ext` (const char *fname, int binary, int oflags,...)
Cross-platform function to open files with a UTF-8 encoded name.
- `int fl_putenv` (const char *var)
Cross-platform function to write environment variables with a UTF-8 encoded name or value.
- `int fl_rename` (const char *f, const char *n)
Cross-platform function to rename a filesystem object using UTF-8 encoded names.
- `int fl_rmdir` (const char *f)
Cross-platform function to remove a directory with a UTF-8 encoded name.
- `int fl_stat` (const char *f, struct stat *b)

- Cross-platform function to stat() a file using a UTF-8 encoded name or value.*

 - int **fl_system** (const char *cmd)

Cross-platform function to run a system command with a UTF-8 encoded string.
- int **fl_tolower** (unsigned int ucs)

Returns the Unicode lower case value of ucs.
- int **fl_toupper** (unsigned int ucs)

Returns the Unicode upper case value of ucs.
- unsigned **fl_ucs_to_Utf16** (const unsigned ucs, unsigned short *dst, const unsigned dstlen)

Convert a single 32-bit Unicode codepoint into an array of 16-bit characters.
- int **fl_unlink** (const char *fname)

Cross-platform function to unlink() (that is, delete) a file using a UTF-8 encoded filename.
- char * **fl_utf2mbcs** (const char *s)

Converts UTF-8 string s to a local multi-byte character string.
- const char * **fl_utf8back** (const char *p, const char *start, const char *end)

Move p backward until it points to the start of a UTF-8 character.
- int **fl_utf8bytes** (unsigned ucs)

Return the number of bytes needed to encode the given UCS4 character in UTF-8.
- unsigned **fl_utf8decode** (const char *p, const char *end, int *len)

Decode a single UTF-8 encoded character starting at p.
- int **fl_utf8encode** (unsigned ucs, char *buf)

Write the UTF-8 encoding of ucs into buf and return the number of bytes written.
- unsigned **fl_utf8from_mb** (char *dst, unsigned dstlen, const char *src, unsigned srclen)

Convert a filename from the locale-specific multibyte encoding used by Windows to UTF-8 as used by FLTK.
- unsigned **fl_utf8froma** (char *dst, unsigned dstlen, const char *src, unsigned srclen)

Convert an ISO-8859-1 (ie normal c-string) byte stream to UTF-8.
- unsigned **fl_utf8fromwc** (char *dst, unsigned dstlen, const wchar_t *src, unsigned srclen)

Turn "wide characters" as returned by some system calls (especially on Windows) into UTF-8.
- const char * **fl_utf8fwd** (const char *p, const char *start, const char *end)

Move p forward until it points to the start of a UTF-8 character.
- int **fl_utf8len** (char c)

Returns the byte length of the UTF-8 sequence with first byte c, or -1 if c is not valid.
- int **fl_utf8len1** (char c)

Returns the byte length of the UTF-8 sequence with first byte c, or 1 if c is not valid.
- int **fl_utf8locale** ()

Return true if the "locale" seems to indicate that UTF-8 encoding is used.
- int **fl_utf8strlen** (const char *text, int len)

Return the length in bytes of a UTF-8 string.
- int **fl_utf8test** (const char *src, unsigned srclen)

Examines the first srclen bytes in src and returns a verdict on whether it is UTF-8 or not.
- unsigned **fl_utf8to_mb** (const char *src, unsigned srclen, char *dst, unsigned dstlen)

Convert the UTF-8 used by FLTK to the locale-specific encoding used for filenames (and sometimes used for data in files).
- unsigned **fl_utf8toa** (const char *src, unsigned srclen, char *dst, unsigned dstlen)

Convert a UTF-8 sequence into an array of 1-byte characters.
- unsigned **fl_utf8toUtf16** (const char *src, unsigned srclen, unsigned short *dst, unsigned dstlen)

Convert a UTF-8 sequence into an array of 16-bit characters.
- unsigned **fl_utf8towc** (const char *src, unsigned srclen, wchar_t *dst, unsigned dstlen)

Converts a UTF-8 string into a wide character string.
- int **fl_utf_nb_char** (const unsigned char *buf, int len)

Returns the number of Unicode chars in the UTF-8 string.
- int **fl_utf_strcasecmp** (const char *s1, const char *s2)

- `UTF-8 aware strcasecmp - converts to Unicode and tests.`
- `int fl_utf_strncasecmp (const char *s1, const char *s2, int n)`
UTF-8 aware strncasecmp - converts to lower case Unicode and tests.
- `int fl_utf_tolower (const unsigned char *str, int len, char *buf)`
Converts the string str to its lower case equivalent into buf.
- `int fl_utf_toupper (const unsigned char *str, int len, char *buf)`
Converts the string str to its upper case equivalent into buf.
- `int fl_wcwidth (const char *src)`
extended wrapper around fl_wcwidth_(unsigned int ucs) function.
- `int fl_wcwidth_ (unsigned int ucs)`
Wrapper to adapt Markus Kuhn's implementation of wcwidth() for FLTK.

12.170.1 Detailed Description

header for Unicode and UTF-8 character handling

12.171 fl_utf8.h

[Go to the documentation of this file.](#)

```

00001 /*
00002  * Author: Jean-Marc Lienher ( http://oksid.ch )
00003  * Copyright 2000-2010 by O'ksi'D.
00004  * Copyright 2016-2021 by Bill Spitzak and others.
00005  *
00006  * This library is free software. Distribution and use rights are outlined in
00007  * the file "COPYING" which should have been included with this file. If this
00008  * file is missing or damaged, see the license at:
00009  *
00010  * https://www.fltk.org/COPYING.php
00011  *
00012  * Please see the following page on how to report bugs and issues:
00013  *
00014  * https://www.fltk.org/bugs.php
00015  */
00016
00017 /* Merged in some functionality from the fltk-2 version. IMM.
00018  * The following code is an attempt to merge the functions incorporated in FLTK2
00019  * with the functions provided in Oksid's fltk-1.1.6-utf8 port
00020  */
00021
00027 #ifndef _HAVE_FL_UTF8_HDR_
00028 #define _HAVE_FL_UTF8_HDR_
00029
00030 #include "Fl_Export.H"
00031 #include "fl_types.h"
00032 #include <stdio.h> // FILE *fl_fopen()
00033 #include <sys/stat.h> // struct stat
00034
00035 #ifdef __cplusplus
00036 extern "C" {
00037 #endif
00038
00043 /* F2: comes from FLTK2 */
00044 /* OD: comes from Oksid */
00045
00051 FL_EXPORT int fl_utf8bytes(unsigned ucs);
00052
00053 /* OD: returns the byte length of the first UTF-8 char sequence (returns -1 if not valid) */
00054 FL_EXPORT int fl_utf8len(char c);
00055
00056 /* OD: returns the byte length of the first UTF-8 char sequence (returns +1 if not valid) */
00057 FL_EXPORT int fl_utf8len1(char c);
00058
00059 /* OD: returns the byte length of a UTF-8 text */
00060 FL_EXPORT int fl_utf8strlen(const char *text, int len);
00061
00062 /* OD: returns the number of Unicode chars in the UTF-8 string */
00063 FL_EXPORT int fl_utf_nb_char(const unsigned char *buf, int len);
00064
00065 /* F2: Convert the next UTF-8 char-sequence into a Unicode value (and say how many bytes were used) */
00066 FL_EXPORT unsigned fl_utf8decode(const char* p, const char* end, int* len);
00067
00068 /* F2: Encode a Unicode value into a UTF-8 sequence, return the number of bytes used */
00069 FL_EXPORT int fl_utf8encode(unsigned ucs, char* buf);
00070

```



```

00071 /* F2: Move forward to the next valid UTF-8 sequence start between start and end */
00072 FL_EXPORT const char* fl_utf8fwd(const char* p, const char* start, const char* end);
00073
00074 /* F2: Move backward to the previous valid UTF-8 sequence start */
00075 FL_EXPORT const char* fl_utf8back(const char* p, const char* start, const char* end);
00076
00077 /* XX: Convert a single 32-bit Unicode value into UTF16 */
00078 FL_EXPORT unsigned fl_ucs_to_Utf16(const unsigned ucs, unsigned short *dst, const unsigned dstlen);
00079
00080 /* F2: Convert a UTF-8 string into UTF16 */
00081 FL_EXPORT unsigned fl_utf8toUtf16(const char* src, unsigned srclen, unsigned short* dst, unsigned
dstlen);
00082
00083 /* F2: Convert a UTF-8 string into a wide character string - makes UTF16 on win32, "UCS4" elsewhere */
00084 FL_EXPORT unsigned fl_utf8towa(const char *src, unsigned srclen, wchar_t *dst, unsigned dstlen);
00085
00086 /* F2: Convert a wide character string to UTF-8 - takes in UTF16 on win32, "UCS4" elsewhere */
00087 FL_EXPORT unsigned fl_utf8fromwc(char *dst, unsigned dstlen, const wchar_t *src, unsigned srclen);
00088
00089 /* F2: Convert a UTF-8 string into ASCII, eliding untranslatable glyphs */
00090 FL_EXPORT unsigned fl_utf8toa (const char *src, unsigned srclen, char *dst, unsigned dstlen);
00091
00092 /* F2: Convert 8859-1 string to UTF-8 */
00093 FL_EXPORT unsigned fl_utf8froma (char *dst, unsigned dstlen, const char *src, unsigned srclen);
00094
00095 /* F2: Returns true if the current O/S locale is UTF-8 */
00096 FL_EXPORT int fl_utf8locale(void);
00097
00098 /* F2: Examine the first len characters of src, to determine if the input text is UTF-8 or not
00099  * NOTE: The value returned is not simply boolean - it contains information about the probable
00100  * type of the src text. */
00101 FL_EXPORT int fl_utf8test(const char *src, unsigned len);
00102
00103 /* XX: return width of "raw" ucs character in columns.
00104  * for internal use only */
00105 FL_EXPORT int fl_wcwidth_(unsigned int ucs);
00106
00107 /* XX: return width of UTF-8 character string in columns.
00108  * NOTE: this may also do C1 control character (0x80 to 0x9f) to CP1252 mapping,
00109  * depending on original build options */
00110 FL_EXPORT int fl_wcwidth(const char *src);
00111
00112 /* OD: Return true if the character is non-spacing */
00113 FL_EXPORT unsigned int fl_nonspacing(unsigned int ucs);
00114
00115 /* F2: Convert UTF-8 to a local multi-byte encoding - mainly for win32? */
00116 FL_EXPORT unsigned fl_utf8to_mb(const char *src, unsigned srclen, char *dst, unsigned dstlen);
00117 /* OD: Convert UTF-8 to a local multi-byte encoding */
00118 FL_EXPORT char* fl_utf2mbcs(const char *src);
00119
00120 /* F2: Convert a local multi-byte encoding to UTF-8 - mainly for win32? */
00121 FL_EXPORT unsigned fl_utf8from_mb(char *dst, unsigned dstlen, const char *src, unsigned srclen);
00122
00123 /*****
00124 #ifdef _WIN32
00125  * these two Windows-only functions are kept for API compatibility */
00126  * OD: Attempt to convert the UTF-8 string to the current locale */
00127 FL_EXPORT char *fl_utf8_to_locale(const char *s, int len, unsigned int codepage);
00128
00129  * OD: Attempt to convert a string in the current locale to UTF-8 */
00130 FL_EXPORT char *fl_locale_to_utf8(const char *s, int len, unsigned int codepage);
00131 #endif /* _WIN32 */
00132
00133 /*****
00134  * The following functions are intended to provide portable, UTF-8 aware
00135  * versions of standard functions
00136  */
00137
00138  * OD: UTF-8 aware strncasecmp - converts to lower case Unicode and tests */
00139 FL_EXPORT int fl_utf_strncasecmp(const char *s1, const char *s2, int n);
00140
00141  * OD: UTF-8 aware strcasecmp - converts to Unicode and tests */
00142 FL_EXPORT int fl_utf_strcasecmp(const char *s1, const char *s2);
00143
00144  * OD: return the Unicode lower case value of ucs */
00145 FL_EXPORT int fl_tolower(unsigned int ucs);
00146
00147  * OD: return the Unicode upper case value of ucs */
00148 FL_EXPORT int fl_toupper(unsigned int ucs);
00149
00150  * OD: converts the UTF-8 string to the lower case equivalent */
00151 FL_EXPORT int fl_utf_tolower(const unsigned char *str, int len, char *buf);
00152
00153  * OD: converts the UTF-8 string to the upper case equivalent */
00154 FL_EXPORT int fl_utf_toupper(const unsigned char *str, int len, char *buf);
00155
00156  * OD: Portable UTF-8 aware chmod wrapper */

```

```

00157 FL_EXPORT int fl_chmod(const char* f, int mode);
00158
00159 /* OD: Portable UTF-8 aware access wrapper */
00160 FL_EXPORT int fl_access(const char* f, int mode);
00161
00162 /* OD: Portable UTF-8 aware stat wrapper */
00163 FL_EXPORT int fl_stat(const char *path, struct stat *buffer);
00164
00165 /* OD: Portable UTF-8 aware getcwd wrapper */
00166 FL_EXPORT char *fl_getcwd(char *buf, int len);
00167
00168 /* Portable UTF-8 aware chdir wrapper */
00169 FL_EXPORT int fl_chdir(const char *path);
00170
00171 /* OD: Portable UTF-8 aware fopen wrapper */
00172 FL_EXPORT FILE *fl_fopen(const char *f, const char *mode);
00173
00174 /* OD: Portable UTF-8 aware system wrapper */
00175 FL_EXPORT int fl_system(const char* f);
00176
00177 /* OD: Portable UTF-8 aware execvp wrapper */
00178 FL_EXPORT int fl_execvp(const char *file, char *const *argv);
00179
00180 /* OD: Portable UTF-8 aware open wrapper */
00181 FL_EXPORT int fl_open(const char *fname, int oflags, ...);
00182
00183 FL_EXPORT int fl_open_ext(const char *fname, int binary, int oflags, ...);
00184
00185 /* Portable wrapper around unix-style close() function */
00186 FL_EXPORT int fl_close_fd(int fd);
00187
00188 /* OD: Portable UTF-8 aware unlink wrapper */
00189 FL_EXPORT int fl_unlink(const char *fname);
00190
00191 /* OD: Portable UTF-8 aware rmdir wrapper */
00192 FL_EXPORT int fl_rmdir(const char *f);
00193
00194 /* OD: Portable UTF-8 aware getenv wrapper */
00195 FL_EXPORT char* fl_getenv(const char *name);
00196
00197 /* Portable UTF-8 aware putenv wrapper */
00198 FL_EXPORT int fl_putenv(const char *var);
00199
00200 /* OD: Portable UTF-8 aware mkdir wrapper */
00201 FL_EXPORT int fl_mkdir(const char* f, int mode);
00202
00203 /* OD: Portable UTF-8 aware rename wrapper */
00204 FL_EXPORT int fl_rename(const char* f, const char *t);
00205
00206
00207 /* OD: Given a full pathname, this will create the directory path needed to hold the file named */
00208 FL_EXPORT void fl_make_path_for_file( const char *path );
00209
00210 /* OD: recursively create a path in the file system */
00211 FL_EXPORT char fl_make_path( const char *path );
00212
00213
00216 /*****
00217
00218 #ifdef __cplusplus
00219 }
00220 #endif /* __cplusplus */
00221
00222
00223 #endif /* _HAVE_FL_UTF8_HDR_ */

```

12.172 Fl_Valuator.H

```

00001 //
00002 // Valuator header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file

```

```

00018     Fl_Valuator widget . */
00019
00020 #ifndef Fl_Valuator_H
00021 #define Fl_Valuator_H
00022
00023 #ifndef Fl_Widget_H
00024 #include "Fl_Widget.H"
00025 #endif
00026
00027 // shared type() values for classes that work in both directions:
00028 #define FL_VERTICAL      0
00029 #define FL_HORIZONTAL    1
00030
00047 class FL_EXPORT Fl_Valuator : public Fl_Widget {
00048
00049     double value_;
00050     double previous_value_;
00051     double min, max; // truncates to this range *after* rounding
00052     double A; int B; // rounds to multiples of A/B, or no rounding if A is zero
00053
00054 protected:
00055     int horizontal() const {return type() & FL_HORIZONTAL;}
00056     Fl_Valuator(int X, int Y, int W, int H, const char* L);
00057
00058     double previous_value() const {return previous_value_;}
00062     void handle_push() {previous_value_ = value_;}
00063     double softclamp(double);
00064     void handle_drag(double newvalue);
00065     void handle_release(); // use drag() value
00066     virtual void value_damage(); // cause damage() due to value() changing
00068     void set_value(double v) {value_ = v;}
00069
00070 public:
00071
00073     ~Fl_Valuator() FL_OVERRIDE { }
00075     void bounds(double a, double b) {min=a; max=b;}
00077     double minimum() const {return min;}
00079     void minimum(double a) {min = a;}
00081     double maximum() const {return max;}
00083     void maximum(double a) {max = a;}
00104     void range(double a, double b) {min = a; max = b;}
00106     void step(int a) {A = a; B = 1;}
00108     void step(double a, int b) {A = a; B = b;}
00109     void step(double s);
00121     double step() const {return A/B;}
00122     void precision(int digits);
00123
00125     double value() const {return value_;}
00126     int value(double);
00127
00128     virtual int format(char*);
00129     double round(double); // round to nearest multiple of step
00130     double clamp(double); // keep in range
00131     double increment(double, int); // add n*step to value
00132 };
00133
00134 #endif

```

12.173 Fl_Value_Input.H

```

00001 //
00002 // Value input header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Value_Input widget . */
00019
00020 #ifndef Fl_Value_Input_H
00021 #define Fl_Value_Input_H
00022
00023 #include "Fl_Valuator.H"
00024 #include "Fl_Input.H"
00025

```

```

00056 class FL_EXPORT Fl_Value_Input : public Fl_Valuator {
00057 public:
00058     /* This is the encapsulated Fl_input attribute to which
00059     this class delegates the value font, color and shortcut */
00060     Fl_Input input;
00061 private:
00062     char soft_;
00063     static void input_cb(Fl_Widget*,void*);
00064     void value_damage() FL_OVERRIDE; // cause damage() due to value() changing
00065 public:
00066     int handle(int) FL_OVERRIDE;
00067 protected:
00068     void draw() FL_OVERRIDE;
00069 public:
00070     void resize(int,int,int,int) FL_OVERRIDE;
00071     Fl_Value_Input(int x,int y,int w,int h,const char *l=0);
00072     ~Fl_Value_Input();
00073
00075     void soft(char s) {soft_ = s;}
00082     char soft() const {return soft_;}
00087     int shortcut() const {return input.shortcut();}
00105     void shortcut(int s) {input.shortcut(s);}
00106
00108     Fl_Font textfont() const {return input.textfont();}
00110     void textfont(Fl_Font s) {input.textfont(s);}
00112     Fl_Fonsize textsize() const {return input.textsize();}
00114     void textsize(Fl_Fonsize s) {input.textsize(s);}
00116     Fl_Color textcolor() const {return input.textcolor();}
00118     void textcolor(Fl_Color n) {input.textcolor(n);}
00120     Fl_Color cursor_color() const {return input.cursor_color();}
00122     void cursor_color(Fl_Color n) {input.cursor_color(n);}
00123
00124 };
00125
00126 #endif

```

12.174 Fl_Value_Output.H

```

00001 //
00002 // Value output header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_Value_Output widget . */
00019
00020 #ifndef Fl_Value_Output_H
00021 #define Fl_Value_Output_H
00022
00023 #ifndef Fl_Valuator_H
00024 #include "Fl_Valuator.H"
00025 #endif
00026
00039 class FL_EXPORT Fl_Value_Output : public Fl_Valuator {
00040     Fl_Font textfont_;
00041     Fl_Fonsize textsize_;
00042     uchar soft_;
00043     Fl_Color textcolor_;
00044
00045 protected:
00046     void draw() FL_OVERRIDE;
00047
00048 public:
00049     int handle(int) FL_OVERRIDE;
00050     Fl_Value_Output(int x,int y,int w,int h,const char *l=0);
00051
00058     void soft(uchar s) {soft_ = s;}
00065     uchar soft() const {return soft_;}
00066
00068     Fl_Font textfont() const {return textfont_;}
00070     void textfont(Fl_Font s) {textfont_ = s;}
00072     Fl_Fonsize textsize() const {return textsize_;}
00073     void textsize(Fl_Fonsize s) {textsize_ = s;}
00075     Fl_Color textcolor() const {return textcolor_;}

```

```

00077 void textcolor(Fl_Color s) {textcolor_ = s;}
00078 };
00079
00080 #endif

```

12.175 Fl_Value_Slider.H

```

00001 //
00002 // Value Slider header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018    Fl_Value_Slider widget . */
00019
00020 #ifndef Fl_Value_Slider_H
00021 #define Fl_Value_Slider_H
00022
00023 #include "Fl_Slider.H"
00024
00031 class FL_EXPORT Fl_Value_Slider : public Fl_Slider {
00032     Fl_Font textfont_;
00033     Fl_Fontsize textsize_;
00034     Fl_Color textcolor_;
00035     short value_width_;
00036     short value_height_;
00037
00038 protected:
00039     void draw() FL_OVERRIDE;
00040
00041 public:
00042     int handle(int) FL_OVERRIDE;
00043     Fl_Value_Slider(int x, int y, int w, int h, const char *l = 0);
00044
00046     Fl_Font textfont() const { return textfont_; }
00047
00049     void textfont(Fl_Font s) { textfont_ = s; }
00050
00052     Fl_Fontsize textsize() const { return textsize_; }
00053
00055     void textsize(Fl_Fontsize s) { textsize_ = s; }
00056
00058     Fl_Color textcolor() const { return textcolor_; }
00059
00061     void textcolor(Fl_Color s) { textcolor_ = s; }
00062
00078     void value_width(int s) {
00079         if (s > w() - 10)
00080             s = w() - 10;
00081         if (s < 10)
00082             s = 10;
00083         value_width_ = (short)s;
00084     }
00085
00090     int value_width() const { return (value_width_); }
00091
00107     void value_height(int s) {
00108         if (s > h() - 10)
00109             s = h() - 10;
00110         if (s < 10)
00111             s = 10;
00112         value_height_ = (short)s;
00113     }
00118     int value_height() const { return (value_height_); }
00119 };
00120
00121 #endif

```

12.176 Fl_Widget.H File Reference

[Fl_Widget](#) and [Fl_Label](#) classes.

```
#include "Fl.H"
```

Classes

- class [Fl_Callback_User_Data](#)
A class prototype that allows for additional data in callbacks.
- struct [Fl_Label](#)
This struct stores all information for a text or mixed graphics label.
- class [Fl_Widget](#)
[Fl_Widget](#) is the base class for all widgets in FLTK.

Macros

- #define [FL_RESERVED_TYPE](#) 100
Reserved type numbers (necessary for my cheapo RTTI) start here.

Typedefs

- typedef void [Fl_Callback](#)([Fl_Widget](#) *, void *)
Default callback type definition for all fltk widgets (by far the most used)
- typedef void [Fl_Callback0](#)([Fl_Widget](#) *)
One parameter callback type definition passing only the widget.
- typedef void [Fl_Callback1](#)([Fl_Widget](#) *, long)
Callback type definition passing the widget and a long data value.
- typedef [Fl_Callback](#) * [Fl_Callback_p](#)
Default callback type pointer definition for all fltk widgets.

12.176.1 Detailed Description

[Fl_Widget](#) and [Fl_Label](#) classes.

12.176.2 Macro Definition Documentation

12.176.2.1 FL_RESERVED_TYPE

```
#define FL_RESERVED_TYPE 100
```

Reserved type numbers (necessary for my cheapo RTTI) start here.

Grep the header files for "RESERVED_TYPE" to find the next available number.

12.177 Fl_Widget.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Widget header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00022 #ifndef Fl_Widget_H
00023 #define Fl_Widget_H
00024
00025 #include "Fl.H"
```

```

00026
00027 class Fl_Widget;
00028 class Fl_Window;
00029 class Fl_Group;
00030 class Fl_Image;
00031
00033 typedef void (Fl_Callback ) (Fl_Widget*, void*);
00035 typedef Fl_Callback* Fl_Callback_p; // needed for BORLAND
00037 typedef void (Fl_Callback0) (Fl_Widget*);
00039 typedef void (Fl_Callback1) (Fl_Widget*, long);
00040
00049 struct FL_EXPORT Fl_Label {
00051     const char* value;
00053     Fl_Image* image;
00055     Fl_Image* deimage;
00057     Fl_Font font;
00059     Fl_Fontsize size;
00061     Fl_Color color;
00063     Fl_Align align_;
00065     uchar type;
00067     signed char h_margin_;
00069     signed char v_margin_;
00071     uchar spacing;
00072
00074     void draw(int,int,int,int, Fl_Align) const ;
00075     void measure(int &w, int &h) const ;
00076 };
00077
00078
00092 class FL_EXPORT Fl_Callback_User_Data {
00093 protected:
00094     Fl_Callback_User_Data() {}
00095 public:
00096     virtual ~Fl_Callback_User_Data() { }
00097 };
00098
00099
00112 class FL_EXPORT Fl_Widget {
00113     friend class Fl_Group;
00114
00115     Fl_Group* parent_;
00116     Fl_Callback* callback_;
00117     void* user_data_;
00118     int x_, y_, w_, h_;
00119     Fl_Label label_;
00120     unsigned int flags_;
00121     Fl_Color color_;
00122     Fl_Color color2_;
00123     uchar type_;
00124     uchar damage_;
00125     uchar box_;
00126     uchar when_;
00127
00128     const char *tooltip_;
00129
00131     Fl_Widget(const Fl_Widget &);
00133     Fl_Widget& operator=(const Fl_Widget &);
00134
00135 protected:
00136
00147     Fl_Widget(int x, int y, int w, int h, const char *label=0L);
00148
00150     void x(int v) {x_ = v;}
00152     void y(int v) {y_ = v;}
00154     void w(int v) {w_ = v;}
00156     void h(int v) {h_ = v;}
00158     unsigned int flags() const {return flags_;}
00160     void set_flag(unsigned int c) {flags_ |= c;}
00162     void clear_flag(unsigned int c) {flags_ &= ~c;}
00166     enum {
00167         INACTIVE          = 1<<0,
00168         INVISIBLE         = 1<<1,
00169         OUTPUT            = 1<<2,
00170         NOBORDER          = 1<<3,
00171         FORCE_POSITION     = 1<<4,
00172         NON_MODAL         = 1<<5,
00173         SHORTCUT_LABEL    = 1<<6,
00174         CHANGED           = 1<<7,
00175         OVERRIDE          = 1<<8,
00176         VISIBLE_FOCUS     = 1<<9,
00177         COPIED_LABEL      = 1<<10,
00178         CLIP_CHILDREN     = 1<<11,
00179         MENU_WINDOW       = 1<<12,
00180         TOOLTIP_WINDOW    = 1<<13,
00181         MODAL             = 1<<14,
00182         NO_OVERLAY        = 1<<15,
00183         GROUP_RELATIVE    = 1<<16,

```

```

00184         COPIED_TOOLTIP = 1<<17,
00185         FULLSCREEN      = 1<<18,
00186         MAC_USE_ACCENTS_MENU = 1<<19,
00187         NEEDS_KEYBOARD  = 1<<20,
00188         IMAGE_BOUND     = 1<<21,
00189         DEIMAGE_BOUND   = 1<<22,
00190         AUTO_DELETE_USER_DATA = 1<<23,
00191         MAXIMIZED       = 1<<24,
00192         POPUP           = 1<<25,
00193         // Note to devs: add new FLTK core flags above this line (up to 1<<28).
00194
00195         // Three more flags, reserved for user code
00196
00197         USERFLAG3       = 1<<29,
00198         USERFLAG2       = 1<<30,
00199         USERFLAG1       = 1<<31
00200     };
00201     void draw_box() const;
00202     void draw_box(Fl_Boxtype t, Fl_Color c) const;
00203     void draw_box(Fl_Boxtype t, int x,int y,int w,int h, Fl_Color c) const;
00204     void draw_backdrop() const;
00205
00211     void draw_focus() const {
00212         draw_focus(box(), x(), y(), w(), h(), color());
00213     }
00214
00220     void draw_focus(Fl_Boxtype t, int X, int Y, int W, int H) const {
00221         draw_focus(t, X, Y, W, H, color());
00222     }
00223
00224     // See documentation in Fl_Widget.cxx
00225     void draw_focus(Fl_Boxtype t, int x, int y, int w, int h, Fl_Color bg) const;
00226
00227     void draw_label() const;
00228     void draw_label(int, int, int, int) const;
00229
00230 public:
00231
00240     virtual ~Fl_Widget();
00241
00258     virtual void draw() = 0;
00259
00289     virtual int handle(int event);
00290
00299     int is_label_copied() const {return ((flags_ & COPIED_LABEL) ? 1 : 0);}
00300
00315     void needs_keyboard(bool needs) {
00316         if (needs) set_flag(NEEDS_KEYBOARD);
00317         else clear_flag(NEEDS_KEYBOARD);
00318     }
00319
00325     bool needs_keyboard() const {
00326         return (flags_ & NEEDS_KEYBOARD);
00327     }
00328
00334     Fl_Group* parent() const {return parent_;}
00335
00344     void parent(Fl_Group* p) {parent_ = p;} // for hacks only, use Fl_Group::add()
00345
00351     uchar type() const {return type_;}
00352
00356     void type(uchar t) {type_ = t;}
00357
00361     int x() const {return x_;}
00362
00366     int y() const {return y_;}
00367
00371     int w() const {return w_;}
00372
00376     int h() const {return h_;}
00377
00397     virtual void resize(int x, int y, int w, int h);
00398
00400     int damage_resize(int,int,int,int);
00401
00409     void position(int X,int Y) {resize(X,Y,w_,h_);}
00410
00418     void size(int W,int H) {resize(x_,y_,W,H);}
00419
00425     Fl_Align align() const {return label_.align_;}
00426
00434     void align(Fl_Align alignment) {label_.align_ = alignment;}
00435
00440     Fl_Boxtype box() const {return (Fl_Boxtype)box_;}
00441
00449     void box(Fl_Boxtype new_box) {box_ = new_box;}
00450

```



```

00455     Fl_Color color() const {return color_;}
00456
00467     void color(Fl_Color bg) {color_ = bg;}
00468
00473     Fl_Color selection_color() const {return color2_;}
00474
00483     void selection_color(Fl_Color a) {color2_ = a;}
00484
00492     void color(Fl_Color bg, Fl_Color sel) {color_=bg; color2_=sel;}
00493
00498     const char* label() const {return label_.value;}
00499
00511     void label(const char* text);
00512
00523     void copy_label(const char *new_label);
00524
00528     void label(Fl_Labeltype a, const char* b) {label_.type = a; label_.value = b;}
00529
00534     Fl_Labeltype labeltype() const {return (Fl_Labeltype)label_.type;}
00535
00544     void labeltype(Fl_Labeltype a) {label_.type = a;}
00545
00550     Fl_Color labelcolor() const {return label_.color;}
00551
00556     void labelcolor(Fl_Color c) {label_.color=c;}
00557
00565     Fl_Font labelfont() const {return label_.font;}
00566
00574     void labelfont(Fl_Font f) {label_.font=f;}
00575
00580     Fl_Fontsize labelsize() const {return label_.size;}
00581
00586     void labelsize(Fl_Fontsize pix) {label_.size=pix;}
00587
00591     Fl_Image* image() {return label_.image;}
00592
00596     const Fl_Image* image() const {return label_.image;}
00597
00613     void image(Fl_Image* img);
00614
00619     void image(Fl_Image& img);
00620
00636     void bind_image(Fl_Image* img);
00637
00642     void bind_image(int f) { if (f) set_flag(IMAGE_BOUND); else clear_flag(IMAGE_BOUND); }
00643
00649     int image_bound() const {return ((flags_ & IMAGE_BOUND) ? 1 : 0);}
00650
00654     Fl_Image* deimage() {return label_.deimage;}
00655
00659     const Fl_Image* deimage() const {return label_.deimage;}
00660
00667     void deimage(Fl_Image* img);
00668
00673     void deimage(Fl_Image& img);
00674
00681     void bind_deimage(Fl_Image* img);
00682
00688     int deimage_bound() const {return ((flags_ & DEIMAGE_BOUND) ? 1 : 0);}
00689
00694     void bind_deimage(int f) { if (f) set_flag(DEIMAGE_BOUND); else clear_flag(DEIMAGE_BOUND); }
00695
00700     void label_image_spacing(int gap) { label_.spacing = (uchar)gap; }
00701
00705     int label_image_spacing() { return label_.spacing; }
00706
00710     void horizontal_label_margin(int px) { label_.h_margin_ = (signed char)px; }
00711
00715     int horizontal_label_margin() { return label_.h_margin_; }
00716
00720     void vertical_label_margin(int px) { label_.v_margin_ = (signed char)px; }
00721
00725     int vertical_label_margin() { return label_.v_margin_; }
00726
00731     const char *tooltip() const {return tooltip_;}
00732
00733     void tooltip(const char *text); // see Fl_Tooltip
00734     void copy_tooltip(const char *text); // see Fl_Tooltip
00735
00740     Fl_Callback_p callback() const {return callback_;}
00741
00747     void callback(Fl_Callback* cb, void* p) {
00748         callback_ = cb;
00749         user_data(p);
00750     }
00751
00759     void callback(Fl_Callback* cb, Fl_Callback_User_Data* p, bool auto_free) {

```

```

00760     callback_ = cb;
00761     user_data(p, auto_free);
00762 }
00763
00768 void callback(Fl_Callback* cb) {callback_ = cb;}
00769
00774 void callback(Fl_Callback0* cb) {
00775     callback_ = (Fl_Callback*)(fl_intptr_t)(cb);
00776 }
00777
00783 void callback(Fl_Callback1* cb, long p = 0) {
00784     callback_ = (Fl_Callback*)(fl_intptr_t)(cb);
00785     user_data((void*)(fl_intptr_t)p);
00786 }
00787
00792 void* user_data() const {return user_data_;}
00793
00795 void user_data(void* v);
00796
00798 void user_data(Fl_Callback_User_Data* v, bool auto_free);
00799
00811 long argument() const {return (long)(fl_intptr_t)user_data_;}
00812
00817 void argument(long v) {user_data((void*)(fl_intptr_t)v);}
00818
00827 Fl_When when() const {return (Fl_When)when_;}
00828
00866 void when(uchar i) {when_ = i;}
00867
00872 unsigned int visible() const {return !(flags_ & INVISIBLE);}
00873
00878 int visible_r() const;
00879
00897 virtual void show();
00898
00902 virtual void hide();
00903
00908 void set_visible() {flags_ &= ~INVISIBLE;}
00909
00914 void clear_visible() {flags_ |= INVISIBLE;}
00915
00920 unsigned int active() const {return !(flags_ & INACTIVE);}
00921
00926 int active_r() const;
00927
00933 void activate();
00934
00949 void deactivate();
00950
00959 unsigned int output() const {return (flags_ & OUTPUT);}
00960
00964 void set_output() {flags_ |= OUTPUT;}
00965
00969 void clear_output() {flags_ &= ~OUTPUT;}
00970
00976 unsigned int takeevents() const {return !(flags_ & (INACTIVE|INVISIBLE|OUTPUT));}
00977
00996 unsigned int changed() const {return flags_ & CHANGED;}
00997
01001 void set_changed() {flags_ |= CHANGED;}
01002
01006 void clear_changed() {flags_ &= ~CHANGED;}
01007
01012 void clear_active() {flags_ |= INACTIVE;}
01013
01018 void set_active() {flags_ &= ~INACTIVE;}
01019
01027 int take_focus();
01028
01035 void set_visible_focus() { flags_ |= VISIBLE_FOCUS; }
01036
01041 void clear_visible_focus() { flags_ &= ~VISIBLE_FOCUS; }
01042
01047 void visible_focus(int v) { if (v) set_visible_focus(); else clear_visible_focus(); }
01048
01053 unsigned int visible_focus() const { return flags_ & VISIBLE_FOCUS; }
01054
01075 static void default_callback(Fl_Widget *widget, void *data);
01076
01089 void do_callback(Fl_Callback_Reason reason=FL_REASON_UNKNOWN) {do_callback(this, user_data_,
reason);}
01090
01098 void do_callback(Fl_Widget *widget, long arg, Fl_Callback_Reason reason=FL_REASON_UNKNOWN) {
01099     do_callback(widget, (void*)(fl_intptr_t)arg, reason);
01100 }
01101
01102 void do_callback(Fl_Widget *widget, void *arg = 0, Fl_Callback_Reason reason=FL_REASON_UNKNOWN);

```

```

01103
01104  /* Internal use only. */
01105  int test_shortcut();
01106  /* Internal use only. */
01107  static unsigned int label_shortcut(const char *t);
01108  /* Internal use only. */
01109  static int test_shortcut(const char*, const bool require_alt = false);
01110  /* Internal use only. */
01111  void _set_fullscreen() {flags_ |= FULLSCREEN;}
01112  void _clear_fullscreen() {flags_ &= ~FULLSCREEN;}
01113
01119  int contains(const Fl_Widget *w) const ;
01120
01127  int inside(const Fl_Widget *wgt) const {return wgt ? wgt->contains(this) : 0;}
01128
01132  void redraw();
01133
01138  void redraw_label();
01139
01146  uchar damage() const {return damage_;}
01147
01160  void clear_damage(uchar c = 0) {damage_ = c;}
01161
01167  void damage(uchar c);
01168
01175  void damage(uchar c, int x, int y, int w, int h);
01176
01177  void draw_label(int, int, int, int, Fl_Align) const;
01178
01186  void measure_label(int& ww, int& hh) const {label_.measure(ww, hh);}
01187
01188  Fl_Window* window() const ;
01189  Fl_Window* top_window() const;
01190  Fl_Window* top_window_offset(int& xoff, int& yoff) const;
01191
01215  virtual Fl_Group* as_group() { return NULL; }
01216  virtual Fl_Group const* as_group() const { return NULL; }
01217
01230  virtual Fl_Window* as_window() { return 0; }
01231  virtual Fl_Window const* as_window() const { return NULL; }
01232
01243  virtual class Fl_Gl_Window* as_gl_window() { return NULL; }
01244  virtual class Fl_Gl_Window const* as_gl_window() const { return NULL; }
01245
01248  int use_accents_menu() { return flags() & MAC_USE_ACCENTS_MENU; }
01249
01253  Fl_Color color2() const {return (Fl_Color)color2_;}
01254
01258  void color2(unsigned a) {color2_ = a;}
01259
01271  void shortcut_label(int value) {
01272      if (value)
01273          set_flag(SHORTCUT_LABEL);
01274      else
01275          clear_flag(SHORTCUT_LABEL);
01276  }
01277
01280  int shortcut_label() const { return flags_ & SHORTCUT_LABEL; }
01281 };
01282
01288 #define FL_RESERVED_TYPE 100
01289
01290 #endif

```

12.178 Fl_Widget_Surface.H

```

00001 //
00002 // Drivers code for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2016 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Widget_Surface_h
00018 #define Fl_Widget_Surface_h
00019

```

```

00020 #include <FL/Fl_Device.H>
00021 #include <FL/Fl_Window.H>
00022
00025 class FL_EXPORT Fl_Widget_Surface : public Fl_Surface_Device {
00026 private:
00027     void traverse(Fl_Widget *widget); // finds subwindows of widget and prints them
00028 protected:
00030     int x_offset;
00032     int y_offset;
00033     Fl_Widget_Surface(Fl_Graphics_Driver *d);
00034 public:
00035     virtual void translate(int x, int y);
00036     virtual void untranslate();
00037     void draw(Fl_Widget* widget, int delta_x = 0, int delta_y = 0);
00038     void draw_decorated_window(Fl_Window *win, int x_offset = 0, int y_offset = 0);
00039     void print_window_part(Fl_Window *win, int x, int y, int w, int h, int delta_x = 0, int delta_y =
00040     0);
00041     virtual int printable_rect(int *w, int *h);
00042     virtual void origin(int x, int y);
00043     virtual void origin(int *x, int *y);
00044 };
00045 #endif /* Fl_Widget_Surface_h */

```

12.179 FL_Window.H File Reference

[FL_Window](#) widget .

```

#include <FL/Fl.H>
#include <FL/Fl_Group.H>
#include <FL/Fl_Bitmap.H>

```

Classes

- class [FL_Window](#)
This widget produces an actual window.

Macros

- #define **FL_DOUBLE_WINDOW** 0xF1
double window type id
- #define **FL_WINDOW** 0xF0
window type id: all subclasses have type() >= this

12.179.1 Detailed Description

[FL_Window](#) widget .

12.180 FL_Window.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Window header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00020 #ifndef FL_Window_H
00021 #define FL_Window_H
00022

```

```

00023 #include <FL/Fl.H>
00024 #include <FL/Fl_Group.H>
00025 #include <FL/Fl_Bitmap.H>
00026
00027 #define FL_WINDOW 0xF0
00028 #define FL_DOUBLE_WINDOW 0xF1
00029
00030 class Fl_X;
00031 class Fl_Window_Driver;
00032 class Fl_RGB_Image;
00033 class Fl_Double_Window;
00034
00055 class FL_EXPORT Fl_Window : public Fl_Group {
00056     friend class Fl_X;
00057     friend class Fl_Window_Driver;
00058 private:
00059     static char *default_xclass_;
00060     static char show_next_window_iconic_; // 1 means create next window in iconic form
00061
00062     int no_fullscreen_x;
00063     int no_fullscreen_y;
00064     int no_fullscreen_w;
00065     int no_fullscreen_h;
00066     int fullscreen_screen_top;
00067     int fullscreen_screen_bottom;
00068     int fullscreen_screen_left;
00069     int fullscreen_screen_right;
00070
00071     // TODO: it would make sense to merge the use of Fl_X and Fl_Window_Driver, maybe simply by
00072     // TODO: deriving Fl_Window_Driver from Fl_X. However, there are a lot of historic kludges
00073     // TODO: for some platforms around Fl_X.
00074     Fl_X *flx_; // points at the system-specific stuff, but exists only after the window is mapped
00075     Fl_Window_Driver *pWindowDriver; // points at the system-specific stuff at window creation time
00076
00077     const char* iconlabel_;
00078     char* xclass_;
00079
00080     // private size_range stuff:
00081     int minw_, minh_, maxw_, maxh_;
00082     int dw_, dh_, aspect_;
00083     uchar size_range_set_; // true (1) if size_range() has been set or calculated
00084
00085     // cursor stuff
00086     Fl_Cursor cursor_default;
00087
00088     void _Fl_Window(); // constructor innards
00089
00090     // unimplemented copy ctor and assignment operator
00091     Fl_Window(const Fl_Window&);
00092     Fl_Window& operator=(const Fl_Window&);
00093
00094     void is_maximized_(bool b);
00095
00096 protected:
00097
00099     static Fl_Window *current_;
00100     void draw() FL_OVERRIDE;
00101
00102 public:
00103
00105     virtual void flush();
00106
00107 protected:
00108
00117     void force_position(int force) {
00118         if (force) set_flag(FORCE_POSITION);
00119         else clear_flag(FORCE_POSITION);
00120     }
00121
00129     int force_position() const { return ((flags() & FORCE_POSITION)?1:0); }
00130
00131     void free_icons();
00132
00133     void default_size_range(); // calculate size_range() if not set explicitly
00134     int is_resizable(); // calculate size_range() and return whether this is resizable
00135
00136 public:
00137
00166     Fl_Window(int w, int h, const char *title = 0);
00189     Fl_Window(int x, int y, int w, int h, const char *title = 0);
00198     virtual ~Fl_Window();
00199
00200     int handle(int) FL_OVERRIDE;
00201
00218     void resize(int X,int Y,int W,int H) FL_OVERRIDE;
00225     void border(int b);
00230     void clear_border() {set_flag(NO BORDER);}
00232     unsigned int border() const {return !(flags() & NO BORDER);}

```

```

00234 void set_override() {set_flag(NO BORDER|OVERRIDE);}
00236 unsigned int override() const { return flags() & OVERRIDE; }
00245 void set_modal() {set_flag(MODAL);}
00247 unsigned int modal() const {return flags() & MODAL;}
00254 void set_non_modal() {set_flag(NON_MODAL);}
00256 unsigned int non_modal() const {return flags() & (NON_MODAL|MODAL);}
00257
00295 void clear_modal_states() {clear_flag(NON_MODAL | MODAL);}
00296
00310 void set_menu_window() {set_flag(MENU_WINDOW);}
00311
00313 unsigned int menu_window() const {return flags() & MENU_WINDOW;}
00314
00331 void set_tooltip_window() { set_flag(TOOLTIP_WINDOW);
00332 clear_flag(MENU_WINDOW); }
00334 unsigned int tooltip_window() const {return flags() & TOOLTIP_WINDOW;}
00335
00343 void hotspot(int x, int y, int offscreen = 0);
00345 void hotspot(const Fl_Widget*, int offscreen = 0);
00347 void hotspot(const Fl_Widget& p, int offscreen = 0) {hotspot(&p, offscreen);}
00348
00357 void free_position() {clear_flag(FORCE_POSITION);}
00358
00359 void size_range(int minw, int minh, int maxw=0, int maxh=0, int dw=0, int dh=0, int aspect=0);
00360
00361 uchar get_size_range(int *minw, int *minh, int *maxw=NULL, int *maxh=NULL, int *dw=NULL, int
*dh=NULL, int *aspect=NULL);
00362
00364 const char* label() const {return Fl_Widget::label();}
00366 const char* iconlabel() const {return iconlabel_;}
00368 void label(const char*);
00370 void iconlabel(const char*);
00372 void label(const char* label, const char* iconlabel); // platform dependent
00373 void copy_label(const char* a);
00374
00375 static void default_xclass(const char*);
00376 static const char *default_xclass();
00377 const char* xclass() const;
00378 void xclass(const char* c);
00379
00380 static void default_icon(const Fl_RGB_Image*);
00381 static void default_icons(const Fl_RGB_Image*[], int);
00382 void icon(const Fl_RGB_Image*);
00383 void icons(const Fl_RGB_Image*[], int);
00384
00385 #if defined(_WIN32) || defined(FL_DOXYGEN)
00386 typedef struct HICON_* HICON;
00387 // These 2 member functions break the driver model but are kept for back compatibility.
00388 // They are implemented in Fl_win32.cxx
00389
00409 static void default_icons(HICON big_icon, HICON small_icon);
00410
00419 void icons(HICON big_icon, HICON small_icon);
00420 #endif // defined(_WIN32) || defined(FL_DOXYGEN)
00421
00422 /* for legacy compatibility */
00423 const void* icon() const;
00424 void icon(const void * ic);
00425
00431 int shown() {return flx_ != 0;}
00458 void show() FL_OVERRIDE;
00463 void hide() FL_OVERRIDE;
00484 void show(int argc, char **argv);
00485
00486 // Enables synchronous show(), docs in Fl_Window.cxx
00487 void wait_for_expose();
00488
00500 void fullscreen();
00504 void fullscreen_off();
00509 void fullscreen_off(int X, int Y, int W, int H);
00513 unsigned int fullscreen_active() const { return flags() & FULLSCREEN; }
00524 void fullscreen_screens(int top, int bottom, int left, int right);
00525
00526 void maximize();
00527 void un_maximize();
00529 unsigned int maximize_active() const { return flags() & MAXIMIZED; }
00530 public:
00546 void iconize();
00547
00548 int x_root() const ;
00549 int y_root() const ;
00550
00551 static Fl_Window *current();
00561 void make_current();
00562
00563 void cursor(Fl_Cursor);
00564 void cursor(const Fl_RGB_Image*, int, int);

```

```

00565 void default_cursor(Fl_Cursor);
00566
00567 /* for legacy compatibility */
00568 void cursor(Fl_Cursor c, Fl_Color, Fl_Color=FL_WHITE);
00569 void default_cursor(Fl_Cursor c, Fl_Color, Fl_Color=FL_WHITE);
00570
00571 static void default_callback(Fl_Window*, void* v);
00572
00578 int decorated_w() const;
00579
00593 int decorated_h() const;
00594
00595 // Note: Doxygen docs in Fl_Widget.H to avoid redundancy.
00596 Fl_Window* as_window() FL_OVERRIDE { return this; }
00597 Fl_Window const* as_window() const FL_OVERRIDE { return this; }
00598
00602 virtual class Fl_Overlay_Window *as_overlay_window() {return 0L; }
00603
00607 virtual class Fl_Double_Window *as_double_window() {return 0L; }
00608
00609 void shape(const Fl_Image* img);
00610 void shape(const Fl_Image& b);
00611 const Fl_Image *shape();
00612 void draw_backdrop();
00613 int screen_num();
00614 void screen_num(int screen_num);
00615 static bool is_a_rescale();
00616 fl_uintptr_t os_id();
00617
00625 static void show_next_window_iconic(char stat) {
00626     show_next_window_iconic_ = stat ? 1 : 0;
00627 }
00628
00636 static char show_next_window_iconic() {
00637     return show_next_window_iconic_;
00638 }
00639
00640 void allow_expand_outside_parent();
00641
00642 };
00643
00644 #endif

```

12.181 Fl_Wizard.H

```

00001 //
00002 // Fl_Wizard widget definitions.
00003 //
00004 // Copyright 1999-2010 by Easy Software Products.
00005 // Copyright 2011-2020 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00018 /* \file
00019    Fl_Wizard widget . */
00020
00021 //
00022 // Include necessary header files...
00023 //
00024
00025 #ifndef _Fl_Wizard_H_
00026 #define _Fl_Wizard_H_
00027
00028 #include <FL/Fl_Group.H>
00029
00040 class FL_EXPORT Fl_Wizard : public Fl_Group {
00041
00042     Fl_Widget *value_;
00043
00044 protected:
00045
00046     void draw() FL_OVERRIDE;
00047
00048 public:
00049
00050     Fl_Wizard(int, int, int, int, const char * = 0);

```

```
00051
00052     void next();
00053     void prev();
00054     Fl_Widget *value();
00055     void value(Fl_Widget *);
00056 };
00057
00058 #endif // !_Fl_Wizard_H_
```

12.182 Fl_XBM_Image.H

```
00001 //
00002 // XBM image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_XBM_Image class . */
00019
00020 #ifndef Fl_XBM_Image_H
00021 #define Fl_XBM_Image_H
00022 #    include "Fl_Bitmap.H"
00023
00024 class FL_EXPORT Fl_XBM_Image : public Fl_Bitmap {
00025 public:
00026     Fl_XBM_Image(const char* filename);
00027 };
00028
00029 #endif // !Fl_XBM_Image_H
```

12.183 Fl_XPM_Image.H

```
00001 //
00002 // XPM image header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /* \file
00018     Fl_XPM_Image class . */
00019
00020 #ifndef Fl_XPM_Image_H
00021 #define Fl_XPM_Image_H
00022 #    include "Fl_Pixmap.H"
00023
00024 class FL_EXPORT Fl_XPM_Image : public Fl_Pixmap {
00025 public:
00026     Fl_XPM_Image(const char* filename);
00027 };
00028
00029 #endif // !Fl_XPM_Image
```


12.184 forms.H

```

00001 //
00002 // Forms emulation header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //      https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef __FORMS_H__
00018 #define __FORMS_H__
00019
00020 #include "Fl.H"
00021 #include "Fl_Group.H"
00022 #include "Fl_Window.H"
00023 #include "fl_draw.H"
00024
00025 typedef Fl_Widget FL_OBJECT;
00026 typedef Fl_Window FL_FORM;
00027
00028 // Random constants & symbols defined by forms.h file:
00029
00030 #ifndef NULL
00031 #define NULL 0
00032 #endif
00033 #ifndef FALSE
00034 #define FALSE 0
00035 #endif
00036 #ifndef TRUE
00037 #define TRUE 1
00038 #endif
00039 #define FL_ON 1
00040 #define FL_OK 1
00041 #define FL_VALID 1
00042 #define FL_PREEMPT 1
00043 #define FL_AUTO 2
00044 #define FL_WHEN_NEEDED FL_AUTO
00045 #define FL_OFF 0
00046 #define FL_NONE 0
00047 #define FL_CANCEL 0
00048 #define FL_INVALID 0
00049 #define FL_IGNORE -1
00050 // #define FL_CLOSE -2 // this variable is never used in FLTK Forms. It is removed
00051 // because it conflicts with the window FL_CLOSE event
00052
00053 #define FL_BLACK FL_BLACK
00054 #define FL_COL1 FL_GRAY
00055 #define FL_MCOL FL_LIGHT1
00056 #define FL_LEFT_BCOL FL_LIGHT3 // 53 is better match
00057 #define FL_TOP_BCOL FL_LIGHT2 // 51
00058 #define FL_BOTTOM_BCOL FL_DARK2 // 40
00059 #define FL_RIGHT_BCOL FL_DARK3 // 36
00060 #define FL_INACTIVE FL_INACTIVE_COLOR
00061 #define FL_INACTIVE_COL FL_INACTIVE_COLOR
00062 #define FL_FREE_COL1 FL_FREE_COLOR
00063 #define FL_FREE_COL2 ((Fl_Color) (FL_FREE_COLOR+1))
00064 #define FL_FREE_COL3 ((Fl_Color) (FL_FREE_COLOR+2))
00065 #define FL_FREE_COL4 ((Fl_Color) (FL_FREE_COLOR+3))
00066 #define FL_FREE_COL5 ((Fl_Color) (FL_FREE_COLOR+4))
00067 #define FL_FREE_COL6 ((Fl_Color) (FL_FREE_COLOR+5))
00068 #define FL_FREE_COL7 ((Fl_Color) (FL_FREE_COLOR+6))
00069 #define FL_FREE_COL8 ((Fl_Color) (FL_FREE_COLOR+7))
00070 #define FL_FREE_COL9 ((Fl_Color) (FL_FREE_COLOR+8))
00071 #define FL_FREE_COL10 ((Fl_Color) (FL_FREE_COLOR+9))
00072 #define FL_FREE_COL11 ((Fl_Color) (FL_FREE_COLOR+10))
00073 #define FL_FREE_COL12 ((Fl_Color) (FL_FREE_COLOR+11))
00074 #define FL_FREE_COL13 ((Fl_Color) (FL_FREE_COLOR+12))
00075 #define FL_FREE_COL14 ((Fl_Color) (FL_FREE_COLOR+13))
00076 #define FL_FREE_COL15 ((Fl_Color) (FL_FREE_COLOR+14))
00077 #define FL_FREE_COL16 ((Fl_Color) (FL_FREE_COLOR+15))
00078 #define FL_TOMATO ((Fl_Color) (131))
00079 #define FL_INDIANRED ((Fl_Color) (164))
00080 #define FL_SLATEBLUE ((Fl_Color) (195))
00081 #define FL_DARKGOLD ((Fl_Color) (84))
00082 #define FL_PALEGREEN ((Fl_Color) (157))
00083 #define FL_ORCHID ((Fl_Color) (203))
00084 #define FL_DARKCYAN ((Fl_Color) (189))
00085 #define FL_DARKTOMATO ((Fl_Color) (113))
00086 #define FL_WHEAT ((Fl_Color) (174))

```

```

00087
00088 #define FL_ALIGN_BESIDE FL_ALIGN_INSIDE
00089
00090 #define FL_PUP_TOGGLE 2 // FL_MENU_TOGGLE
00091 #define FL_PUP_INACTIVE 1 // FL_MENU_INACTIVE
00092 #define FL_NO_FRAME FL_NO_BOX
00093 #define FL_ROUNDED3D_UPBOX FL_ROUND_UP_BOX
00094 #define FL_ROUNDED3D_DOWNBOX FL_ROUND_DOWN_BOX
00095 #define FL_OVAL3D_UPBOX FL_ROUND_UP_BOX
00096 #define FL_OVAL3D_DOWNBOX FL_ROUND_DOWN_BOX
00097
00098 #define FL_MBUTTON1 1
00099 #define FL_LEFTMOUSE 1
00100 #define FL_MBUTTON2 2
00101 #define FL_MIDDLEMOUSE 2
00102 #define FL_MBUTTON3 3
00103 #define FL_RIGHTMOUSE 3
00104 #define FL_MBUTTON4 4
00105 #define FL_MBUTTON5 5
00106
00107 #define FL_INVALID_STYLE 255
00108 #define FL_NORMAL_STYLE FL_HELVETICA
00109 #define FL_BOLD_STYLE FL_HELVETICA_BOLD
00110 #define FL_ITALIC_STYLE FL_HELVETICA_ITALIC
00111 #define FL_BOLDITALIC_STYLE FL_HELVETICA_BOLD_ITALIC
00112 #define FL_FIXED_STYLE FL_COURIER
00113 #define FL_FIXEDBOLD_STYLE FL_COURIER_BOLD
00114 #define FL_FIXEDITALIC_STYLE FL_COURIER_ITALIC
00115 #define FL_FIXEDBOLDITALIC_STYLE FL_COURIER_BOLD_ITALIC
00116 #define FL_TIMES_STYLE FL_TIMES
00117 #define FL_TIMESBOLD_STYLE FL_TIMES_BOLD
00118 #define FL_TIMESITALIC_STYLE FL_TIMES_ITALIC
00119 #define FL_TIMESBOLDITALIC_STYLE FL_TIMES_BOLD_ITALIC
00120
00121 // hacks to change the labeltype() when passed to fl_set_object_lstyle():
00122 #define FL_SHADOW_STYLE (FL_SHADOW_LABEL<<8)
00123 #define FL_ENGRAVED_STYLE (FL_ENGRAVED_LABEL<<8)
00124 #define FL_EMBOSSED_STYLE (FL_EMBOSSED_LABEL<<0)
00125
00126 // size values are different from XForms, match older Forms:
00127 #define FL_TINY_SIZE 8
00128 #define FL_SMALL_SIZE 11 // 10
00129 // #define FL_NORMAL_SIZE 14 // 12
00130 #define FL_MEDIUM_SIZE 18 // 14
00131 #define FL_LARGE_SIZE 24 // 18
00132 #define FL_HUGE_SIZE 32 // 24
00133 #define FL_DEFAULT_SIZE FL_SMALL_SIZE
00134 #define FL_TINY_FONT FL_TINY_SIZE
00135 #define FL_SMALL_FONT FL_SMALL_SIZE
00136 #define FL_NORMAL_FONT FL_NORMAL_SIZE
00137 #define FL_MEDIUM_FONT FL_MEDIUM_SIZE
00138 #define FL_LARGE_FONT FL_LARGE_SIZE
00139 #define FL_HUGE_FONT FL_HUGE_SIZE
00140 #define FL_NORMAL_FONT1 FL_SMALL_FONT
00141 #define FL_NORMAL_FONT2 FL_NORMAL_FONT
00142 #define FL_DEFAULT_FONT FL_SMALL_FONT
00143
00144 #define FL_RETURN_END_CHANGED FL_WHEN_RELEASE
00145 #define FL_RETURN_CHANGED FL_WHEN_CHANGED
00146 #define FL_RETURN_END FL_WHEN_RELEASE_ALWAYS
00147 #define FL_RETURN_ALWAYS (FL_WHEN_CHANGED|FL_WHEN_NOT_CHANGED)
00148
00149 #define FL_BOUND_WIDTH 3
00150
00151 typedef int FL_Coord;
00152 typedef int FL_COLOR;
00153
00154 // fltk interaction:
00155
00156 #define FL_CMD_OPT void
00157 extern FL_EXPORT void fl_initialize(int*, char*[], const char*, FL_CMD_OPT*, int);
00158 inline void fl_finish() {}
00159
00160 typedef void (*FL_IO_CALLBACK) (FL_SOCKET, void*);
00161 inline void fl_add_io_callback(int fd, short w, FL_IO_CALLBACK cb, void* v) {
00162     Fl::add_fd(fd, w, cb, v);}
00163
00164 inline void fl_remove_io_callback(int fd, short, FL_IO_CALLBACK) {
00165     Fl::remove_fd(fd);} // removes all the callbacks!
00166
00167 // type of callback is different and no "id" number is returned:
00168 inline void fl_add_timeout(long msec, void (*cb)(void*), void* v) {
00169     Fl::add_timeout(msec*.001, cb, v);}
00170 inline void fl_remove_timeout(int) {}
00171
00172 // type of callback is different!
00173 inline void fl_set_idle_callback(void (*cb)()) {Fl::set_idle(cb);}
00174

```

```

00175 FL_EXPORT Fl_Widget* fl_do_forms(void);
00176 FL_EXPORT Fl_Widget* fl_check_forms();
00177 inline Fl_Widget* fl_do_only_forms(void) {return fl_do_forms();}
00178 inline Fl_Widget* fl_check_only_forms(void) {return fl_check_forms();}
00179
00180 // because of new redraw behavior, these are no-ops:
00181 inline void fl_freeze_object(Fl_Widget*) {}
00182 inline void fl_unfreeze_object(Fl_Widget*) {}
00183 inline void fl_freeze_form(Fl_Window*) {}
00184 inline void fl_unfreeze_form(Fl_Window*) {}
00185 inline void fl_freeze_all_forms() {}
00186 inline void fl_unfreeze_all_forms() {}
00187
00188 inline void fl_set_focus_object(Fl_Window*, Fl_Widget* o) {Fl::focus(o);}
00189 inline void fl_reset_focus_object(Fl_Widget* o) {Fl::focus(o);}
00190 #define fl_set_object_focus fl_set_focus_object
00191
00192 // void fl_set_form_atclose(Fl_Window*w,int (*cb)(Fl_Window*,void*),void* v)
00193 // void fl_set_atclose(int (*cb)(Fl_Window*,void*),void*)
00194 // fl_set_form_atactivate/atdeactivate not implemented!
00195
00197 // Fl_Widget:
00198
00199 inline void fl_set_object_boxtype(Fl_Widget* o, Fl_Boxtype a) {o->box(a);}
00200 inline void fl_set_object_lsize(Fl_Widget* o,int s) {o->labelsize(s);}
00201
00202 /* forms lib font indexes must be byte sized - extract correct byte from style word */
00203 inline void fl_set_object_lstyle(Fl_Widget* o,int a) {
00204     o->labelfont((Fl_Font)(a&0xf)); o->labeltype((Fl_Labeltype)(a>>8));}
00205 inline void fl_set_object_lcol(Fl_Widget* o, Fl_Color a) {o->labelcolor(a);}
00206 #define fl_set_object_lcolor fl_set_object_lcol
00207 inline void fl_set_object_lalign(Fl_Widget* o, Fl_Align a) {o->align(a);}
00208 #define fl_set_object_align fl_set_object_lalign
00209 inline void fl_set_object_color(Fl_Widget* o,Fl_Color a,Fl_Color b) {o->color(a,b);}
00210 inline void fl_set_object_label(Fl_Widget* o, const char* a) {o->label(a); o->redraw();}
00211 inline void fl_set_object_position(Fl_Widget*o,int x,int y) {o->position(x,y);}
00212 inline void fl_set_object_size(Fl_Widget* o, int w, int h) {o->size(w,h);}
00213 inline void fl_set_object_geometry(Fl_Widget* o,int x,int y,int w,int h) {o->resize(x,y,w,h);}
00214
00215 inline void fl_get_object_geometry(Fl_Widget* o,int*x,int*y,int*w,int*h) {
00216     *x = o->x(); *y = o->y(); *w = o->w(); *h = o->h();}
00217 inline void fl_get_object_position(Fl_Widget* o,int*x,int*y) {
00218     *x = o->x(); *y = o->y();}
00219
00220 typedef void (*Forms_CB)(Fl_Widget*, long);
00221 inline void fl_set_object_callback(Fl_Widget*o,Forms_CB c,long a) {o->callback(c,a);}
00222 #define fl_set_call_back fl_set_object_callback
00223 inline void fl_call_object_callback(Fl_Widget* o) {o->do_callback();}
00224 inline void fl_trigger_object(Fl_Widget* o) {o->do_callback();}
00225 inline void fl_set_object_return(Fl_Widget* o, int v) {
00226     o->when((Fl_When)(v|FL_WHEN_RELEASE));}
00227
00228 inline void fl_redraw_object(Fl_Widget* o) {o->redraw();}
00229 inline void fl_show_object(Fl_Widget* o) {o->show();}
00230 inline void fl_hide_object(Fl_Widget* o) {o->hide();}
00231 inline void fl_free_object(Fl_Widget* x) {delete x;}
00232 inline void fl_delete_object(Fl_Widget* o) {o->parent()->remove(*o);}
00233 inline void fl_activate_object(Fl_Widget* o) {o->activate();}
00234 inline void fl_deactivate_object(Fl_Widget* o) {o->deactivate();}
00235
00236 inline void fl_add_object(Fl_Window* f, Fl_Widget* x) {f->add(x);}
00237 inline void fl_insert_object(Fl_Widget* o, Fl_Widget* b) {b->parent()->insert(*o,b);}
00238
00239 inline Fl_Window* FL_ObjWin(Fl_Widget* o) {return o->window();}
00240
00242 // things that appered in the demos a lot that I don't emulate, but
00243 // I did not want to edit out of all the demos...
00244
00245 inline int fl_get_border_width() {return 3;}
00246 inline void fl_set_border_width(int) {}
00247 inline void fl_set_object_dblbuffer(Fl_Widget*, int) {}
00248 inline void fl_set_form_dblbuffer(Fl_Window*, int) {}
00249
00251 // Fl_Window:
00252
00253 inline void fl_free_form(Fl_Window* x) {delete x;}
00254 inline void fl_redraw_form(Fl_Window* f) {f->redraw();}
00255
00256 inline Fl_Window* fl_bgn_form(Fl_Boxtype b,int w,int h) {
00257     Fl_Window* g = new Fl_Window(w,h,0);
00258     g->box(b);
00259     return g;
00260 }
00261 FL_EXPORT void fl_end_form();
00262 inline void fl_addto_form(Fl_Window* f) {f->begin();}
00263 inline Fl_Group* fl_bgn_group() {return new Fl_Group(0,0,0,0);}
00264 inline void fl_end_group() {Fl_Group::current()->forms_end();}

```

```

00265 inline void fl_addto_group(Fl_Widget* o) {((Fl_Group*)o)->begin();}
00266 #define resizebox_dddfeign_kludge()
00267
00268 inline void fl_scale_form(Fl_Window* f, double x, double y) {
00269     f->resizable(f); f->size(int(f->w()*x),int(f->h()*y));}
00270 inline void fl_set_form_position(Fl_Window* f,int x,int y) {f->position(x,y);}
00271 inline void fl_set_form_size(Fl_Window* f, int w, int h) {f->size(w,h);}
00272 inline void fl_set_form_geometry(Fl_Window* f,int x,int y,int w,int h) {
00273     f->resize(x,y,w,h);}
00274 #define fl_set_initial_placement fl_set_form_geometry
00275 inline void fl_adjust_form_size(Fl_Window*) {}
00276
00277 FL_EXPORT void fl_show_form(Fl_Window* f,int p,int b,const char* n);
00278 enum { // "p" argument values:
00279     FL_PLACE_FREE = 0, // make resizable
00280     FL_PLACE_MOUSE = 1, // mouse centered on form
00281     FL_PLACE_CENTER = 2, // center of the screen
00282     FL_PLACE_POSITION = 4, // fixed position, resizable
00283     FL_PLACE_SIZE = 8, // fixed size, normal fltk behavior
00284     FL_PLACE_GEOMETRY = 16, // fixed size and position
00285     FL_PLACE_ASPECT = 32, // keep aspect ratio (ignored)
00286     FL_PLACE_FULLSCREEN = 64, // fill screen
00287     FL_PLACE_HOTSPOT = 128, // enables hotspot
00288     FL_PLACE_ICONIC = 256, // iconic (ignored)
00289     FL_FREE_SIZE = (1<14), // force resizable
00290     FL_FIX_SIZE = (1<15) // force off resizable
00291 };
00292 #define FL_PLACE_FREE_CENTER (FL_PLACE_CENTER|FL_FREE_SIZE)
00293 #define FL_PLACE_CENTERFREE (FL_PLACE_CENTER|FL_FREE_SIZE)
00294 enum { // "b" argument values:
00295     FL_NOBORDER = 0,
00296     FL_FULLBORDER,
00297     FL_TRANSIENT
00298 //FL_MODAL = (1<8) // not implemented yet in Forms
00299 };
00300 inline void fl_set_form_hotspot(Fl_Window* w,int x,int y) {w->hotspot(x,y);}
00301 inline void fl_set_form_hotobject(Fl_Window* w, Fl_Widget* o) {w->hotspot(o);}
00302 extern FL_EXPORT char fl_flip; // in forms.C
00303 inline void fl_flip_yorigin() {fl_flip = 1;}
00304
00305 #define fl_prepare_form_window fl_show_form
00306 inline void fl_show_form_window(Fl_Window*) {}
00307
00308 inline void fl_raise_form(Fl_Window* f) {f->show();}
00309
00310 inline void fl_hide_form(Fl_Window* f) {f->hide();}
00311 inline void fl_pop_form(Fl_Window* f) {f->show();}
00312
00313 extern FL_EXPORT char fl_modal_next; // in forms.C
00314 inline void fl_activate_all_forms() {}
00315 inline void fl_deactivate_all_forms() {fl_modal_next = 1;}
00316 inline void fl_deactivate_form(Fl_Window*w) {w->deactivate();}
00317 inline void fl_activate_form(Fl_Window*w) {w->activate();}
00318
00319 inline void fl_set_form_title(Fl_Window* f, const char* s) {f->label(s);}
00320 inline void fl_title_form(Fl_Window* f, const char* s) {f->label(s);}
00321
00322 typedef void (*Forms_FormCB)(Fl_Widget*);
00323 inline void fl_set_form_callback(Fl_Window* f,Forms_FormCB c) {f->callback(c);}
00324 #define fl_set_form_callback fl_set_form_callback
00325
00326 inline void fl_init() {}
00327 FL_EXPORT void fl_set_graphics_mode(int,int);
00328
00329 inline int fl_form_is_visible(Fl_Window* f) {return f->visible();}
00330
00331 inline int fl_mouse_button() {return Fl::event_button();}
00332 #define fl_mousebutton fl_mouse_button
00333
00334 #define fl_free free
00335 #define fl_malloc malloc
00336 #define fl_calloc calloc
00337 #define fl_realloc realloc
00338
00340 // Drawing functions. Only usable inside an Fl_Free object?
00341
00342 inline void fl_drw_box(Fl_Boxtype b,int x,int y,int w,int h,Fl_Color bgc,int=3) {
00343     fl_draw_box(b,x,y,w,h,bgc);}
00344 inline void fl_drw_frame(Fl_Boxtype b,int x,int y,int w,int h,Fl_Color bgc,int=3) {
00345     fl_draw_box(b,x,y,w,h,bgc);}
00346
00347 inline void fl_drw_text(Fl_Align align, int x, int y, int w, int h,
00348     Fl_Color fgcolor, int size, Fl_Font style,
00349     const char* s) {
00350     fl_font(style,size);
00351     fl_color(fgcolor);
00352     fl_draw(s,x,y,w,h,align);

```

```

00353 }
00354
00355 // this does not work except for CENTER...
00356 inline void fl_drw_text_beside(Fl_Align align, int x, int y, int w, int h,
00357                               Fl_Color fgcolor, int size, Fl_Font style,
00358                               const char* s) {
00359     fl_font(style,size);
00360     fl_color(fgcolor);
00361     fl_draw(s,x,y,w,h,align);
00362 }
00363
00364 inline void fl_set_font_name(Fl_Font n,const char* s) {Fl::set_font(n,s);}
00365
00366 inline void fl_mapcolor(Fl_Color c, uchar r, uchar g, uchar b) {Fl::set_color(c,r,g,b);}
00367
00368 #define fl_set_clipping(x,y,w,h) fl_push_clip(x,y,w,h)
00369 #define fl_unset_clipping() fl_pop_clip()
00370
00372 // Forms classes:
00373
00374 inline Fl_Widget* fl_add_new(Fl_Widget* p) {return p;}
00375 inline Fl_Widget* fl_add_new(uchar t,Fl_Widget* p) {p->type(t); return p;}
00376
00377 #define forms_constructor(type,name) \
00378 inline type* name(uchar t,int x,int y,int w,int h,const char* l) { \
00379     return (type*)(fl_add_new(t, new type(x,y,w,h,l)));}
00380 #define forms_constructort(type,name) \
00381 inline type* name(uchar t,int x,int y,int w,int h,const char* l) { \
00382     return (type*)(fl_add_new(new type(t,x,y,w,h,l)));}
00383 #define forms_constructorb(type,name) \
00384 inline type* name(Fl_Boxtype t,int x,int y,int w,int h,const char* l) { \
00385     return (type*)(fl_add_new(new type(t,x,y,w,h,l)));}
00386
00387 #include "Fl_FormsBitmap.H"
00388 #define FL_NORMAL_BITMAP FL_NO_BOX
00389 forms_constructorb(Fl_FormsBitmap, fl_add_bitmap)
00390 inline void fl_set_bitmap_data(Fl_Widget* o, int w, int h, const uchar* b) {
00391     ((Fl_FormsBitmap*)o)->set(w,h,b);
00392 }
00393
00394 #include "Fl_FormsPixmap.H"
00395 #define FL_NORMAL_PIXMAP FL_NO_BOX
00396 forms_constructorb(Fl_FormsPixmap, fl_add_pixmap)
00397 inline void fl_set_pixmap_data(Fl_Widget* o, char*const* b) {
00398     ((Fl_FormsPixmap*)o)->set(b);
00399 }
00400 //inline void fl_set_pixmap_file(Fl_Widget*, const char*);
00401 inline void fl_set_pixmap_align(Fl_Widget* o,Fl_Align a,int,int) {o->align(a);}
00402 //inline void fl_set_pixmap_colorcloseness(int, int, int);
00403
00404 #include "Fl_Box.H"
00405 forms_constructorb(Fl_Box, fl_add_box)
00406
00407 #include "Fl_Browser.H"
00408 forms_constructor(Fl_Browser, fl_add_browser)
00409
00410 inline void fl_clear_browser(Fl_Widget* o) {
00411     ((Fl_Browser*)o)->clear();}
00412 inline void fl_add_browser_line(Fl_Widget* o, const char* s) {
00413     ((Fl_Browser*)o)->add(s);}
00414 inline void fl_addto_browser(Fl_Widget* o, const char* s) {
00415     ((Fl_Browser*)o)->add(s);} /* should also scroll to bottom */
00416 //inline void fl_addto_browser_chars(Fl_Widget*, const char*)
00417 //define fl_append_browser fl_addto_browser_chars
00418 inline void fl_insert_browser_line(Fl_Widget* o, int n, const char* s) {
00419     ((Fl_Browser*)o)->insert(n,s);}
00420 inline void fl_delete_browser_line(Fl_Widget* o, int n) {
00421     ((Fl_Browser*)o)->remove(n);}
00422 inline void fl_replace_browser_line(Fl_Widget* o, int n, const char* s) {
00423     ((Fl_Browser*)o)->replace(n,s);}
00424 inline char* fl_get_browser_line(Fl_Widget* o, int n) {
00425     return (char*)((Fl_Browser*)o)->text(n);}
00426 inline int fl_load_browser(Fl_Widget* o, const char* f) {
00427     return ((Fl_Browser*)o)->load(f);}
00428 inline void fl_select_browser_line(Fl_Widget* o, int n) {
00429     ((Fl_Browser*)o)->select(n,1);}
00430 inline void fl_deselect_browser_line(Fl_Widget* o, int n) {
00431     ((Fl_Browser*)o)->select(n,0);}
00432 inline void fl_deselect_browser(Fl_Widget* o) {
00433     ((Fl_Browser*)o)->deselect();}
00434 inline int fl_isselected_browser_line(Fl_Widget* o, int n) {
00435     return ((Fl_Browser*)o)->selected(n);}
00436 inline int fl_get_browser_topline(Fl_Widget* o) {
00437     return ((Fl_Browser*)o)->topline();}
00438 inline int fl_get_browser(Fl_Widget* o) {
00439     return ((Fl_Browser*)o)->value();}
00440 inline int fl_get_browser_maxline(Fl_Widget* o) {

```

```

00441     return ((Fl_Browser*)o)->size();}
00442 //inline int fl_get_browser_screenlines(Fl_Widget*);
00443 inline void fl_set_browser_topleft(Fl_Widget* o, int n) {
00444     ((Fl_Browser*)o)->topleft(n);}
00445 inline void fl_set_browser_fontsize(Fl_Widget* o, int s) {
00446     ((Fl_Browser*)o)->textsize(s);}
00447 inline void fl_set_browser_fontstyle(Fl_Widget* o, Fl_Font s) {
00448     ((Fl_Browser*)o)->textfont(s);}
00449 inline void fl_set_browser_specialkey(Fl_Widget* o, char c) {
00450     ((Fl_Browser*)o)->format_char(c);}
00451 //inline void fl_set_browser_vscrollbar(Fl_Widget*, int);
00452 //inline void fl_set_browser_hscrollbar(Fl_Widget*, int);
00453 //inline void fl_set_browser_leftslider(Fl_Widget*, int);
00454 //define fl_set_browser_leftscrollbar fl_set_browser_leftslider
00455 //inline void fl_set_browser_line_selectable(Fl_Widget*, int, int);
00456 //inline void fl_get_browser_dimension(Fl_Widget*,int*,int*,int*,int*);
00457 //inline void fl_set_browser_dbclick_callback(Fl_Widget*,FL_CALLBACKPTR,long);
00458 //inline void fl_set_browser_xoffset(Fl_Widget*, FL_Coord);
00459 //inline void fl_set_browser_scrollbarsize(Fl_Widget*, int, int);
00460 inline void fl_setdisplayed_browser_line(Fl_Widget* o, int n, int i) {
00461     ((Fl_Browser*)o)->display(n,i);}
00462 inline int fl_isdisplayed_browser_line(Fl_Widget* o, int n) {
00463     return ((Fl_Browser*)o)->displayed(n);}
00464
00465 #include "Fl_Button.H"
00466
00467 #define FL_NORMAL_BUTTON      0
00468 #define FL_TOUCH_BUTTON      4
00469 #define FL_INOUT_BUTTON      5
00470 #define FL_RETURN_BUTTON      6
00471 #define FL_HIDDEN_RET_BUTTON  7
00472 #define FL_PUSH_BUTTON        FL_TOGGLE_BUTTON
00473 #define FL_MENU_BUTTON        9
00474
00475 FL_EXPORT Fl_Button* fl_add_button(uchar t,int x,int y,int w,int h,const char* l);
00476 inline int fl_get_button(Fl_Widget* b) {return ((Fl_Button*)b)->value();}
00477 inline void fl_set_button(Fl_Widget* b, int v) {((Fl_Button*)b)->value(v);}
00478 inline int fl_get_button_numb(Fl_Widget*) {return Fl::event_button();}
00479 inline void fl_set_button_shortcut(Fl_Widget* b, const char* s,int=0) {
00480     ((Fl_Button*)b)->shortcut(s);}
00481 //define fl_set_object_shortcut(b,s) fl_set_button_shortcut(b,s)
00482
00483 #include "Fl_Light_Button.H"
00484 forms_constructor(Fl_Light_Button, fl_add_lightbutton)
00485
00486 #include "Fl_Round_Button.H"
00487 forms_constructor(Fl_Round_Button, fl_add_roundbutton)
00488 forms_constructor(Fl_Round_Button, fl_add_round3dbutton)
00489
00490 #include "Fl_Check_Button.H"
00491 forms_constructor(Fl_Check_Button, fl_add_checkbutton)
00492
00493 inline Fl_Widget* fl_add_bitmapbutton(int t,int x,int y,int w,int h,const char* l) {Fl_Widget* o =
    fl_add_button(t,x,y,w,h,l); return o;}
00494 inline void fl_set_bitmapbutton_data(Fl_Widget* o,int a,int b,uchar* c) {
00495     (new Fl_Bitmap(c,a,b))->label(o);} // does not delete old Fl_Bitmap!
00496
00497 inline Fl_Widget* fl_add_pixmapbutton(int t,int x,int y,int w,int h,const char* l) {Fl_Widget* o =
    fl_add_button(t,x,y,w,h,l); return o;}
00498 inline void fl_set_pixmapbutton_data(Fl_Widget* o, const char*const* c) {
00499     (new Fl_Pixmap(c))->label(o);} // does not delete old Fl_Pixmap!
00500
00501 // Fl_Canvas object not yet implemented!
00502
00503 #include "Fl_Chart.H"
00504
00505 forms_constructor(Fl_Chart, fl_add_chart)
00506 inline void fl_clear_chart(Fl_Widget* o) {
00507     ((Fl_Chart*)o)->clear();}
00508 inline void fl_add_chart_value(Fl_Widget* o,double v,const char* s,uchar c){
00509     ((Fl_Chart*)o)->add(v,s,c);}
00510 inline void fl_insert_chart_value(Fl_Widget* o, int i, double v, const char* s, uchar c) {
00511     ((Fl_Chart*)o)->insert(i,v,s,c);}
00512 inline void fl_replace_chart_value(Fl_Widget* o, int i, double v, const char* s, uchar c) {
00513     ((Fl_Chart*)o)->replace(i,v,s,c);}
00514 inline void fl_set_chart_bounds(Fl_Widget* o, double a, double b) {
00515     ((Fl_Chart*)o)->bounds(a,b);}
00516 inline void fl_set_chart_maxnumb(Fl_Widget* o, int v) {
00517     ((Fl_Chart*)o)->maxsize(v);}
00518 inline void fl_set_chart_autosize(Fl_Widget* o, int v) {
00519     ((Fl_Chart*)o)->autosize(v);}
00520 inline void fl_set_chart_lstyle(Fl_Widget* o, Fl_Font v) {
00521     ((Fl_Chart*)o)->textfont(v);}
00522 inline void fl_set_chart_lsize(Fl_Widget* o, int v) {
00523     ((Fl_Chart*)o)->textsize(v);}
00524 inline void fl_set_chart_lcolor(Fl_Widget* o, Fl_Color v) {
00525     ((Fl_Chart*)o)->textcolor(v);}

```

```

00526 #define fl_set_chart_lcol    fl_set_chart_lcolor
00527
00528 #include "Fl_Choice.H"
00529
00530 #define FL_NORMAL_CHOICE      0
00531 #define FL_NORMAL_CHOICE2     0
00532 #define FL_DROPLIST_CHOICE    0
00533
00534 forms_constructor(Fl_Choice, fl_add_choice)
00535 inline void fl_clear_choice(Fl_Widget* o) {
00536     ((Fl_Choice*)o)->clear();}
00537 inline void fl_addto_choice(Fl_Widget* o, const char* s) {
00538     ((Fl_Choice*)o)->add(s);}
00539 inline void fl_replace_choice(Fl_Widget* o, int i, const char* s) {
00540     ((Fl_Choice*)o)->replace(i-1,s);}
00541 inline void fl_delete_choice(Fl_Widget* o, int i) {
00542     ((Fl_Choice*)o)->remove(i-1);}
00543 inline void fl_set_choice(Fl_Widget* o, int i) {
00544     ((Fl_Choice*)o)->value(i-1);}
00545 // inline void fl_set_choice_text(Fl_Widget*, const char*);
00546 inline int fl_get_choice(Fl_Widget* o) {
00547     return ((Fl_Choice*)o)->value()+1;}
00548 // inline const char* fl_get_choice_item_text(Fl_Widget*, int);
00549 // inline int fl_get_choice_maxitems(Fl_Widget*);
00550 inline const char* fl_get_choice_text(Fl_Widget* o) {
00551     return ((Fl_Choice*)o)->text();}
00552 inline void fl_set_choice_fontsize(Fl_Widget* o, int x) {
00553     ((Fl_Choice*)o)->textsize(x);}
00554 inline void fl_set_choice_fontstyle(Fl_Widget* o, Fl_Font x) {
00555     ((Fl_Choice*)o)->textfont(x);}
00556 // inline void fl_set_choice_item_mode(Fl_Widget*, int, unsigned);
00557 // inline void fl_set_choice_item_shortcut(Fl_Widget*, int, const char*);
00558
00559 #include "Fl_Clock.H"
00560 forms_constructor(Fl_Clock, fl_add_clock)
00561 inline void fl_get_clock(Fl_Widget* o, int* h, int* m, int* s) {
00562     *h = ((Fl_Clock*)o)->hour();
00563     *m = ((Fl_Clock*)o)->minute();
00564     *s = ((Fl_Clock*)o)->second();
00565 }
00566
00567 #include "Fl_Counter.H"
00568 forms_constructor(Fl_Counter, fl_add_counter)
00569 inline void fl_set_counter_value(Fl_Widget* o, double v) {
00570     ((Fl_Counter*)o)->value(v);}
00571 inline void fl_set_counter_bounds(Fl_Widget* o, double a, double b) {
00572     ((Fl_Counter*)o)->bounds(a,b);}
00573 inline void fl_set_counter_step(Fl_Widget* o, double a, double b) {
00574     ((Fl_Counter*)o)->step(a,b);}
00575 inline void fl_set_counter_precision(Fl_Widget* o, int v) {
00576     ((Fl_Counter*)o)->precision(v);}
00577 inline void fl_set_counter_return(Fl_Widget* o, int v) {
00578     ((Fl_Counter*)o)->when((Fl_When)(v|FL_WHEN_RELEASE));}
00579 inline double fl_get_counter_value(Fl_Widget* o) {
00580     return ((Fl_Counter*)o)->value();}
00581 inline void fl_get_counter_bounds(Fl_Widget* o, float* a, float* b) {
00582     *a = float(((Fl_Counter*)o)->minimum());
00583     *b = float(((Fl_Counter*)o)->maximum());
00584 }
00585 //inline void fl_set_counter_filter(Fl_Widget*,const char* (*)(Fl_Widget*,double,int));
00586
00587 // Cursor stuff cannot be emulated because it uses X stuff
00588 inline void fl_set_cursor(Fl_Window* w, Fl_Cursor c) {w->cursor(c);}
00589 #define FL_INVISIBLE_CURSOR FL_CURSOR_NONE
00590 #define FL_DEFAULT_CURSOR FL_CURSOR_DEFAULT
00591
00592 #include "Fl_Dial.H"
00593
00594 #define FL_DIAL_COL1 FL_GRAY
00595 #define FL_DIAL_COL2 37
00596
00597 forms_constructor(Fl_Dial, fl_add_dial)
00598 inline void fl_set_dial_value(Fl_Widget* o, double v) {
00599     ((Fl_Dial*)o)->value(v);}
00600 inline double fl_get_dial_value(Fl_Widget* o) {
00601     return ((Fl_Dial*)o)->value();}
00602 inline void fl_set_dial_bounds(Fl_Widget* o, double a, double b) {
00603     ((Fl_Dial*)o)->bounds(a, b);}
00604 inline void fl_get_dial_bounds(Fl_Widget* o, float* a, float* b) {
00605     *a = float(((Fl_Dial*)o)->minimum());
00606     *b = float(((Fl_Dial*)o)->maximum());
00607 }
00608 inline void fl_set_dial_return(Fl_Widget* o, int i) {
00609     ((Fl_Dial*)o)->when((Fl_When)(i|FL_WHEN_RELEASE));}
00610 inline void fl_set_dial_angles(Fl_Widget* o, int a, int b) {
00611     ((Fl_Dial*)o)->angles((short)a, (short)b);}
00612 //inline void fl_set_dial_cross(Fl_Widget* o, int);

```



```

00613 // inline void fl_set_dial_direction(Fl_Widget* o, uchar d) {
00614 //     ((Fl_Dial*)o)->direction(d);}
00615 inline void fl_set_dial_step(Fl_Widget* o, double v) {
00616     ((Fl_Dial*)o)->step(v);}
00617
00618 // Frames:
00619
00620 inline Fl_Widget* fl_add_frame(Fl_Boxtype i, int x, int y, int w, int h, const char* l) {
00621     return fl_add_box(i, x-3, y-3, w+6, h+6, l);}
00622
00623 // labelframe nyi
00624 inline Fl_Widget* fl_add_labelframe(Fl_Boxtype i, int x, int y, int w, int h, const char* l) {
00625     Fl_Widget* o = fl_add_box(i, x-3, y-3, w+6, h+6, l);
00626     o->align(FL_ALIGN_TOP_LEFT);
00627     return o;
00628 }
00629
00630 #include "Fl_Free.H"
00631 inline Fl_Free*
00632 fl_add_free(int t, double x, double y, double w, double h, const char* l,
00633             FL_HANDLEPTR hdl) {
00634     return (Fl_Free*)(fl_add_new(
00635         new Fl_Free(t, int(x), int(y), int(w), int(h), l, hdl)));
00636 }
00637
00638 #include "fl_ask.H"
00639 #include "fl_show_colormap.H"
00640
00641 inline int fl_show_question(const char* c, int = 0) {return fl_choice("%s", fl_no, fl_yes, 0L, c);}
00642 FL_EXPORT void fl_show_message(const char *, const char *, const char *);
00643 FL_EXPORT void fl_show_alert(const char *, const char *, const char *, int=0);
00644 FL_EXPORT int fl_show_question(const char *, const char *, const char *);
00645 inline const char *fl_show_input(const char *l, const char*d=0) {return fl_input("%s", d, l);}
00646 FL_EXPORT /*const*/ char *fl_show_simple_input(const char *label, const char *deflt = 0);
00647 FL_EXPORT int fl_show_choice(
00648     const char *m1,
00649     const char *m2,
00650     const char *m3,
00651     int numb,
00652     const char *b0,
00653     const char *b1,
00654     const char *b2);
00655
00656 inline void fl_set_goodies_font(Fl_Font a, Fl_Fontsize b) {fl_message_font(a, b);}
00657 #define fl_show_messages fl_message
00658 inline int fl_show_choices(const char* c, int n, const char* b1, const char* b2,
00659     const char* b3, int) {
00660     return fl_show_choice(0, c, 0, n, b1, b2, b3);
00661 }
00662
00663 #include "filename.H"
00664 #include "Fl_File_Chooser.H"
00665 inline int do_matching(char* a, const char* b) {return fl_filename_match(a, b);}
00666
00667 // Forms-compatible file chooser (implementation in fselect.C):
00668 FL_EXPORT char* fl_show_file_selector(const char* message, const char* dir,
00669     const char* pat, const char* fname);
00670 FL_EXPORT char* fl_get_directory();
00671 FL_EXPORT char* fl_get_pattern();
00672 FL_EXPORT char* fl_get_filename();
00673
00674 #include "Fl_Input.H"
00675 forms_constructor(Fl_Input, fl_add_input)
00676 inline void fl_set_input(Fl_Widget* o, const char* v) {
00677     ((Fl_Input*)o)->value(v);}
00678 inline void fl_set_input_return(Fl_Widget* o, int x) {
00679     ((Fl_Input*)o)->when((Fl_When)(x | FL_WHEN_RELEASE));}
00680 inline void fl_set_input_color(Fl_Widget* o, Fl_Color a, Fl_Color b) {
00681     ((Fl_Input*)o)->textcolor(a);
00682     ((Fl_Input*)o)->cursor_color(b);
00683 }
00684 // inline void fl_set_input_scroll(Fl_Widget*, int);
00685 inline void fl_set_input_cursorpos(Fl_Widget* o, int x, int /*y*/) {
00686     ((Fl_Input*)o)->insert_position(x);}
00687 // inline void fl_set_input_selected(Fl_Widget*, int);
00688 // inline void fl_set_input_selected_range(Fl_Widget*, int, int);
00689 // inline void fl_set_input_maxchars(Fl_Widget*, int);
00690 // inline void fl_set_input_format(Fl_Widget*, int, int);
00691 // inline void fl_set_input_hscrollbar(Fl_Widget*, int);
00692 // inline void fl_set_input_vscrollbar(Fl_Widget*, int);
00693 // inline void fl_set_input_xoffset(Fl_Widget*, int);
00694 // inline void fl_set_input_topline(Fl_Widget*, int);
00695 // inline void fl_set_input_scrollbarsize(Fl_Widget*, int, int);
00696 // inline int fl_get_input_topline(Fl_Widget*);
00697 // inline int fl_get_input_screenlines(Fl_Widget*);
00698 inline int fl_get_input_cursorpos(Fl_Widget* o, int*x, int*y) {
00699     *x = ((Fl_Input*)o)->insert_position(); *y = 0; return *x;}

```



```

00700 // inline int fl_get_input_numberoflines(Fl_Widget*);
00701 // inline void fl_get_input_format(Fl_Widget*, int*, int*);
00702 inline const char* fl_get_input(Fl_Widget* o) {return ((Fl_Input*)o)->value();}
00703
00704 #include "Fl_Menu_Button.H"
00705
00706 // types are not implemented, they all act like FL_PUSH_MENU:
00707 #define FL_TOUCH_MENU 0
00708 #define FL_PUSH_MENU 1
00709 #define FL_PULLDOWN_MENU 2
00710 forms_constructor(Fl_Menu_Button, fl_add_menu)
00711
00712 inline void fl_clear_menu(Fl_Widget* o) {
00713     ((Fl_Menu_Button*)o)->clear();}
00714 inline void fl_set_menu(Fl_Widget* o, const char* s) {
00715     ((Fl_Menu_Button*)o)->clear(); ((Fl_Menu_Button*)o)->add(s);}
00716 inline void fl_addto_menu(Fl_Widget* o, const char* s) {
00717     ((Fl_Menu_Button*)o)->add(s);}
00718 inline void fl_replace_menu_item(Fl_Widget* o, int i, const char* s) {
00719     ((Fl_Menu_Button*)o)->replace(i-1,s);}
00720 inline void fl_delete_menu_item(Fl_Widget* o, int i) {
00721     ((Fl_Menu_Button*)o)->remove(i-1);}
00722 inline void fl_set_menu_item_shortcut(Fl_Widget* o, int i, const char* s) {
00723     ((Fl_Menu_Button*)o)->shortcut(i-1,fl_old_shortcut(s));}
00724 inline void fl_set_menu_item_mode(Fl_Widget* o, int i, long x) {
00725     ((Fl_Menu_Button*)o)->mode(i-1,(int)x);}
00726 inline void fl_show_menu_symbol(Fl_Widget*, int ) {
00727     /* ((Fl_Menu_Button*)o)->show_menu_symbol(i); */}
00728 // inline void fl_set_menu_popup(Fl_Widget*, int);
00729 inline int fl_get_menu(Fl_Widget* o) {
00730     return ((Fl_Menu_Button*)o)->value()+1;}
00731 inline const char* fl_get_menu_item_text(Fl_Widget* o, int i) {
00732     return ((Fl_Menu_Button*)o)->text(i);}
00733 inline int fl_get_menu_maxitems(Fl_Widget* o) {
00734     return ((Fl_Menu_Button*)o)->size();}
00735 inline int fl_get_menu_item_mode(Fl_Widget* o, int i) {
00736     return ((Fl_Menu_Button*)o)->mode(i);}
00737 inline const char* fl_get_menu_text(Fl_Widget* o) {
00738     return ((Fl_Menu_Button*)o)->text();}
00739
00740 #include "Fl_Positioner.H"
00741 #define FL_NORMAL_POSITIONER 0
00742 forms_constructor(Fl_Positioner, fl_add_positioner)
00743 inline void fl_set_positioner_xvalue(Fl_Widget* o, double v) {
00744     ((Fl_Positioner*)o)->xvalue(v);}
00745 inline double fl_get_positioner_xvalue(Fl_Widget* o) {
00746     return ((Fl_Positioner*)o)->xvalue();}
00747 inline void fl_set_positioner_xbounds(Fl_Widget* o, double a, double b) {
00748     ((Fl_Positioner*)o)->xbounds(a,b);}
00749 inline void fl_get_positioner_xbounds(Fl_Widget* o, float* a, float* b) {
00750     *a = float(((Fl_Positioner*)o)->xminimum());}
00751     *b = float(((Fl_Positioner*)o)->xmaximum());}
00752 }
00753 inline void fl_set_positioner_yvalue(Fl_Widget* o, double v) {
00754     ((Fl_Positioner*)o)->yvalue(v);}
00755 inline double fl_get_positioner_yvalue(Fl_Widget* o) {
00756     return ((Fl_Positioner*)o)->yvalue();}
00757 inline void fl_set_positioner_ybounds(Fl_Widget* o, double a, double b) {
00758     ((Fl_Positioner*)o)->ybounds(a,b);}
00759 inline void fl_get_positioner_ybounds(Fl_Widget* o, float* a, float* b) {
00760     *a = float(((Fl_Positioner*)o)->yminimum());}
00761     *b = float(((Fl_Positioner*)o)->ymaximum());}
00762 }
00763 inline void fl_set_positioner_xstep(Fl_Widget* o, double v) {
00764     ((Fl_Positioner*)o)->xstep(v);}
00765 inline void fl_set_positioner_ystep(Fl_Widget* o, double v) {
00766     ((Fl_Positioner*)o)->ystep(v);}
00767 inline void fl_set_positioner_return(Fl_Widget* o, int v) {
00768     ((Fl_Positioner*)o)->when((Fl_When)(v|FL_WHEN_RELEASE));}
00769
00770 #include "Fl_Slider.H"
00771
00772 #define FL_HOR_BROWSER_SLIDER FL_HOR_SLIDER
00773 #define FL_VERT_BROWSER_SLIDER FL_VERT_SLIDER
00774
00775 forms_constructor(Fl_Slider, fl_add_slider)
00776 #define FL_SLIDER_COL1 FL_GRAY
00777 inline void fl_set_slider_value(Fl_Widget* o, double v) {
00778     ((Fl_Slider*)o)->value(v);}
00779 inline double fl_get_slider_value(Fl_Widget* o) {
00780     return ((Fl_Slider*)o)->value();}
00781 inline void fl_set_slider_bounds(Fl_Widget* o, double a, double b) {
00782     ((Fl_Slider*)o)->bounds(a, b);}
00783 inline void fl_get_slider_bounds(Fl_Widget* o, float* a, float* b) {
00784     *a = float(((Fl_Slider*)o)->minimum());}
00785     *b = float(((Fl_Slider*)o)->maximum());}
00786 }

```

```

00787 inline void fl_set_slider_return(Fl_Widget* o, int i) {
00788     ((Fl_Slider*)o)->when((Fl_When) (i|FL_WHEN_RELEASE));}
00789 inline void fl_set_slider_step(Fl_Widget* o, double v) {
00790     ((Fl_Slider*)o)->step(v);}
00791 // inline void fl_set_slider_increment(Fl_Widget* o, double v, double);
00792 inline void fl_set_slider_size(Fl_Widget* o, double v) {
00793     ((Fl_Slider*)o)->slider_size(v);}
00794
00795 #include "Fl_Value_Slider.H"
00796 forms_constructor(Fl_Value_Slider, fl_add_valslider)
00797
00798 inline void fl_set_slider_precision(Fl_Widget* o, int i) {
00799     ((Fl_Value_Slider*)o)->precision(i);}
00800 // filter function!
00801
00802 // The forms text object was the same as an Fl_Box except it inverted the
00803 // meaning of FL_ALIGN_INSIDE. Implementation in forms.cxx
00804 class FL_EXPORT Fl_FormsText : public Fl_Widget {
00805 protected:
00806     void draw() FL_OVERRIDE;
00807 public:
00808     Fl_FormsText(Fl_Boxtype b, int X, int Y, int W, int H, const char* l=0)
00809         : Fl_Widget(X,Y,W,H,l) {box(b); align(FL_ALIGN_LEFT);}
00810 };
00811 #define FL_NORMAL_TEXT FL_NO_BOX
00812 forms_constructorb(Fl_FormsText, fl_add_text)
00813
00814 #include "Fl_Timer.H"
00815 forms_constructort(Fl_Timer, fl_add_timer)
00816 inline void fl_set_timer(Fl_Widget* o, double v) {((Fl_Timer*)o)->value(v);}
00817 inline double fl_get_timer(Fl_Widget* o) {return ((Fl_Timer*)o)->value();}
00818 inline void fl_suspend_timer(Fl_Widget* o) {((Fl_Timer*)o)->suspended(1);}
00819 inline void fl_resume_timer(Fl_Widget* o) {((Fl_Timer*)o)->suspended(0);}
00820 inline void fl_set_timer_countup(Fl_Widget* o, char d) {((Fl_Timer*)o)->direction(d);}
00821 void FL_EXPORT fl_gettime(long* sec, long* usec);
00822
00823 // Fl_XYPlot nyi
00824
00825
00826 // stuff from DDForms:
00827
00828 inline int fl_double_click() {return Fl::event_clicks();}
00829 inline void fl_draw() {Fl::flush();}
00830
00831 #endif /* define __FORMS_H__ */

```

12.185 gl.h File Reference

This file defines wrapper functions for OpenGL in FLTK.

```

#include "Enumerations.H"
#include <GL/gl.h>

```

Functions

- void **gl_color** (Fl_Color i)
Sets the current OpenGL color to an FLTK color.
- void **gl_color** (int c)
back compatibility
- int **gl_descent** ()
Returns the current font's descent.
- void **gl_draw** (const char *)
Draws a nul-terminated string in the current font at the current position.
- void **gl_draw** (const char *, float x, float y)
Draws a nul-terminated string in the current font at the given position.
- void **gl_draw** (const char *, int n)
Draws an array of n characters of the string in the current font at the current position.
- void **gl_draw** (const char *, int n, float x, float y)
Draws n characters of the string in the current font at the given position.
- void **gl_draw** (const char *, int n, int x, int y)

- Draws n characters of the string in the current font at the given position.*
- void **gl_draw** (const char *, int x, int y)
- Draws a nul-terminated string in the current font at the given position.*
- void **gl_draw** (const char *, int x, int y, int w, int h, **FL_Align**)
- Draws a string formatted into a box, with newlines and tabs expanded, other control characters changed to ^X.*
- void **gl_draw_image** (const **uchar** *, int x, int y, int w, int h, int d=3, int ld=0)
- void **gl_finish** ()
- Releases an OpenGL context.*
- void **gl_font** (int fontid, int size)
- Sets the current OpenGL font to the same font as calling **fl_font()**.*
- int **gl_height** ()
- Returns the current font's height.*
- void **gl_measure** (const char *, int &x, int &y)
- Measure how wide and tall the string will be when drawn by the **gl_draw()** function.*
- void **gl_rect** (int x, int y, int w, int h)
- Outlines the given rectangle with the current color.*
- void **gl_rectf** (int x, int y, int w, int h)
- Fills the given rectangle with the current color.*
- void **gl_start** ()
- Creates an OpenGL context.*
- int **gl_texture_pile_height** ()
- Returns the current maximum height of the pile of pre-computed string textures.*
- void **gl_texture_pile_height** (int max)
- Changes the maximum height of the pile of pre-computed string textures.*
- void **gl_texture_reset** ()
- To call after GL operations that may invalidate textures used to draw text in GL scenes (e.g., switch between **FL_↔DOUBLE** / **FL_SINGLE** modes).*
- double **gl_width** (const char *)
- Returns the width of the string in the current font.*
- double **gl_width** (const char *, int n)
- Returns the width of n characters of the string in the current font.*
- double **gl_width** (**uchar**)
- Returns the width of the character in the current font.*

12.185.1 Detailed Description

This file defines wrapper functions for OpenGL in FLTK.

To use OpenGL from within an FLTK application you MUST use **gl_visual()** to select the default visual before doing **show()** on any windows. Mesa will crash if you try to use a visual not returned by **glxChooseVisual**.

Historically, this did not always work well with **FL_Double_Window**'s! It can try to draw into the front buffer. Depending on the system this might either crash or do nothing (when pixmaps are being used as back buffer and GL is being done by hardware), work correctly (when GL is done with software, such as Mesa), or draw into the front buffer and be erased when the buffers are swapped (when double buffer hardware is being used)

12.185.2 Function Documentation

12.185.2.1 **gl_color()**

```
void gl_color (
    Fl_Color i)
```

Sets the current OpenGL color to an FLTK color.

For color-index modes it will use **fl_xpixel(c)**, which is only right if the window uses the default colormap!

12.185.2.2 gl_draw() [1/7]

```
void gl_draw (  
    const char * str)
```

Draws a nul-terminated string in the current font at the current position.

See also

`gl_texture_pile_height(int)`

12.185.2.3 gl_draw() [2/7]

```
void gl_draw (  
    const char * str,  
    float x,  
    float y)
```

Draws a nul-terminated string in the current font at the given position.

See also

`gl_texture_pile_height(int)`

12.185.2.4 gl_draw() [3/7]

```
void gl_draw (  
    const char * str,  
    int n)
```

Draws an array of n characters of the string in the current font at the current position.

See also

`gl_texture_pile_height(int)`

12.185.2.5 gl_draw() [4/7]

```
void gl_draw (  
    const char * str,  
    int n,  
    float x,  
    float y)
```

Draws n characters of the string in the current font at the given position.

See also

`gl_texture_pile_height(int)`

12.185.2.6 gl_draw() [5/7]

```
void gl_draw (  
    const char * str,  
    int n,  
    int x,  
    int y)
```

Draws n characters of the string in the current font at the given position.

See also

`gl_texture_pile_height(int)`

12.185.2.7 gl_draw() [6/7]

```
void gl_draw (  
    const char * str,  
    int x,  
    int y)
```

Draws a nul-terminated string in the current font at the given position.

See also

[gl_texture_pile_height\(int\)](#)

12.185.2.8 gl_draw() [7/7]

```
void gl_draw (  
    const char * str,  
    int x,  
    int y,  
    int w,  
    int h,  
    Fl\_Align align)
```

Draws a string formatted into a box, with newlines and tabs expanded, other control characters changed to ^X. and aligned with the edges or center. Exactly the same output as [fl_draw\(\)](#).

12.185.2.9 gl_font()

```
void gl_font (  
    int fontid,  
    int size)
```

Sets the current OpenGL font to the same font as calling [fl_font\(\)](#).

See also

[Fl::draw_GL_text_with_textures\(int val\)](#)

12.185.2.10 gl_rect()

```
void gl_rect (  
    int x,  
    int y,  
    int w,  
    int h)
```

Outlines the given rectangle with the current color.

If [Fl_Gl_Window::ortho\(\)](#) has been called, then the rectangle will exactly fill the given pixel rectangle.

12.185.2.11 gl_rectf()

```
void gl_rectf (  
    int x,  
    int y,  
    int w,  
    int h)
```

Fills the given rectangle with the current color.

See also

[gl_rect\(int x, int y, int w, int h\)](#)

12.185.2.12 gl_texture_pile_height() [1/2]

```
int gl_texture_pile_height (
    void )
```

Returns the current maximum height of the pile of pre-computed string textures.
The default value is 100

See also

[Fl::draw_GL_text_with_textures\(int\)](#)

12.185.2.13 gl_texture_pile_height() [2/2]

```
void gl_texture_pile_height (
    int max)
```

Changes the maximum height of the pile of pre-computed string textures.
Strings that are often re-displayed can be processed much faster if this pile is set high enough to hold all of them.

Parameters

<i>max</i>	Maximum height of the texture pile

See also

[Fl::draw_GL_text_with_textures\(int\)](#)

12.186 gl.h

[Go to the documentation of this file.](#)

```
00001 //
00002 // OpenGL header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2018 by Bill Spitzak and others.
00005 //
00006 // You must include this instead of GL/gl.h to get the Microsoft
00007 // APIENTRY stuff included (from <windows.h>) prior to the OpenGL
00008 // header files.
00009 //
00010 // This file also provides "missing" OpenGL functions, and
00011 // gl_start() and gl_finish() to allow OpenGL to be used in any window
00012 //
00013 // This library is free software. Distribution and use rights are outlined in
00014 // the file "COPYING" which should have been included with this file. If this
00015 // file is missing or damaged, see the license at:
00016 //
00017 //     https://www.fltk.org/COPYING.php
00018 //
00019 // Please see the following page on how to report bugs and issues:
00020 //
00021 //     https://www.fltk.org/bugs.php
00022 //
00023
00042 #ifndef FL_gl_H
00043 # define FL_gl_H
00044
00045 # include "Enumerations.H" // for color names
00046 # ifdef _WIN32
00047 #   include <windows.h>
00048 # endif
00049 # ifndef APIENTRY
00050 #   if defined(__CYGWIN__)
00051 #     define APIENTRY __attribute__((__stdcall__))
00052 #   else
00053 #     define APIENTRY
00054 #   endif
00055 # endif
00056
00057 # ifdef __APPLE__ // PORTME: OpenGL path abstraction
00058 #   ifndef GL_SILENCE_DEPRECATION
00059 #     define GL_SILENCE_DEPRECATION 1
00060 #   endif
00061 #   if !defined(__gl3_h_) // make sure OpenGL/gl3.h was not included before
00062 #     include <OpenGL/gl.h>
```

```

00063 #     endif
00064 #     else
00065 #         include <GL/gl.h>
00066 #     endif // __APPLE__ // PORTME: OpenGL Path abstraction
00067
00068 FL_EXPORT void gl_start();
00069 FL_EXPORT void gl_finish();
00070
00071 FL_EXPORT void gl_color(Fl_Color i);
00073 inline void gl_color(int c) {gl_color((Fl_Color)c);}
00074
00075 FL_EXPORT void gl_rect(int x,int y,int w,int h);
00076 FL_EXPORT void gl_rectf(int x,int y,int w,int h);
00077
00078 FL_EXPORT void gl_font(int fontid, int size);
00079 FL_EXPORT int gl_height();
00080 FL_EXPORT int gl_descent();
00081 FL_EXPORT double gl_width(const char *);
00082 FL_EXPORT double gl_width(const char *, int n);
00083 FL_EXPORT double gl_width(uchar);
00084
00085 FL_EXPORT void gl_draw(const char*);
00086 FL_EXPORT void gl_draw(const char*, int n);
00087 FL_EXPORT void gl_draw(const char*, int x, int y);
00088 FL_EXPORT void gl_draw(const char*, float x, float y);
00089 FL_EXPORT void gl_draw(const char*, int n, int x, int y);
00090 FL_EXPORT void gl_draw(const char*, int n, float x, float y);
00091 FL_EXPORT void gl_draw(const char*, int x, int y, int w, int h, Fl_Align);
00092 FL_EXPORT void gl_measure(const char*, int& x, int& y);
00093 FL_EXPORT void gl_texture_pile_height(int max);
00094 FL_EXPORT int gl_texture_pile_height();
00095 FL_EXPORT void gl_texture_reset();
00096
00097 FL_EXPORT void gl_draw_image(const uchar *, int x,int y,int w,int h, int d=3, int ld=0);
00098
00099 #endif // !FL_gl_H

```

12.187 gl2opengl.h

```

00001 /*      gl.h
00002
00003      GL to OpenGL translator.
00004      If you include this, you might be able to port old GL programs.
00005      There are also much better emulators available on the net.
00006
00007 */
00008
00009 #ifndef _FL_gl2opengl_h_
00010 #define _FL_gl2opengl_h_
00011
00012 #include <FL/gl.h>
00013 #include "gl_draw.H"
00014
00015 inline void clear() {glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);}
00016 #define RGBcolor(r,g,b) glColor3ub(r,g,b)
00017 #define bgnline() glBegin(GL_LINE_STRIP)
00018 #define bgnpolygon() glBegin(GL_POLYGON)
00019 #define bgnclosedline() glBegin(GL_LINE_LOOP)
00020 #define endline() glEnd()
00021 #define endpolygon() glEnd()
00022 #define endclosedline() glEnd()
00023 #define v2f(v) glVertex2fv(v)
00024 #define v2s(v) glVertex2sv(v)
00025 #define cmov(x,y,z) glRasterPos3f(x,y,z)
00026 #define charstr(s) gl_draw(s)
00027 #define fmprstr(s) gl_draw(s)
00028 typedef float Matrix[4][4];
00029 inline void pushmatrix() {glPushMatrix();}
00030 inline void popmatrix() {glPopMatrix();}
00031 inline void multmatrix(Matrix m) {glMultMatrixf((float *)m);}
00032 inline void color(int n) {glIndexi(n);}
00033 inline void rect(int x,int y,int r,int t) {gl_rect(x,y,r-x,t-y);}
00034 inline void rectf(int x,int y,int r,int t) {glRectf(x,y,r+1,t+1);}
00035 inline void recti(int x,int y,int r,int t) {gl_rect(x,y,r-x,t-y);}
00036 inline void rectfi(int x,int y,int r,int t) {glRecti(x,y,r+1,t+1);}
00037 inline void rects(int x,int y,int r,int t) {gl_rect(x,y,r-x,t-y);}
00038 inline void rectfs(int x,int y,int r,int t) {glRects(x,y,r+1,t+1);}
00039
00040 #endif /* _FL_gl2opengl_h_ */

```

12.188 gl_draw.H

```
00001 //
00002 // OpenGL header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00017 #include "gl.h"
00018 //
00019 extern FL_EXPORT void gl_remove_displaylist_fonts();
```

12.189 glu.h

```
00001 //
00002 // GLu header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // You must include this instead of GL/gl.h to get the Microsoft
00007 // APIENTRY stuff included (from <windows.h>) prior to the OpenGL
00008 // header files.
00009 //
00010 // This file also provides "missing" OpenGL functions, and
00011 // gl_start() and gl_finish() to allow OpenGL to be used in any window
00012 //
00013 // This library is free software. Distribution and use rights are outlined in
00014 // the file "COPYING" which should have been included with this file.  If this
00015 // file is missing or damaged, see the license at:
00016 //
00017 //     https://www.fltk.org/COPYING.php
00018 //
00019 // Please see the following page on how to report bugs and issues:
00020 //
00021 //     https://www.fltk.org/bugs.php
00022 //
00023 //
00024 #ifndef FL_glu_H
00025 # define FL_glu_H
00026 //
00027 # include "Enumerations.H" // for color names
00028 # ifdef _WIN32
00029 #   include <windows.h>
00030 # endif
00031 # ifndef APIENTRY
00032 #   if defined(__CYGWIN__)
00033 #     define APIENTRY __attribute__((__stdcall__))
00034 #   else
00035 #     define APIENTRY
00036 #   endif
00037 # endif
00038 //
00039 # ifdef __APPLE__ // PORTME: OpenGL Path abstraction
00040 #   include <OpenGL/glu.h>
00041 # else
00042 #   include <GL/glu.h>
00043 # endif
00044 //
00045 #endif // !FL_glu_H
```

12.190 glut.H

```
00001 //
00002 // GLUT emulation header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
```



```

00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 // Emulation of GLUT using fltk.
00018
00019 // GLUT is Copyright (c) Mark J. Kilgard, 1994, 1995, 1996:
00020 // "This program is freely distributable without licensing fees and is
00021 // provided without guarantee or warranty expressed or implied. This
00022 // program is -not- in the public domain."
00023
00024 // Although I have copied the GLUT API, none of my code is based on
00025 // any GLUT implementation details and is therefore covered by the LGPL.
00026
00027 // Commented out lines indicate parts of GLUT that are not emulated.
00028
00029 // Notes: as pointed out in STR #3458 the current GLUT window,
00030 // i.e. the global static variable 'glut_window' can be NULL ...
00031 // (a) if not (yet) initialized
00032 // (b) if the current GLUT window is deleted at any time.
00033 // The FLTK implementation silently ignores function calls if the current
00034 // window is NULL to avoid dereferencing a NULL pointer. This is obviously
00035 // compatible with GLUT version 3.7 according to comment #5 on STR #3458.
00036 // According to the same comment FreeGLUT 3.0 would issue an error message
00037 // and quit.
00038 // Albrecht-S, Oct 2023
00039
00040 #ifndef _FL_glut_H_
00041 # define _FL_glut_H_
00042
00043 # include "gl.h"
00044
00045
00046 # include "Fl.H"
00047 # include "Fl_Gl_Window.H"
00048
00053 class FL_EXPORT Fl_Glut_Window : public Fl_Gl_Window {
00054     void _init();
00055     int mouse_down;
00056 protected:
00057     void draw() FL_OVERRIDE;
00058     void draw_overlay() FL_OVERRIDE;
00059     int handle(int) FL_OVERRIDE;
00060 public: // so the inline functions work
00061     int number;
00062     int menu[3];
00063     void make_current();
00064     void (*display)();
00065     void (*overlaydisplay)();
00066     void (*reshape)(int w, int h);
00067     void (*keyboard)(uchar, int x, int y);
00068     void (*mouse)(int b, int state, int x, int y);
00069     void (*motion)(int x, int y);
00070     void (*passivemotion)(int x, int y);
00071     void (*entry)(int);
00072     void (*visibility)(int);
00073     void (*special)(int, int x, int y);
00074     Fl_Glut_Window(int w, int h, const char *t=0);
00075     Fl_Glut_Window(int x, int y, int w, int h, const char *t=0);
00076     ~Fl_Glut_Window();
00077 };
00078
00079 extern FL_EXPORT Fl_Glut_Window *glut_window; // the current window
00080 extern FL_EXPORT int glut_menu; // the current menu
00081
00082 // function pointers that are not per-window:
00083 extern FL_EXPORT void (*glut_idle_function)();
00084 extern FL_EXPORT void (*glut_menustate_function)(int);
00085 extern FL_EXPORT void (*glut_menustatus_function)(int,int,int);
00086
00087
00088
00089 //# define GLUT_API_VERSION This does not match any version of GLUT exactly...
00090
00091 FL_EXPORT void glutInit(int *argcp, char **argv); // creates first window
00092
00093 FL_EXPORT void glutInitDisplayMode(unsigned int mode);
00094 // the FL_ symbols have the same value as the GLUT ones:
00095 # define GLUT_RGB FL_RGB
00096 # define GLUT_RGBA FL_RGBA
00097 # define GLUT_INDEX FL_INDEX
00098 # define GLUT_SINGLE FL_SINGLE
00099 # define GLUT_DOUBLE FL_DOUBLE
00100 # define GLUT_ACCUM FL_ACCUM
00101 # define GLUT_ALPHA FL_ALPHA
00102 # define GLUT_DEPTH FL_DEPTH

```

```

00103 # define GLUT_STENCIL          FL_STENCIL
00104 # define GLUT_MULTISAMPLE       FL_MULTISAMPLE
00105 # define GLUT_STEREO            FL_STEREO
00106 // # define GLUT_LUMINANCE      512
00107
00108 FL_EXPORT void glutInitWindowPosition(int x, int y);
00109
00110 FL_EXPORT void glutInitWindowSize(int w, int h);
00111
00112 FL_EXPORT void glutMainLoop();
00113
00114 FL_EXPORT int glutCreateWindow(char *title);
00115 FL_EXPORT int glutCreateWindow(const char *title);
00116
00117 FL_EXPORT int glutCreateSubWindow(int win, int x, int y, int width, int height);
00118
00119 FL_EXPORT void glutDestroyWindow(int win);
00120
00121 inline void glutPostRedisplay() {
00122     if (glut_window) glut_window->redraw();
00123 }
00124
00125 FL_EXPORT void glutPostWindowRedisplay(int win);
00126
00127 FL_EXPORT void glutSwapBuffers();
00128
00129 inline int glutGetWindow() {
00130     return glut_window ? glut_window->number : 0;
00131 }
00132
00133 FL_EXPORT void glutSetWindow(int win);
00134
00135 inline void glutSetWindowTitle(char *t) {
00136     if (glut_window) glut_window->label(t);
00137 }
00138
00139 inline void glutSetIconTitle(char *t) {
00140     if (glut_window) glut_window->iconlabel(t);
00141 }
00142
00143 inline void glutPositionWindow(int x, int y) {
00144     if (glut_window) glut_window->position(x,y);
00145 }
00146
00147 inline void glutReshapeWindow(int w, int h) {
00148     if (glut_window) glut_window->size(w,h);
00149 }
00150
00151 inline void glutPopWindow() {
00152     if (glut_window) glut_window->show();
00153 }
00154
00155 inline void glutPushWindow() { /* do nothing */ }
00156
00157 inline void glutIconifyWindow() {
00158     if (glut_window) glut_window->iconize();
00159 }
00160
00161 inline void glutShowWindow() {
00162     if (glut_window) glut_window->show();
00163 }
00164
00165 inline void glutHideWindow() {
00166     if (glut_window) glut_window->hide();
00167 }
00168
00169 inline void glutFullScreen() {
00170     if (glut_window) glut_window->fullscreen();
00171 }
00172
00173 inline void glutSetCursor(Fl_Cursor cursor) {
00174     if (glut_window) glut_window->cursor(cursor);
00175 }
00176
00177 // notice that the numeric values are different than glut:
00178 # define GLUT_CURSOR_RIGHT_ARROW    ((Fl_Cursor)2)
00179 # define GLUT_CURSOR_LEFT_ARROW     ((Fl_Cursor)67)
00180 # define GLUT_CURSOR_INFO            FL_CURSOR_HAND
00181 # define GLUT_CURSOR_DESTROY        ((Fl_Cursor)45)
00182 # define GLUT_CURSOR_HELP           FL_CURSOR_HELP
00183 # define GLUT_CURSOR_CYCLE          ((Fl_Cursor)26)
00184 # define GLUT_CURSOR_SPRAY          ((Fl_Cursor)63)
00185 # define GLUT_CURSOR_WAIT           FL_CURSOR_WAIT
00186 # define GLUT_CURSOR_TEXT           FL_CURSOR_INSERT
00187 # define GLUT_CURSOR_CROSSHAIR      FL_CURSOR_CROSS
00188 # define GLUT_CURSOR_UP_DOWN        FL_CURSOR_NS
00189 # define GLUT_CURSOR_LEFT_RIGHT     FL_CURSOR_WE

```

```

00190 # define GLUT_CURSOR_TOP_SIDE          FL_CURSOR_N
00191 # define GLUT_CURSOR_BOTTOM_SIDE       FL_CURSOR_S
00192 # define GLUT_CURSOR_LEFT_SIDE        FL_CURSOR_W
00193 # define GLUT_CURSOR_RIGHT_SIDE       FL_CURSOR_E
00194 # define GLUT_CURSOR_TOP_LEFT_CORNER  FL_CURSOR_NW
00195 # define GLUT_CURSOR_TOP_RIGHT_CORNER FL_CURSOR_NE
00196 # define GLUT_CURSOR_BOTTOM_RIGHT_CORNER FL_CURSOR_SE
00197 # define GLUT_CURSOR_BOTTOM_LEFT_CORNER FL_CURSOR_SW
00198 # define GLUT_CURSOR_INHERIT          FL_CURSOR_DEFAULT
00199 # define GLUT_CURSOR_NONE             FL_CURSOR_NONE
00200 # define GLUT_CURSOR_FULL_CROSSHAIR   FL_CURSOR_CROSS
00201
00202 inline void glutWarpPointer(int, int) { /* do nothing */ }
00203
00204 inline void glutEstablishOverlay() {
00205     if (glut_window) glut_window->make_overlay_current();
00206 }
00207
00208 inline void glutRemoveOverlay() {
00209     if (glut_window) glut_window->hide_overlay();
00210 }
00211
00212 inline void glutUseLayer(GLenum layer) {
00213     if (!glut_window)
00214         return;
00215     layer ? glut_window->make_overlay_current() : glut_window->make_current();
00216 }
00217
00218 enum {GLUT_NORMAL, GLUT_OVERLAY};
00219
00220 inline void glutPostOverlayRedisplay() {
00221     if (glut_window) glut_window->redraw_overlay();
00222 }
00223
00224 inline void glutShowOverlay() {
00225     if (glut_window) glut_window->redraw_overlay();
00226 }
00227
00228 inline void glutHideOverlay() {
00229     if (glut_window) glut_window->hide_overlay();
00230 }
00231
00232 FL_EXPORT int glutCreateMenu(void (*)(int));
00233
00234 FL_EXPORT void glutDestroyMenu(int menu);
00235
00236 inline int glutGetMenu() {return glut_menu;}
00237
00238 inline void glutSetMenu(int m) {glut_menu = m;}
00239
00240 FL_EXPORT void glutAddMenuEntry(const char *label, int value);
00241
00242 FL_EXPORT void glutAddSubMenu(char *label, int submenu);
00243
00244 FL_EXPORT void glutChangeToMenuEntry(int item, char *labela, int value);
00245
00246 FL_EXPORT void glutChangeToSubMenu(int item, char *label, int submenu);
00247
00248 FL_EXPORT void glutRemoveMenuItem(int item);
00249
00250 inline void glutAttachMenu(int b) {
00251     if (glut_window) glut_window->menu[b] = glut_menu;
00252 }
00253
00254 inline void glutDetachMenu(int b) {
00255     if (glut_window) glut_window->menu[b] = 0;
00256 }
00257
00258 inline void glutDisplayFunc(void (*f)()) {
00259     if (glut_window) glut_window->display = f;
00260 }
00261
00262 inline void glutReshapeFunc(void (*f)(int w, int h)) {
00263     if (glut_window) glut_window->reshape = f;
00264 }
00265
00266 inline void glutKeyboardFunc(void (*f)(uchar key, int x, int y)) {
00267     if (glut_window) glut_window->keyboard = f;
00268 }
00269
00270 inline void glutMouseFunc(void (*f)(int b, int state, int x, int y)) {
00271     if (glut_window) glut_window->mouse = f;
00272 }
00273
00274 # define GLUT_LEFT_BUTTON      0
00275 # define GLUT_MIDDLE_BUTTON    1
00276 # define GLUT_RIGHT_BUTTON     2

```

```

00277 # define GLUT_DOWN 0
00278 # define GLUT_UP 1
00279
00280 inline void glutMotionFunc(void (*f)(int x, int y)) {
00281     if (glut_window) glut_window->motion = f;
00282 }
00283
00284 inline void glutPassiveMotionFunc(void (*f)(int x, int y)) {
00285     if (glut_window) glut_window->passivemotion = f;
00286 }
00287
00288 inline void glutEntryFunc(void (*f)(int s)) {
00289     if (glut_window) glut_window->entry = f;
00290 }
00291
00292 enum {GLUT_LEFT, GLUT_ENTERED};
00293
00294 inline void glutVisibilityFunc(void (*f)(int s)) {
00295     if (glut_window) glut_window->visibility = f;
00296 }
00297 enum {GLUT_NOT_VISIBLE, GLUT_VISIBLE};
00298
00299 FL_EXPORT void glutIdleFunc(void (*f)());
00300
00301 inline void glutTimerFunc(unsigned int msec, void (*f)(int), int value) {
00302     Fl::add_timeout(msec*.001, (void *) (void *) f, (void *) (fl_intptr_t) value);
00303 }
00304
00305 inline void glutMenuStateFunc(void (*f)(int state)) {
00306     glut_menustate_function = f;
00307 }
00308 inline void glutMenuStatusFunc(void (*f)(int status, int x, int y)) {
00309     glut_menustatus_function = f;
00310 }
00311 enum {GLUT_MENU_NOT_IN_USE, GLUT_MENU_IN_USE};
00312
00312 inline void glutSpecialFunc(void (*f)(int key, int x, int y)) {
00313     if (glut_window) glut_window->special = f;
00314 }
00315
00316 # define GLUT_KEY_F1 1
00317 # define GLUT_KEY_F2 2
00318 # define GLUT_KEY_F3 3
00319 # define GLUT_KEY_F4 4
00320 # define GLUT_KEY_F5 5
00321 # define GLUT_KEY_F6 6
00322 # define GLUT_KEY_F7 7
00323 # define GLUT_KEY_F8 8
00324 # define GLUT_KEY_F9 9
00325 # define GLUT_KEY_F10 10
00326 # define GLUT_KEY_F11 11
00327 # define GLUT_KEY_F12 12
00328 // WARNING: Different values than GLUT uses:
00329 # define GLUT_KEY_LEFT FL_Left
00330 # define GLUT_KEY_UP FL_Up
00331 # define GLUT_KEY_RIGHT FL_Right
00332 # define GLUT_KEY_DOWN FL_Down
00333 # define GLUT_KEY_PAGE_UP FL_Page_Up
00334 # define GLUT_KEY_PAGE_DOWN FL_Page_Down
00335 # define GLUT_KEY_HOME FL_Home
00336 # define GLUT_KEY_END FL_End
00337 # define GLUT_KEY_INSERT FL_Insert
00338
00339 // inline void glutSpaceballMotionFunc(void (*)(int x, int y, int z));
00340
00341 // inline void glutSpaceballRotateFunc(void (*)(int x, int y, int z));
00342
00343 // inline void glutSpaceballButtonFunc(void (*)(int button, int state));
00344
00345 // inline void glutButtonBoxFunc(void (*)(int button, int state));
00346
00347 // inline void glutDialsFunc(void (*)(int dial, int value));
00348
00349 // inline void glutTabletMotionFunc(void (*)(int x, int y));
00350
00351 // inline void glutTabletButtonFunc(void (*)(int button, int state, int x, int y));
00352
00353 inline void glutOverlayDisplayFunc(void (*f)()) {
00354     if (glut_window) glut_window->overlaydisplay = f;
00355 }
00356
00357 // inline void glutWindowStatusFunc(void (*)(int state));
00358 // enum {GLUT_HIDDEN, GLUT_FULLY_RETAINED, GLUT_PARTIALLY_RETAINED,
00359 //        GLUT_FULLY_COVERED};
00360
00361 // inline void glutSetColor(int, GLfloat red, GLfloat green, GLfloat blue);
00362
00363 // inline GLfloat glutGetColor(int ndx, int component);

```

```

00364 // #define GLUT_RED 0
00365 // #define GLUT_GREEN 1
00366 // #define GLUT_BLUE 2
00367
00368 // inline void glutCopyColormap(int win);
00369
00370 // Warning: values are changed from GLUT!
00371 // Also relies on the GL_ symbols having values greater than 100
00372 FL_EXPORT int glutGet(GLenum type);
00373 enum {
00374     GLUT_RETURN_ZERO = 0,
00375     GLUT_WINDOW_X,
00376     GLUT_WINDOW_Y,
00377     GLUT_WINDOW_WIDTH,
00378     GLUT_WINDOW_HEIGHT,
00379     GLUT_WINDOW_PARENT,
00380     GLUT_SCREEN_WIDTH,
00381     GLUT_SCREEN_HEIGHT,
00382     GLUT_MENU_NUM_ITEMS,
00383     GLUT_DISPLAY_MODE_POSSIBLE,
00384     GLUT_INIT_WINDOW_X,
00385     GLUT_INIT_WINDOW_Y,
00386     GLUT_INIT_WINDOW_WIDTH,
00387     GLUT_INIT_WINDOW_HEIGHT,
00388     GLUT_INIT_DISPLAY_MODE,
00389     GLUT_WINDOW_BUFFER_SIZE,
00390     GLUT_VERSION,
00391     //GLUT_WINDOW_NUM_CHILDREN,
00392     //GLUT_WINDOW_CURSOR,
00393     //GLUT_SCREEN_WIDTH_MM,
00394     //GLUT_SCREEN_HEIGHT_MM,
00395     GLUT_ELAPSED_TIME
00396 };
00397
00398 # define GLUT_WINDOW_STENCIL_SIZE GL_STENCIL_BITS
00399 # define GLUT_WINDOW_DEPTH_SIZE GL_DEPTH_BITS
00400 # define GLUT_WINDOW_RED_SIZE GL_RED_BITS
00401 # define GLUT_WINDOW_GREEN_SIZE GL_GREEN_BITS
00402 # define GLUT_WINDOW_BLUE_SIZE GL_BLUE_BITS
00403 # define GLUT_WINDOW_ALPHA_SIZE GL_ALPHA_BITS
00404 # define GLUT_WINDOW_ACCUM_RED_SIZE GL_ACCUM_RED_BITS
00405 # define GLUT_WINDOW_ACCUM_GREEN_SIZE GL_ACCUM_GREEN_BITS
00406 # define GLUT_WINDOW_ACCUM_BLUE_SIZE GL_ACCUM_BLUE_BITS
00407 # define GLUT_WINDOW_ACCUM_ALPHA_SIZE GL_ACCUM_ALPHA_BITS
00408 # define GLUT_WINDOW_DOUBLEBUFFER GL_DOUBLEBUFFER
00409 # define GLUT_WINDOW_RGBA GL_RGBA
00410 # define GLUT_WINDOW_COLORMAP_SIZE GL_INDEX_BITS
00411 # ifdef GL_SAMPLES_SGIS
00412 #     define GLUT_WINDOW_NUM_SAMPLES GL_SAMPLES_SGIS
00413 # else
00414 #     define GLUT_WINDOW_NUM_SAMPLES GLUT_RETURN_ZERO
00415 # endif
00416 # define GLUT_WINDOW_STEREO GL_STEREO
00417
00418 # define GLUT_HAS_KEYBOARD 600
00419 # define GLUT_HAS_MOUSE 601
00420 # define GLUT_HAS_SPACEBALL 602
00421 # define GLUT_HAS_DIAL_AND_BUTTON_BOX 603
00422 # define GLUT_HAS_TABLET 604
00423 # define GLUT_NUM_MOUSE_BUTTONS 605
00424 # define GLUT_NUM_SPACEBALL_BUTTONS 606
00425 # define GLUT_NUM_BUTTON_BOX_BUTTONS 607
00426 # define GLUT_NUM_DIALS 608
00427 # define GLUT_NUM_TABLET_BUTTONS 609
00428 FL_EXPORT int glutDeviceGet(GLenum type);
00429
00430 // WARNING: these values are different than GLUT uses:
00431 # define GLUT_ACTIVE_SHIFT FL_SHIFT
00432 # define GLUT_ACTIVE_CTRL FL_CTRL
00433 # define GLUT_ACTIVE_ALT FL_ALT
00434
00435 inline int glutGetModifiers() {
00436     return Fl::event_state() & (GLUT_ACTIVE_SHIFT | GLUT_ACTIVE_CTRL | GLUT_ACTIVE_ALT);
00437 }
00438
00439 FL_EXPORT int glutLayerGet(GLenum);
00440 # define GLUT_OVERLAY_POSSIBLE 800
00441 // #define GLUT_LAYER_IN_USE 801
00442 // #define GLUT_HAS_OVERLAY 802
00443 # define GLUT_TRANSPARENT_INDEX 803
00444 # define GLUT_NORMAL_DAMAGED 804
00445 # define GLUT_OVERLAY_DAMAGED 805
00446
00447 extern "C" {
00448     typedef void (*GLUTproc)();
00449 }
00450

```

```

00451 FL_EXPORT GLUTproc glutGetProcAddress(const char *procName);
00452
00453 // inline int glutVideoResizeGet(GLenum param);
00454 // #define GLUT_VIDEO_RESIZE_POSSIBLE 900
00455 // #define GLUT_VIDEO_RESIZE_IN_USE 901
00456 // #define GLUT_VIDEO_RESIZE_X_DELTA 902
00457 // #define GLUT_VIDEO_RESIZE_Y_DELTA 903
00458 // #define GLUT_VIDEO_RESIZE_WIDTH_DELTA 904
00459 // #define GLUT_VIDEO_RESIZE_HEIGHT_DELTA 905
00460 // #define GLUT_VIDEO_RESIZE_X 906
00461 // #define GLUT_VIDEO_RESIZE_Y 907
00462 // #define GLUT_VIDEO_RESIZE_WIDTH 908
00463 // #define GLUT_VIDEO_RESIZE_HEIGHT 909
00464
00465 // inline void glutSetupVideoResizing();
00466
00467 // inline void glutStopVideoResizing();
00468
00469 // inline void glutVideoResize(int x, int y, int width, int height);
00470
00471 // inline void glutVideoPan(int x, int y, int width, int height);
00472
00473 // Font argument must be a void* for compatibility, so...
00475 struct Fl_Glut_Bitmap_Font {Fl_Font font; Fl_Fonsize size;};
00476
00477 extern FL_EXPORT struct Fl_Glut_Bitmap_Font
00478 glutBitmap9By15, glutBitmap8By13, glutBitmapTimesRoman10,
00479 glutBitmapTimesRoman24, glutBitmapHelvetica10, glutBitmapHelvetica12,
00480 glutBitmapHelvetica18;
00481 # define GLUT_BITMAP_9_BY_15 (&glutBitmap9By15)
00482 # define GLUT_BITMAP_8_BY_13 (&glutBitmap8By13)
00483 # define GLUT_BITMAP_TIMES_ROMAN_10 (&glutBitmapTimesRoman10)
00484 # define GLUT_BITMAP_TIMES_ROMAN_24 (&glutBitmapTimesRoman24)
00485 # define GLUT_BITMAP_HELVETICA_10 (&glutBitmapHelvetica10)
00486 # define GLUT_BITMAP_HELVETICA_12 (&glutBitmapHelvetica12)
00487 # define GLUT_BITMAP_HELVETICA_18 (&glutBitmapHelvetica18)
00488
00489 FL_EXPORT void glutBitmapCharacter(void *font, int character);
00490 FL_EXPORT int glutBitmapHeight(void *font);
00491 FL_EXPORT int glutBitmapLength(void *font, const unsigned char *string);
00492 FL_EXPORT void glutBitmapString(void *font, const unsigned char *string);
00493 FL_EXPORT int glutBitmapWidth(void *font, int character);
00494
00495 FL_EXPORT int glutExtensionSupported(char *name);
00496
00497 /* GLUT stroked font sub-API */
00498 struct Fl_Glut_StrokeVertex {
00499     GLfloat X, Y;
00500 };
00501
00502 struct Fl_Glut_StrokeStrip {
00503     int Number;
00504     const Fl_Glut_StrokeVertex* Vertices;
00505 };
00506
00507 struct Fl_Glut_StrokeChar {
00508     GLfloat Right;
00509     int Number;
00510     const Fl_Glut_StrokeStrip* Strips;
00511 };
00512
00513 struct Fl_Glut_StrokeFont {
00514     char* Name; // The source font name
00515     int Quantity; // Number of chars in font
00516     GLfloat Height; // Height of the characters
00517     const Fl_Glut_StrokeChar** Characters; // The characters mapping
00518 };
00519 extern FL_EXPORT Fl_Glut_StrokeFont glutStrokeRoman;
00520 extern FL_EXPORT Fl_Glut_StrokeFont glutStrokeMonoRoman;
00521 # define GLUT_STROKE_ROMAN (&glutStrokeRoman)
00522 # define GLUT_STROKE_MONO_ROMAN (&glutStrokeMonoRoman)
00523
00524 FL_EXPORT void glutStrokeCharacter(void *font, int character);
00525 FL_EXPORT GLfloat glutStrokeHeight(void *font);
00526 FL_EXPORT int glutStrokeLength(void *font, const unsigned char *string);
00527 FL_EXPORT void glutStrokeString(void *font, const unsigned char *string);
00528 FL_EXPORT int glutStrokeWidth(void *font, int character);
00529
00530 /* GLUT pre-built models sub-API */
00531 FL_EXPORT void glutWireSphere(GLdouble radius, GLint slices, GLint stacks);
00532 FL_EXPORT void glutSolidSphere(GLdouble radius, GLint slices, GLint stacks);
00533 FL_EXPORT void glutWireCone(GLdouble base, GLdouble height, GLint slices, GLint stacks);
00534 FL_EXPORT void glutSolidCone(GLdouble base, GLdouble height, GLint slices, GLint stacks);
00535 FL_EXPORT void glutWireCube(GLdouble size);
00536 FL_EXPORT void glutSolidCube(GLdouble size);
00537 FL_EXPORT void glutWireTorus(GLdouble innerRadius, GLdouble outerRadius, GLint sides, GLint rings);
00538 FL_EXPORT void glutSolidTorus(GLdouble innerRadius, GLdouble outerRadius, GLint sides, GLint rings);

```

```

00539 FL_EXPORT void glutWireDodecahedron();
00540 FL_EXPORT void glutSolidDodecahedron();
00541 FL_EXPORT void glutWireTeapot(GLdouble size);
00542 FL_EXPORT void glutSolidTeapot(GLdouble size);
00543 FL_EXPORT void glutWireOctahedron();
00544 FL_EXPORT void glutSolidOctahedron();
00545 FL_EXPORT void glutWireTetrahedron();
00546 FL_EXPORT void glutSolidTetrahedron();
00547 FL_EXPORT void glutWireIcosahedron();
00548 FL_EXPORT void glutSolidIcosahedron();
00549
00550 #endif // !_FL_glut_H_

```

12.191 mac.H File Reference

Mac OS X-specific symbols.

Classes

- class [FL_Mac_App_Menu](#)

Functions

- [FL_Window](#) * [fl_mac_find](#) (FLWindow *)
Returns the [FL_Window](#) corresponding to the given macOS-specific window reference.
- CGContextRef [fl_mac_gc](#) ()
Returns the macOS-specific graphics context for the current window.
- void [fl_mac_set_about](#) ([FL_Callback](#) *cb, void *user_data, int shortcut=0)
Attaches a callback to the "About myprog" item of the system application menu.
- FLWindow * [fl_mac_xid](#) (const [FL_Window](#) *win)
Returns the macOS-specific window reference corresponding to the given [FL_Window](#) object.

Variables

- int [fl_mac_os_version](#)
The version number of the running Mac OS X (e.g., 100604 for 10.6.4, 101300 for 10.13, 140102 for 14.1.2).

12.191.1 Detailed Description

Mac OS X-specific symbols.

12.192 mac.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Mac header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2018 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 // Do not directly include this file, instead use <FL/platform.H>. It will
00018 // include this file if "__APPLE__" is defined. This is to encourage
00019 // portability of even the system-specific code...
00020 #ifndef FL_DOXYGEN
00021
00022 #if !defined(FL_PLATFORM_H)
00023 #   error "Never use <FL/mac.H> directly; include <FL/platform.H> instead."

```

```

00024 #endif // !FL_PLATFORM_H
00025
00026 #ifdef __OBJC__
00027     @class NSOpenGLContext;
00028     #   ifndef GL_SILENCE_DEPRECATION
00029     #       define GL_SILENCE_DEPRECATION 1
00030     #   endif
00031 #elif defined(__cplusplus)
00032     class NSOpenGLContext;
00033 #endif /* __OBJC__ */
00034 extern NSOpenGLContext *fl_mac_glcontext(GLContext rc);
00035
00036 #ifdef __OBJC__
00037 @class FLWindow; // a subclass of the NSWindow Cocoa class
00038 typedef FLWindow *Window;
00039 #else
00040 typedef class FLWindow *Window; // pointer to the FLWindow objective-c class
00041 #endif // __OBJC__
00042
00043 #include <FL/Fl_Widget.H> // for Fl_Callback
00044
00045 #if (defined(FL_LIBRARY) || defined(FL_INTERNALS)) // this part must be compiled when building the
    FLTK libraries
00046
00047 // Standard MacOS C/C++ includes...
00048 #include <ApplicationServices/ApplicationServices.h>
00049 #undef check // because of Fl::check()
00050
00051 #ifndef MAC_OS_X_VERSION_10_4
00052 #define MAC_OS_X_VERSION_10_4 1040
00053 #endif
00054 #ifndef MAC_OS_X_VERSION_10_5
00055 #define MAC_OS_X_VERSION_10_5 1050
00056 #endif
00057 #ifndef MAC_OS_X_VERSION_10_6
00058 #define MAC_OS_X_VERSION_10_6 1060
00059 #endif
00060 #ifndef MAC_OS_X_VERSION_10_7
00061 #define MAC_OS_X_VERSION_10_7 1070
00062 #endif
00063 #ifndef MAC_OS_X_VERSION_10_8
00064 #define MAC_OS_X_VERSION_10_8 1080
00065 #endif
00066 #ifndef MAC_OS_X_VERSION_10_9
00067 #define MAC_OS_X_VERSION_10_9 1090
00068 #endif
00069 #ifndef MAC_OS_X_VERSION_10_10
00070 #define MAC_OS_X_VERSION_10_10 101000
00071 #endif
00072 #ifndef MAC_OS_X_VERSION_10_11
00073 #define MAC_OS_X_VERSION_10_11 101100
00074 #endif
00075 #ifndef MAC_OS_X_VERSION_10_12
00076 #define MAC_OS_X_VERSION_10_12 101200
00077 #endif
00078 #ifndef MAC_OS_X_VERSION_10_13
00079 #define MAC_OS_X_VERSION_10_13 101300
00080 #endif
00081 #ifndef MAC_OS_X_VERSION_10_14
00082 #define MAC_OS_X_VERSION_10_14 101400
00083 #endif
00084 #ifndef MAC_OS_X_VERSION_10_15
00085 #define MAC_OS_X_VERSION_10_15 101500
00086 #endif
00087 #ifndef MAC_OS_X_VERSION_10_16
00088 #define MAC_OS_X_VERSION_10_16 101600
00089 #endif
00090 #ifndef MAC_OS_VERSION_11_0
00091 #define MAC_OS_VERSION_11_0 110000
00092 #endif
00093 #ifndef MAC_OS_VERSION_12_0
00094 #define MAC_OS_VERSION_12_0 120000
00095 #endif
00096 #ifndef MAC_OS_VERSION_13_0
00097 #define MAC_OS_VERSION_13_0 130000
00098 #endif
00099 #ifndef MAC_OS_VERSION_14_0
00100 #define MAC_OS_VERSION_14_0 140000
00101 #endif
00102 #ifndef MAC_OS_VERSION_15_0
00103 #define MAC_OS_VERSION_15_0 150000
00104 #endif
00105
00106
00107 #ifndef NSINTEGER_DEFINED // appears with 10.5 in NSObjCRuntime.h
00108 #if defined(__LP64__) && __LP64__
00109     typedef long NSInteger;

```



```

00110 typedef unsigned long NSInteger;
00111 #else
00112 typedef int NSInteger;
00113 typedef unsigned int NSUInteger;
00114 #endif
00115 #endif
00116
00117 #if MAC_OS_X_VERSION_MAX_ALLOWED < MAC_OS_X_VERSION_10_4
00118 typedef CGImageAlphaInfo CGBitmapInfo;
00119 #endif
00120
00121 struct flCocoaRegion {
00122     int count;
00123     CGRect *rects;
00124 }; // a region is the union of a series of rectangles
00125
00126 #ifndef CGFLOAT_DEFINED //appears with 10.5 in CGBase.h
00127 #if defined(__LP64__) && __LP64__
00128 typedef double CGFloat;
00129 #else
00130 typedef float CGFloat;
00131 #endif
00132 #endif // CGFLOAT_DEFINED
00133
00134 #else
00135
00136 typedef struct CGContext* CGContextRef;
00137
00138 #endif // FL_LIBRARY || FL_INTERNALS
00139
00140 extern CGContextRef fl_gc;
00141
00142 #endif // FL_DOXYGEN
00157 void fl_mac_set_about(Fl_Callback *cb, void *user_data, int shortcut = 0);
00158
00160 extern CGContextRef fl_mac_gc();
00162 extern FLWindow *fl_mac_xid(const FL_Window *win);
00164 extern FL_Window *fl_mac_find(FLWindow *);
00165 class Fl_Gl_Window;
00166
00172 extern int fl_mac_os_version;
00173
00174 struct Fl_Menu_Item;
00175
00176 class Fl_Mac_App_Menu {
00177 public:
00179     static const char *about;
00184     static const char *print;
00186     static const char *print_no_titlebar;
00188     static const char *toggle_print_titlebar;
00190     static const char *services;
00192     static const char *hide;
00194     static const char *hide_others;
00196     static const char *show;
00198     static const char *quit;
00204     static void custom_application_menu_items(const Fl_Menu_Item *m);
00205 };
00206

```

12.193 math.h

```

00001 //
00002 // Math header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2020 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 // Xcode on macOS includes files by recursing down into directories.
00018 // This code catches the cycle and directly includes the required file.
00019 #ifdef fl_math_h_cyclic_include
00020 # include "/usr/include/math.h"
00021 #endif
00022
00023 #ifndef fl_math_h
00024 # define fl_math_h

```

```

00025
00026 # define fl_math_h_cyclic_include
00027 # include <math.h>
00028 # undef fl_math_h_cyclic_include
00029
00030 # ifndef M_PI
00031 #     define M_PI            3.14159265358979323846
00032 #     define M_PI_2          1.57079632679489661923
00033 #     define M_PI_4          0.78539816339744830962
00034 #     define M_1_PI          0.31830988618379067154
00035 #     define M_2_PI          0.63661977236758134308
00036 # endif // !M_PI
00037
00038 # ifndef M_SQRT2
00039 #     define M_SQRT2          1.41421356237309504880
00040 #     define M_SQRT1_2        0.70710678118654752440
00041 # endif // !M_SQRT2
00042
00043 # if (defined(_WIN32) || defined(CRAY)) && !defined(__MINGW32__)
00044
00045 inline double rint(double v) {return floor(v+.5);}
00046 inline double copysign(double a, double b) {return b<0 ? -a : a;}
00047
00048 # endif // (_WIN32 || CRAY) && !__MINGW32__
00049
00050 #endif // !fl_math_h

```

12.194 names.h File Reference

This file defines arrays of human readable names for FLTK symbolic constants.

Variables

- const char *const [fl_callback_reason_names](#) []
This is an array of callback reason names you can use to convert callback reasons into names.
- const char *const [fl_eventnames](#) []
This is an array of event names you can use to convert event numbers into names.
- const char *const [fl_fontnames](#) []
This is an array of font names you can use to convert font numbers into names.

12.194.1 Detailed Description

This file defines arrays of human readable names for FLTK symbolic constants.

12.195 names.h

[Go to the documentation of this file.](#)

```

00001 //
00002 // Event and other names header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 // Thanks to Greg Ercolano for this addition.
00018
00024 #ifndef FL_NAMES_H
00025 #define FL_NAMES_H
00026
00046 const char * const fl_eventnames[] =
00047 {
00048     "FL_NO_EVENT",
00049     "FL_PUSH",
00050     "FL_RELEASE",

```

```

00051     "FL_ENTER",
00052     "FL_LEAVE",
00053     "FL_DRAG",
00054     "FL_FOCUS",
00055     "FL_UNFOCUS",
00056     "FL_KEYDOWN",
00057     "FL_KEYUP",
00058     "FL_CLOSE",
00059     "FL_MOVE",
00060     "FL_SHORTCUT",
00061     "FL_DEACTIVATE",
00062     "FL_ACTIVATE",
00063     "FL_HIDE",
00064     "FL_SHOW",
00065     "FL_PASTE",
00066     "FL_SELECTIONCLEAR",
00067     "FL_MOUSEWHEEL",
00068     "FL_DND_ENTER",
00069     "FL_DND_DRAG",
00070     "FL_DND_LEAVE",
00071     "FL_DND_RELEASE",
00072     "FL_SCREEN_CONFIGURATION_CHANGED",
00073     "FL_FULLSCREEN",
00074     "FL_ZOOM_GESTURE",
00075     "FL_ZOOM_EVENT",
00076     "FL_EVENT_28", // not yet defined, just in case it /will/ be defined ...
00077     "FL_EVENT_29", // not yet defined, just in case it /will/ be defined ...
00078     "FL_EVENT_30" // not yet defined, just in case it /will/ be defined ...
00079 };
00080
00098 const char * const fl_fontnames[] =
00099 {
00100     "FL_HELVETICA",
00101     "FL_HELVETICA_BOLD",
00102     "FL_HELVETICA_ITALIC",
00103     "FL_HELVETICA_BOLD_ITALIC",
00104     "FL_COURIER",
00105     "FL_COURIER_BOLD",
00106     "FL_COURIER_ITALIC",
00107     "FL_COURIER_BOLD_ITALIC",
00108     "FL_TIMES",
00109     "FL_TIMES_BOLD",
00110     "FL_TIMES_ITALIC",
00111     "FL_TIMES_BOLD_ITALIC",
00112     "FL_SYMBOL",
00113     "FL_SCREEN",
00114     "FL_SCREEN_BOLD",
00115     "FL_ZAPF_DINGBATS",
00116 };
00117
00123 const char * const fl_callback_reason_names[] =
00124 {
00125     "FL_REASON_UNKNOWN",
00126     "FL_REASON_SELECTED",
00127     "FL_REASON_DESELECTED",
00128     "FL_REASON_RESELECTED",
00129     "FL_REASON_OPENED",
00130     "FL_REASON_CLOSED",
00131     "FL_REASON_DRAGGED",
00132     "FL_REASON_CANCELLED",
00133     "FL_REASON_CHANGED",
00134     "FL_REASON_GOT_FOCUS",
00135     "FL_REASON_LOST_FOCUS",
00136     "FL_REASON_RELEASED",
00137     "FL_REASON_ENTER_KEY",
00138     NULL, NULL, NULL,
00139     NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL,
00140     NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL,
00141     "FL_REASON_USER", "FL_REASON_USER+1", "FL_REASON_USER+2", "FL_REASON_USER+3",
00142 };
00143
00146 #endif /* FL_NAMES_H */

```

12.196 platform.H

```

00001 //
00002 // Platform header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php

```

```

00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 // These are FLTK symbols that are necessary or useful for calling
00018 // platform specific functions. This file #include's certain platform
00019 // specific system header files that are necessary to declare platform
00020 // specific FLTK functions, for instance "Windows.h" under Windows.
00021
00022 // You should include this file if (and ONLY if) you need to call
00023 // platform specific functions directly.
00024
00025 // See FLTK documentation: chapter "Operating System Issues" on when
00026 // you need to #include <FL/platform.H>
00027
00028 #if !defined(FL_PLATFORM_H) && !defined(FL_DOXYGEN)
00029 #   define FL_PLATFORM_H
00030
00031 #   include <FL/Fl_Export.H>
00032 #   include <FL/platform_types.h> // will bring in FL/fl_config.h
00033 #   include <FL/fl_types.h> // for uchar
00034 class Fl_Window;
00035
00036 #   ifdef _WIN32
00037 #       include "win32.H"
00038 #   elif defined(FLTK_USE_WAYLAND)
00039 #       include "wayland.H"
00040 #   elif defined(FLTK_USE_X11)
00041 #       include "x11.H"
00042 #   elif defined(__APPLE__)
00043 #       include "mac.H"
00044 #   endif // _WIN32
00045
00046 //
00047 // cross-platform declarations
00048 //
00049 #if defined(FL_LIBRARY) || defined(FL_INTERNALS)
00050 #   include <FL/Fl_Window.H>
00051
00052 class FL_EXPORT Fl_X {
00053 public:
00054     fl_uintptr_t xid;
00055     Fl_Window* w;
00056     Fl_Region region;
00057     Fl_X *next;
00058     // static variables, static functions and member functions
00059     static Fl_X* first;
00060     static Fl_X* flx(const Fl_Window* w) {return w ? (Fl_X*)w->flx_ : 0;}
00061 #   if defined(FLTK_USE_X11) && FLTK_USE_X11 // for backward compatibility
00062     static void make_xid(Fl_Window*, XVisualInfo* =fl_visual, Colormap=fl_colormap);
00063     static Fl_X* set_xid(Fl_Window*, Window);
00064     static inline Fl_X* i(const Fl_Window* w) {return flx(w);}
00065 #   endif
00066 };
00067
00068 inline Window fl_xid(const Fl_Window* w) { Fl_X *xTemp = Fl_X::flx(w); return xTemp ?
(Window)xTemp->xid : 0; }
00069 #else
00070 extern FL_EXPORT Window fl_xid_(const Fl_Window* w);
00071 #   define fl_xid(w) fl_xid_(w)
00072 #endif // FL_LIBRARY || FL_INTERNALS
00073
00074 extern FL_EXPORT Fl_Window* fl_find(Window xid);
00075 extern FL_EXPORT void fl_open_display();
00076 extern FL_EXPORT void fl_close_display();
00077 extern FL_EXPORT Window fl_window;
00078 extern FL_EXPORT int fl_parse_color(const char* p, uchar& r, uchar& g, uchar& b);
00079 extern FL_EXPORT void fl_open_callback(void (*) (const char *));
00080
00081 #endif // !FL_PLATFORM_H

```

12.197 platform_types.h File Reference

Definitions of platform-dependent types.

Macros

- #define **FL_COMMAND** opaque

An alias for FL_CTRL on Windows and X11, or FL_META on MacOS X.

- `#define FL_CONTROL` opaque
An alias for FL_META on Windows and X11, or FL_CTRL on MacOS X.

Typedefs

- typedef opaque `fl_intptr_t`
An integral type large enough to store a pointer or a long value.
- typedef opaque `FL_Offscreen`
Platform-specific value representing an offscreen drawing buffer.
- typedef struct opaque * `FL_Region`
Pointer to a platform-specific structure representing a collection of rectangles.
- typedef opaque `FL_SOCKET`
socket or file descriptor
- typedef opaque `FL_Timestamp`
Platform-specific point in time, used for delta time calculation.
- typedef opaque `fl_uintptr_t`
An unsigned integral type large enough to store a pointer or an unsigned long value.
- typedef struct opaque * `GLContext`
Pointer to a platform-specific structure representing the window's OpenGL rendering context.

12.197.1 Detailed Description

Definitions of platform-dependent types.

The exact nature of these types varies with the platform. Therefore, portable FLTK applications should not assume these types have a specific size, or that they are pointers.

12.197.2 Typedef Documentation

12.197.2.1 fl_intptr_t

```
typedef opaque fl_intptr_t
```

An integral type large enough to store a pointer or a long value.

A pointer value can be safely cast to `fl_intptr_t`, and later cast back to its initial pointer type without change to the pointer value. A variable of type `fl_intptr_t` can also store a long int value.

12.197.2.2 FL_Offscreen

```
typedef opaque FL_Offscreen
```

Platform-specific value representing an offscreen drawing buffer.

Note

This value can be safely cast to these types on each platform:

- X11: `Pixmap`
- Wayland: `cairo_t *`
- Windows: `HBITMAP`
- macOS: `CGContextRef`

12.197.2.3 FL_Region

```
typedef struct opaque* FL_Region
```

Pointer to a platform-specific structure representing a collection of rectangles.

Note

This pointer can be safely cast to these types on each platform:

- X11: Region as defined by X11
- Wayland: `cairo_region_t *`
- Windows: `HRGN`
- macOS: `struct flCocoaRegion *`

12.197.2.4 Fl_Timestamp

```
typedef opaque Fl_Timestamp
```

Platform-specific point in time, used for delta time calculation.

Note

This type may be a struct. `sizeof(Fl_Timestamp)` may be different on different platforms. `Fl_Timestamp` may change with future ABI changes.

12.197.2.5 fl_uintptr_t

```
typedef opaque fl_uintptr_t
```

An unsigned integral type large enough to store a pointer or an unsigned long value.

A pointer value can be safely cast to `fl_uintptr_t`, and later cast back to its initial pointer type without change to the pointer value. A variable of type `fl_uintptr_t` can also store an unsigned long int value.

12.197.2.6 GLContext

```
typedef struct opaque* GLContext
```

Pointer to a platform-specific structure representing the window's OpenGL rendering context.

Note

This pointer can be safely cast to these types on each platform:

- X11: `GLXContext`
- Wayland: `EGLContext`
- Windows: `HGLRC`
- macOS: `NSOpenGLContext *`

12.198 platform_types.h

[Go to the documentation of this file.](#)

```
00001 /*
00002  * Copyright 2016-2023 by Bill Spitzak and others.
00003  *
00004  * This library is free software. Distribution and use rights are outlined in
00005  * the file "COPYING" which should have been included with this file. If this
00006  * file is missing or damaged, see the license at:
00007  *
00008  *     https://www.fltk.org/COPYING.php
00009  *
00010  * Please see the following page on how to report bugs and issues:
00011  *
00012  *     https://www.fltk.org/bugs.php
00013  */
00014
00015 #ifndef Fl_Platform_Types_H
00016 #define Fl_Platform_Types_H
00017
00025 #ifdef FL_DOXYGEN
00026
00031 typedef opaque fl_intptr_t;
00036 typedef opaque fl_uintptr_t;
00037
00046 typedef opaque Fl_Offscreen;
```

```

00047
00056 typedef struct opaque *Fl_Region;
00057 typedef opaque FL_SOCKET;
00066 typedef struct opaque *GLContext;
00067
00073 typedef opaque Fl_Timestamp;
00074
00075 # define FL_COMMAND opaque
00076 # define FL_CONTROL opaque
00078 #else /* FL_DOXYGEN */
00079
00080 #ifndef FL_PLATFORM_TYPES_H
00081 #define FL_PLATFORM_TYPES_H
00082
00083 #include <FL/fl_config.h>
00084 #include <time.h> // for time_t
00085
00086 /* Platform-dependent types are defined here.
00087 These types must be defined by any platform:
00088 FL_SOCKET, struct dirent, fl_intptr_t, fl_uintptr_t
00089
00090 NOTE: *FIXME* AlbrechtS 13 Apr 2016 (concerning FL_SOCKET)
00091 -----
00092 The Fl::add_fd() API is partially inconsistent because some of the methods
00093 explicitly use 'int', but the callback typedefs use FL_SOCKET. With the
00094 definition of FL_SOCKET below we can have different data sizes and
00095 different signedness of socket numbers on *some* platforms.
00096 */
00097
00098 #ifdef _WIN64
00099
00100 #if defined(_MSC_VER) && (_MSC_VER < 1600)
00101 # include <stdint.h> /* stdint.h not available before VS 2010 (1600) */
00102 #else
00103 # include <stdint.h>
00104 #endif
00105
00106 typedef intptr_t fl_intptr_t;
00107 typedef uintptr_t fl_uintptr_t;
00108
00109 #else /* !_WIN64 */
00110
00111 typedef long fl_intptr_t;
00112 typedef unsigned long fl_uintptr_t;
00113
00114 #endif /* !_WIN64 */
00115
00116 typedef void *GLContext;
00117 typedef void *Fl_Region;
00118 typedef fl_uintptr_t Fl_Offscreen;
00119
00120 /* Allows all hybrid combinations except WIN32 + X11 with MSVC */
00121 #if defined(_WIN32) && !defined(__MINGW32__)
00122 struct dirent {char d_name[1];};
00123 #else
00124 # include <dirent.h>
00125 #endif
00126
00127 # if defined(_WIN64) && defined(_MSC_VER)
00128 typedef unsigned __int64 FL_SOCKET; /* *FIXME* - FL_SOCKET (see above) */
00129 # else
00130 typedef int FL_SOCKET;
00131 # endif
00132
00133 #include <FL/Fl_Export.H>
00134 extern FL_EXPORT int fl_command_modifier();
00135 extern FL_EXPORT int fl_control_modifier();
00136 # define FL_COMMAND fl_command_modifier()
00137 # define FL_CONTROL fl_control_modifier()
00138
00139 #endif /* FL_PLATFORM_TYPES_H */
00140
00141 /* This is currently the same for all platforms but may change in the future */
00142 struct Fl_Timestamp_t {
00143     time_t sec;
00144     int usec;
00145 };
00146
00147 typedef struct Fl_Timestamp_t Fl_Timestamp;
00148
00149 #endif /* FL_DOXYGEN */
00150
00151 #endif /* Fl_Platform_Types_H */
00152

```

12.199 wayland.H File Reference

Definitions of functions specific to the Wayland platform.

Typedefs

- typedef struct _cairo **cairo_t**
- typedef void * **EGLContext**

Functions

- int **fl_wl_buffer_scale** ([FL_Window](#) *window)
Returns the current buffer scaling factor for window.
- struct wl_compositor * **fl_wl_compositor** ()
Returns the wl_compositor of the current Wayland session.
- struct wl_display * **fl_wl_display** ()
Returns the Wayland display in use.
- [FL_Window](#) * **fl_wl_find** (struct wld_window *)
Returns the [FL_Window](#) corresponding to a given the platform-specific window reference.
- cairo_t * **fl_wl_gc** ()
Returns the cairo context associated to the current window or [FL_Image_Surface](#).
- EGLContext **fl_wl_glcontext** ([GLContext](#) rc)
Returns the EGLContext corresponding to the given GLContext.
- struct wl_surface * **fl_wl_surface** (struct wld_window *xid)
Returns the wl_surface associated to a shown window.
- struct wld_window * **fl_wl_xid** (const [FL_Window](#) *win)
Returns a platform-specific reference associated to a shown window.

12.199.1 Detailed Description

Definitions of functions specific to the Wayland platform.

12.199.2 Function Documentation

12.199.2.1 fl_wl_compositor()

```
struct wl_compositor * fl_wl_compositor () [extern]
```

Returns the wl_compositor of the current Wayland session.

This allows, for example, to create a wl_surface with

```
struct wl_surface *my_wl_surface = wl_compositor_create_surface(fl_wl_compositor());
```

12.200 wayland.H

[Go to the documentation of this file.](#)

```
00001 //
00002 // Wayland/X11 hybrid platform header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #if !defined(FL_PLATFORM_H)
00018 #   error "Never use <FL/wayland.H> directly; include <FL/platform.H> instead."
```



```

00019 #endif // !FL_PLATFORM_H
00020
00025 // ***** for Wayland component *****
00026
00027 typedef struct _cairo cairo_t;
00028
00030 extern FL_EXPORT struct wl_display *fl_wl_display();
00032 extern FL_EXPORT struct wl_surface *fl_wl_surface(struct wld_window *xid);
00034 extern FL_EXPORT struct wld_window *fl_wl_xid(const FL_Window *win);
00036 extern FL_EXPORT FL_Window *fl_wl_find(struct wld_window *);
00038 extern FL_EXPORT cairo_t *fl_wl_gc();
00045 extern FL_EXPORT struct wl_compositor *fl_wl_compositor();
00047 extern FL_EXPORT int fl_wl_buffer_scale(FL_Window *window);
00048 typedef void *EGLContext;
00050 extern FL_EXPORT EGLContext fl_wl_glcontext(GLContext rc);
00051
00052 #ifndef FL_DOXYGEN
00053
00054 #   ifdef FLTK_USE_X11
00055 // ***** for X11 component *****
00056 #       include "x11.H"
00057 #   else
00058         typedef struct wld_window *Window;
00059 #   endif
00060
00061 #endif // FL_DOXYGEN

```

12.201 win32.H File Reference

Definitions of functions specific to the Windows platform.

Functions

- **HINSTANCE fl_win32_display ()**
Returns the Windows-specific display in use
- **FL_Window * fl_win32_find (HWND)**
Returns the [FL_Window](#) corresponding to the given Windows-specific window reference.
- **HDC fl_win32_gc ()**
Returns the Windows-specific graphics context for the current window
- **HGLRC fl_win32_glcontext (GLContext rc)**
Returns the Windows-specific GL rendering context corresponding to the given GLContext
- **HWND fl_win32_xid (const [FL_Window](#) *win)**
Returns the Windows-specific window reference corresponding to the given [FL_Window](#) object.

12.201.1 Detailed Description

Definitions of functions specific to the Windows platform.

12.202 win32.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // Windows platform header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //

```

```

00016
00017 // Do not directly include this file, instead use <FL/platform.H>. It will
00018 // include this file if _WIN32 is defined. This is to encourage
00019 // portability of even the system-specific code...
00020
00021 #ifdef FL_DOXYGEN
00022
00028 extern HWND fl_win32_xid(const Fl_Window *win);
00030 extern Fl_Window *fl_win32_find(HWND);
00032 extern HGLRC fl_win32_glcontext(GLContext rc);
00034 extern HDC fl_win32_gc();
00036 extern HINSTANCE fl_win32_display();
00037
00038 #else
00039
00040 #ifndef FL_PLATFORM_H
00041 # error "Never use <FL/win32.H> directly; include <FL/platform.H> instead."
00042 #endif // !FL_PLATFORM_H
00043
00044 #include <windows.h>
00045 typedef HWND Window;
00046
00047 typedef struct HGLRC__ *HGLRC;
00048 extern FL_EXPORT HGLRC fl_win32_glcontext(GLContext rc);
00049 extern FL_EXPORT HWND fl_win32_xid(const Fl_Window *win);
00050 extern FL_EXPORT Fl_Window *fl_win32_find(HWND);
00051
00052 // this part is included only when compiling the FLTK library or if requested explicitly
00053 #if defined(FL_LIBRARY) || defined(FL_INTERNALS)
00054
00055 // In some of the distributions, the gcc header files are missing some stuff:
00056 #ifndef LPMINMAXINFO
00057 #define LPMINMAXINFO MINMAXINFO*
00058 #endif
00059 #ifndef VK_LWIN
00060 #define VK_LWIN 0x5B
00061 #define VK_RWIN 0x5C
00062 #define VK_APPS 0x5D
00063 #endif
00064
00065 extern FL_EXPORT UINT fl_wake_msg;
00066 extern FL_EXPORT char fl_override_redirect; // hack into Fl_Window::make_xid()
00067 extern FL_EXPORT HPALETTE fl_palette; // non-zero only on 8-bit displays!
00068 extern void fl_release_dc(HWND w, HDC dc);
00069 extern FL_EXPORT void fl_save_dc( HWND w, HDC dc);
00070
00071 #endif // FL_LIBRARY || FL_INTERNALS
00072
00073 // most recent fl_color() or fl_rgbcolor() points at one of these:
00074 extern FL_EXPORT struct Fl_XMap {
00075     COLORREF rgb; // this should be the type the RGB() macro returns
00076     HPEN pen; // pen, 0 if none created yet
00077     int brush; // ref to solid brush, 0 if none created yet
00078     int pwidth; // the width of the pen, if present
00079 } *fl_current_xmap;
00080 inline COLORREF fl_RGB() {return fl_current_xmap->rgb;}
00081 inline HPEN fl_pen() {return fl_current_xmap->pen;}
00082 FL_EXPORT HBRUSH fl_brush(); // allocates a brush if necessary
00083 FL_EXPORT HBRUSH fl_brush_action(int); // now does the real work
00084
00085 extern FL_EXPORT HINSTANCE fl_display;
00086 extern FL_EXPORT HINSTANCE fl_win32_display();
00087 extern FL_EXPORT HDC fl_gc;
00088 extern FL_EXPORT HDC fl_win32_gc();
00089 extern FL_EXPORT MSG fl_msg;
00090 extern FL_EXPORT HDC fl_GetDC(Window);
00091 extern FL_EXPORT HDC fl_makeDC(HBITMAP);
00092
00093 #endif // FL_DOXYGEN

```

12.203 x.H

```

00001 //
00002 // *Deprecated* platform header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2018 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //

```

```

00014 //      https://www.fltk.org/bugs.php
00015 //
00016
00017 // IMPORTANT: This file is deprecated since FLTK 1.4.0. DO NOT include it.
00018 // FL/x.H will be removed in a future FLTK release.
00019
00020 // Please #include <FL/platform.H> instead if you really need it. See
00021 // documentation in FL/platform.H to decide whether you need that file.
00022
00023 #if !defined(FL_X_H) && !defined(FL_DOXYGEN)
00024 #   define FL_X_H
00025 #   include <FL/platform.H>
00026 #endif // !FL_X_H

```

12.204 x11.H File Reference

Definitions of functions specific to the X11 platform.

Functions

- `cairo_t * fl_cairo_gc ()`
Returns the Cairo-specific currently active graphics context (*FLTK_GRAPHICS_CAIRO=On*)
- `Display * fl_x11_display ()`
Returns the X11 Display in use.
- `Fl_Window * fl_x11_find (Window xid)`
Returns the *Fl_Window* corresponding to the given Window reference.
- `GC fl_x11_gc ()`
Returns the X11-specific currently active graphics context.
- `void fl_x11_use_display (Display *d)`
Have FLTK use a pre-established X11 connection.
- `Window fl_x11_xid (const Fl_Window *win)`
Returns the Window reference for the given *Fl_Window*, or zero if not *shown()*.

12.204.1 Detailed Description

Definitions of functions specific to the X11 platform.

12.204.2 Function Documentation

12.204.2.1 fl_x11_find()

```

Fl_Window * fl_x11_find (
    Window xid) [extern]

```

Returns the *Fl_Window* corresponding to the given Window reference.

12.204.2.2 fl_x11_gc()

```

GC fl_x11_gc () [extern]

```

Returns the X11-specific currently active graphics context.

12.204.2.3 fl_x11_use_display()

```

void fl_x11_use_display (
    Display * d) [extern]

```

Have FLTK use a pre-established X11 connection.

This function must be called before FLTK attempts to open its own X11 connection, that is, as long as *fl_x11_display()* returns NULL.

Parameters

<i>d</i>	the X11 Display* value representing a valid, pre-established X11 connection

12.204.2.4 fl_x11_xid()

Window fl_x11_xid (
 const [Fl_Window](#) * win) [extern]

Returns the Window reference for the given [Fl_Window](#), or zero if not shown().

12.205 x11.H

[Go to the documentation of this file.](#)

```

00001 //
00002 // X11 platform header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 // Do not directly include this file, instead use <FL/platform.H>. It will
00018 // include this file if FLTK_USE_X11 is defined. This is to encourage
00019 // portability of even the system-specific code...
00020
00021 #ifdef FL_DOXYGEN
00022
00023 extern Display *fl_x11_display();
00024 extern void fl_x11_use_display(Display *d);
00025 extern Window fl_x11_xid(const Fl_Window *win);
00026 extern Fl_Window *fl_x11_find(Window xid);
00027 extern GC fl_x11_gc();
00028 extern cairo_t* fl_cairo_gc();
00029 #else // ! FL_DOXYGEN
00030
00031 #ifndef FL_PLATFORM_H
00032 #   error "Never use <FL/x11.H> directly; include <FL/platform.H> instead."
00033 #endif // !FL_PLATFORM_H
00034
00035 #include <FL/Enumerations.H>
00036
00037 #if defined(_ABIN32) || defined(_ABI64) // fix for broken SGI Irix X .h files
00038 #   pragma set woff 3322
00039 #endif
00040
00041 #include <X11/Xlib.h>
00042 #include <X11/Xutil.h>
00043 #if defined(_ABIN32) || defined(_ABI64)
00044 #   pragma reset woff 3322
00045 #endif
00046
00047 #include <X11/Xatom.h>
00048
00049 typedef struct __GLXcontextRec *GLXContext;
00050 extern GLXContext fl_x11_glcontext(GLXContext rc);
00051
00052 // constant info about the X server connection:
00053 extern FL_EXPORT Display *fl_display;
00054 extern FL_EXPORT Display *fl_x11_display();
00055 extern FL_EXPORT void fl_x11_use_display(Display *);
00056 extern FL_EXPORT Window fl_x11_xid(const Fl_Window *win);
00057 extern FL_EXPORT Fl_Window *fl_x11_find(Window);
00058 extern FL_EXPORT int fl_screen;
00059 extern FL_EXPORT XVisualInfo *fl_visual;
00060 extern FL_EXPORT Colormap fl_colormap;
00061
00062 // drawing functions:
00063 extern FL_EXPORT GC fl_gc;
00064 #if FLTK_USE_CAIRO

```

```

00077     typedef struct _cairo cairo_t;
00078     extern FL_EXPORT cairo_t* fl_cairo_gc();
00079 #endif
00080 extern FL_EXPORT GC fl_x11_gc();
00081 FL_EXPORT ulong fl_xpixel(Fl_Color i);
00082 FL_EXPORT ulong fl_xpixel(uchar r, uchar g, uchar b);
00083
00084 // feed events into fltk:
00085 FL_EXPORT int fl_handle(const XEvent&);
00086
00087 // you can use these in Fl::add_handler() to look at events:
00088 extern FL_EXPORT const XEvent* fl_xevent;
00089 extern FL_EXPORT ulong fl_event_time;
00090
00091 #if defined(FL_LIBRARY) || defined(FL_INTERNALS)
00092 extern FL_EXPORT Window fl_message_window;
00093 extern FL_EXPORT void *fl_xftfont;
00094
00095 // access to core fonts:
00096 // This class provides a "smart pointer" that returns a pointer to an XFontStruct.
00097 // The global variable fl_xfont can be called wherever a bitmap "core" font is
00098 // needed, e.g. when rendering to a GL context under X11.
00099 // With Xlib / X11 fonts, fl_xfont will return the current selected font.
00100 // With XFT / X11 fonts, fl_xfont will attempt to return the bitmap "core" font most
00101 // similar to (usually the same as) the current XFT font.
00102 class FL_EXPORT Fl_XFont_On_Demand
00103 {
00104 public:
00105     Fl_XFont_On_Demand(XFontStruct* p = NULL) : ptr(p) { }
00106     Fl_XFont_On_Demand& operator=(const Fl_XFont_On_Demand& x)
00107     { ptr = x.ptr; return *this; }
00108     Fl_XFont_On_Demand& operator=(XFontStruct* p)
00109     { ptr = p; return *this; }
00110     XFontStruct* value();
00111     operator XFontStruct*() { return value(); }
00112     XFontStruct& operator*() { return *value(); }
00113     XFontStruct* operator->() { return value(); }
00114     bool operator==(const Fl_XFont_On_Demand& x) { return ptr == x.ptr; }
00115     bool operator!=(const Fl_XFont_On_Demand& x) { return ptr != x.ptr; }
00116 private:
00117     XFontStruct *ptr;
00118 };
00119 extern FL_EXPORT Fl_XFont_On_Demand fl_xfont;
00120
00121 extern FL_EXPORT char fl_override_redirect; // hack into Fl_X::make_xid()
00122
00123 #endif // FL_LIBRARY || FL_INTERNALS
00124
00125 #endif // FL_DOXYGEN

```

12.206 cgdebug.h

```

00001 //
00002 // OS X Core Graphics debugging help for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00017 // This file allows easier debugging of Mac OS X Core Graphics
00018 // code. This file is normally not included into any FLTK builds,
00019 // but since it has proven to be tremendously useful in debugging
00020 // the FLTK port to "Quartz", I decided to add this file in case
00021 // more bugs show up.
00022 //
00023 // This header is activated by adding the following
00024 // line to "config.h"
00025 //     #include "src/cgdebug.h"
00026 //
00027 // Running "./configure" will remove this line from "config.h".
00028 //
00029 // When used erroneously, Core Graphics prints warnings to
00030 // stderr. This is helpful, however it is not possible to
00031 // associate a line number or source file with the warning message.
00032 // This header file outputs a trace of CG calls, interweaving
00033 // them with CG warnings.

```

```

00034 //
00035 // Matthias
00036
00037 #ifndef CGDEBUG
00038 #define CGDEBUG
00039
00040 #include <stdio.h>
00041 #include <Carbon/Carbon.h>
00042
00043 //+BitmapContextCreate
00044 //+BitmapContextGetData
00045 // ClipCGContextToRegion
00046 // QDBeginCGContext
00047 // QDEndCGContext
00048
00049 //+AddArc
00050 //+AddLineToPoint
00051 // ClipToRect
00052 // ClosePath
00053 //+ConcatCTM
00054 //+DrawImage
00055 // FillPath
00056 // FillRect
00057 // Flush
00058 //+GetCTM
00059 // MoveToPoint
00060 //+Release
00061 // RestoreGState
00062 // SaveGState
00063 //+ScaleCTM
00064 //+SetLineCap
00065 //+SetLineDash
00066 //+SetLineJoin
00067 //+SetLineWidth
00068 //+SetRGBFillColor
00069 //+SetRGBStrokeColor
00070 //+SetShouldAntialias
00071 //+SetTextMatrix
00072 //+StrokePath
00073 //+TranslateCTM
00074
00075 inline OSStatus dbgLocation(const char *file, int line)
00076 {
00077     fprintf(stderr, "%s:%d ", file, line);
00078     return 0;
00079 }
00080
00081 inline OSStatus dbgEndl()
00082 {
00083     fprintf(stderr, "\n");
00084     return 0;
00085 }
00086
00087
00088 inline void dbgCGContextClipToRect(CGContextRef a, CGRect b)
00089 {
00090     CGContextClipToRect(a, b);
00091 }
00092
00093 #define CGContextClipToRect(a, b) { \
00094     fprintf(stderr, "%s:%d ", __FILE__, __LINE__); \
00095     dbgCGContextClipToRect(a, b); \
00096     fprintf(stderr, "\n"); }
00097
00098 inline void dbgCGContextFillRect(CGContextRef a, CGRect b)
00099 {
00100     CGContextFillRect(a, b);
00101 }
00102
00103 #define CGContextFillRect(a, b) { \
00104     fprintf(stderr, "%s:%d ", __FILE__, __LINE__); \
00105     dbgCGContextFillRect(a, b); \
00106     fprintf(stderr, "\n"); }
00107
00108 inline OSStatus dbgQDEndCGContext(CGrafPtr a, CGContextRef *b)
00109 {
00110     return QDEndCGContext(a, b);
00111 }
00112
00113 #define QDEndCGContext(a, b) ( \
00114     dbgLocation(__FILE__, __LINE__) + \
00115     dbgQDEndCGContext(a, b) + \
00116     dbgEndl() )
00117
00118 inline OSStatus dbgQDBeginCGContext(CGrafPtr a, CGContextRef *b)
00119 {
00120     return QDBeginCGContext(a, b);

```

```

00121 }
00122
00123 #define QDBeginCGContext(a, b) ( \
00124     dbgLocation(__FILE__, __LINE__) + \
00125     dbgQDBeginCGContext(a, b) + \
00126     dbgEndl() )
00127
00128 inline void dbgClipCGContextToRegion(CGContextRef a, const Rect *b, RgnHandle c)
00129 {
00130     ClipCGContextToRegion(a, b, c);
00131 }
00132
00133 #define ClipCGContextToRegion(a, b, c) { \
00134     fprintf(stderr, "%s:%d ", __FILE__, __LINE__); \
00135     dbgClipCGContextToRegion(a, b, c); \
00136     fprintf(stderr, "\n"); }
00137
00138 inline void dbgCGContextMoveToPoint(CGContextRef context, float x, float y)
00139 {
00140     CGContextMoveToPoint(context, x, y);
00141 }
00142
00143 #define CGContextMoveToPoint(a, b, c) { \
00144     fprintf(stderr, "%s:%d ", __FILE__, __LINE__); \
00145     dbgCGContextMoveToPoint(a, b, c); \
00146     fprintf(stderr, "\n"); }
00147
00148 inline void dbgCGContextFillPath(CGContextRef context)
00149 {
00150     CGContextFillPath(context);
00151 }
00152
00153 #define CGContextFillPath(a) { \
00154     fprintf(stderr, "%s:%d ", __FILE__, __LINE__); \
00155     dbgCGContextFillPath(a); \
00156     fprintf(stderr, "\n"); }
00157
00158 inline void dbgCGContextClosePath(CGContextRef context)
00159 {
00160     CGContextClosePath(context);
00161 }
00162
00163 #define CGContextClosePath(a) { \
00164     fprintf(stderr, "%s:%d ", __FILE__, __LINE__); \
00165     dbgCGContextClosePath(a); \
00166     fprintf(stderr, "\n"); }
00167
00168 inline void dbgCGContextFlush(CGContextRef context)
00169 {
00170     CGContextFlush(context);
00171 }
00172
00173 #define CGContextFlush(a) { \
00174     fprintf(stderr, "%s:%d ", __FILE__, __LINE__); \
00175     dbgCGContextFlush(a); \
00176     fprintf(stderr, "\n"); }
00177
00178 inline void dbgCGContextSaveGState(CGContextRef context)
00179 {
00180     CGContextSaveGState(context);
00181 }
00182
00183 #define CGContextSaveGState(a) { \
00184     fprintf(stderr, "%s:%d ", __FILE__, __LINE__); \
00185     dbgCGContextSaveGState(a); \
00186     fprintf(stderr, "\n"); }
00187
00188 inline void dbgCGContextRestoreGState(CGContextRef context)
00189 {
00190     CGContextRestoreGState(context);
00191 }
00192
00193 #define CGContextRestoreGState(a) { \
00194     fprintf(stderr, "%s:%d ", __FILE__, __LINE__); \
00195     dbgCGContextRestoreGState(a); \
00196     fprintf(stderr, "\n"); }
00197
00198
00199 #endif
00200

```

12.207 fastarrow.h

```

00001 #define fastarrow_width 16
00002 #define fastarrow_height 16

```

```

00003 static const unsigned char fastarrow_bits[] = {
00004     0x00, 0x00, 0x00, 0x07, 0xe0, 0x07, 0xfc, 0x03, 0xff, 0xff, 0xfc, 0x03,
00005     0xe0, 0x07, 0x00, 0x07, 0xe0, 0x00, 0xe0, 0x07, 0xc0, 0x3f, 0xff, 0xff,
00006     0xc0, 0x3f, 0xe0, 0x07, 0xe0, 0x00, 0x00, 0x00};

```

12.208 Fl.cxx File Reference

Implementation of the member functions of class [Fl](#).

```

#include <FL/Fl.H>
#include <FL/platform.H>
#include "Fl_Screen_Driver.H"
#include "Fl_Window_Driver.H"
#include "Fl_System_Driver.H"
#include "Fl_Timeout.h"
#include <FL/Fl_Window.H>
#include <FL/Fl_Tooltip.H>
#include <FL/fl_draw.H>
#include <ctype.h>
#include <stdlib.h>
#include "flstring.h"

```

Macros

- `#define FOREVER 1e20`

Functions

- `bool fl_clipboard_notify_empty (void)`
- `void fl_close_display ()`
Closes the connection to the windowing system when that's possible.
- `const char * fl_filename_name (const char *name)`
Gets the file name from a path.
- `Fl_Window * fl_find (Window xid)`
Returns the [Fl_Window](#) that corresponds to the given window reference, or `NULL` if not found.
- `void fl_fix_focus ()`
- `void fl_open_callback (void(*cb)(const char *))`
Register a function called for each file dropped onto an application icon.
- `void fl_open_display ()`
Opens the display.
- `int fl_send_system_handlers (void *e)`
- `void fl_throw_focus (Fl_Widget *o)`
- `void fl_trigger_clipboard_notify (int source)`
- `Window fl_xid_ (const Fl_Window *w)`

Variables

- `bool fl_disable_wayland = true`
Prevent the FLTK library from using its Wayland backend and force it to use its X11 backend.
- `const char * fl_local_alt = Fl::system_driver()->alt_name()`
string pointer used in shortcuts, you can change it to another language
- `const char * fl_local_ctrl = Fl::system_driver()->control_name()`
string pointer used in shortcuts, you can change it to another language
- `int(* fl_local_grab)(int)`
- `const char * fl_local_meta = Fl::system_driver()->meta_name()`
string pointer used in shortcuts, you can change it to another language

- `const char * fl_local_shift = Fl::system_driver()->shift_name()`
string pointer used in shortcuts, you can change it to another language
- `Fl_Widget * fl_oldfocus`
- `Fl_Widget * fl_selection_requestor`

12.208.1 Detailed Description

Implementation of the member functions of class `Fl`.

12.208.2 Function Documentation

12.208.2.1 fl_close_display()

```
void fl_close_display ()
```

Closes the connection to the windowing system when that's possible.

You do *not* need to call this to exit, and in fact it is faster to not do so. It may be useful to call this if you want your program to continue without a GUI. You cannot open the display again, and cannot call any FLTK functions.

Note

Requires `#include <FL/platform.H>`

12.208.2.2 fl_find()

```
Fl_Window * fl_find (
    Window xid)
```

Returns the `Fl_Window` that corresponds to the given window reference, or `NULL` if not found.

Deprecated Kept in the X11, Windows, and macOS platforms for compatibility with FLTK versions before 1.4. Please use `fl_x11_find(Window)`, `fl_wl_find(struct wld_window*)`, `fl_win32_find(HWND)` or `fl_mac_find(FLWindow*)` with FLTK 1.4.0 and above.

12.208.2.3 fl_open_display()

```
void fl_open_display ()
```

Opens the display.

Automatically called by the library when the first window is `show()`'n. Does nothing if the display is already open.

Note

Requires `#include <FL/platform.H>`

12.208.3 Variable Documentation

12.208.3.1 fl_disable_wayland

```
bool fl_disable_wayland = true
```

Prevent the FLTK library from using its Wayland backend and force it to use its X11 backend.

Put this declaration somewhere in your source code outside the body of any function:

```
FL_EXPORT bool fl_disable_wayland = true;
```

This declaration makes sure that source code developed for FLTK 1.3, including X11-specific code, will build and run with FLTK 1.4 and its Wayland platform with a single line source code level change. This declaration has no effect on non-Wayland platforms. Don't add this declaration if you want the Wayland backend to be used when it's available.

Note

Please see also chapter 2.1 of `README.Wayland.txt` for further information on how to build your application to ensure that this declaration is effective.

Since

1.4.0

12.209 fl_arc.cxx File Reference

Utility functions for drawing arcs and circles.

```
#include <FL/fl_draw.H>
#include <FL/math.h>
```

12.209.1 Detailed Description

Utility functions for drawing arcs and circles.

12.210 fl_ask.cxx File Reference

Utility functions for common dialogs.

```
#include <FL/Fl.H>
#include <FL/Fl_Box.H>
#include <FL/Fl_Input_.H>
#include "flstring.h"
#include "Fl_Screen_Driver.H"
#include <FL/fl_ask.H>
#include "Fl_Message.h"
#include <stdio.h>
#include <stdarg.h>
```

Functions

- void [fl_alert](#) (const char *fmt,...)
Shows an alert message dialog box.
- int [fl_ask](#) (const char *fmt,...)
*Shows a dialog displaying the *fmt* message, this dialog features 2 yes/no buttons.*
- void [fl_beep](#) (int type)
Emits a system beep.
- int [fl_choice](#) (const char *fmt, const char *b0, const char *b1, const char *b2,...)
*Shows a dialog displaying the printf style *fmt* message.*
- int [fl_choice_n](#) (const char *fmt, const char *b0, const char *b1, const char *b2,...)
*Shows a dialog displaying the printf style *fmt* message.*
- const char * [fl_input](#) (const char *fmt, const char *defstr,...)
*Shows an input dialog displaying the *fmt* message with variable arguments.*
- const char * [fl_input](#) (int maxchar, const char *fmt, const char *defstr,...)
*Shows an input dialog displaying the *fmt* message with variable arguments.*
- void [fl_message](#) (const char *fmt,...)
Shows an information message dialog box.
- int [fl_message_hotspot](#) ()
Gets whether or not to move the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#) to follow the mouse pointer.
- void [fl_message_hotspot](#) (int enable)
Sets whether or not to move the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#) to follow the mouse pointer.
- [Fl_Widget](#) * [fl_message_icon](#) ()
Gets the [Fl_Box](#) icon container of the current default dialog used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).
- void [fl_message_icon_label](#) (const char *str)
Sets the icon label of the dialog window used in many common dialogs.

- void [fl_message_position](#) (const int x, const int y, const int center)
Sets the preferred position for the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).
- void [fl_message_position](#) (FL_Widget *widget)
Sets the preferred position for the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).
- int [fl_message_position](#) (int *x, int *y)
Gets the preferred position for the message box used in many common dialogs like [fl_message\(\)](#), [fl_alert\(\)](#), [fl_ask\(\)](#), [fl_choice\(\)](#), [fl_input\(\)](#), [fl_password\(\)](#).
- void [fl_message_title](#) (const char *title)
Sets the title of the dialog window used in many common dialogs.
- void [fl_message_title_default](#) (const char *title)
Sets the default title of the dialog window used in many common dialogs.
- const char * [fl_password](#) (const char *fmt, const char *defstr,...)
*Shows an input dialog displaying the *fmt* message with variable arguments.*
- const char * [fl_password](#) (int maxchar, const char *fmt, const char *defstr,...)
*Shows an input dialog displaying the *fmt* message with variable arguments.*

Variables

- const char * [fl_cancel](#) = "Cancel"
string pointer used in common dialogs, you can change it to another language
- const char * [fl_close](#) = "Close"
string pointer used in common dialogs, you can change it to another language
- [FL_Font](#) [fl_message_font](#) = [FL_HELVETICA](#)
- [FL_Fonsize](#) [fl_message_size](#) = -1
- const char * [fl_no](#) = "No"
string pointer used in common dialogs, you can change it to another language
- const char * [fl_ok](#) = "OK"
string pointer used in common dialogs, you can change it to another language
- const char * [fl_yes](#) = "Yes"
string pointer used in common dialogs, you can change it to another language

12.210.1 Detailed Description

Utility functions for common dialogs.

This file defines the functions

- [fl_alert\(\)](#)
- [fl_beep\(\)](#)
- [fl_message\(\)](#)
- [fl_ask\(\)](#)
- [fl_choice\(\)](#)
- [fl_input\(\)](#)
- [fl_input_str\(\)](#)
- [fl_password\(\)](#)
- [fl_password_str\(\)](#)

and some more functions to change their behavior (positioning, window title, and more). Since FLTK 1.4.0 a big part of these functions is implemented in class [FL_Message](#).

12.211 fl_boxtype.cxx File Reference

Drawing code for common box types.

```
#include <FL/Fl.H>
#include <FL/Fl_Widget.H>
#include <FL/fl_draw.H>
#include <config.h>
```

Macros

- `#define D1 BORDER_WIDTH`
- `#define D2 (BORDER_WIDTH+BORDER_WIDTH)`
- `#define fl_border_box fl_rectbound`

allow consistent naming

Functions

- void **fl_border_frame** (int x, int y, int w, int h, [Fl_Color](#) c)
Draws a frame of type FL_BORDER_FRAME.
- void **fl_down_box** (int x, int y, int w, int h, [Fl_Color](#) c)
Draws a box of type FL_DOWN_BOX.
- void **fl_down_frame** (int x, int y, int w, int h, [Fl_Color](#))
Draws a frame of type FL_DOWN_FRAME.
- void **fl_draw_box** ([Fl_Boxtype](#) t, int x, int y, int w, int h, [Fl_Color](#) c)
Draws a box using given type, position, size and color.
- void **fl_draw_box_focus** ([Fl_Boxtype](#) bt, int x, int y, int w, int h, [Fl_Color](#) fg, [Fl_Color](#) bg)
Draws the focus rectangle inside a box using given type, position, size and color.
- void **fl_embossed_box** (int x, int y, int w, int h, [Fl_Color](#) c)
Draws a box of type FL_EMBOSSED_BOX.
- void **fl_embossed_frame** (int x, int y, int w, int h, [Fl_Color](#))
Draws a frame of type FL_EMBOSSED_FRAME.
- void **fl_engraved_box** (int x, int y, int w, int h, [Fl_Color](#) c)
Draws a box of type FL_ENGRAVED_BOX.
- void **fl_engraved_frame** (int x, int y, int w, int h, [Fl_Color](#))
Draws a frame of type FL_ENGRAVED_FRAME.
- void **fl_flat_box** (int x, int y, int w, int h, [Fl_Color](#) c)
Draws a box of type FL_FLAT_BOX.
- void **fl_frame** (const char *s, int x, int y, int w, int h)
Draws a series of line segments around the given box.
- void **fl_frame2** (const char *s, int x, int y, int w, int h)
Draws a series of line segments around the given box.
- const [uchar](#) * **fl_gray_ramp** ()
- void **fl_internal_boxtype** ([Fl_Boxtype](#) t, [Fl_Box_Draw_F](#) *f, [Fl_Box_Draw_Focus_F](#) *ff)
Sets the drawing function for a given box type.
- void **fl_no_box** (int, int, int, int, [Fl_Color](#))
Draws a box of type FL_NO_BOX.
- void **fl_rectbound** (int x, int y, int w, int h, [Fl_Color](#) bgcolor)
Draws a bounded rectangle with a given position, size and color.
- void **fl_thin_down_box** (int x, int y, int w, int h, [Fl_Color](#) c)
Draws a box of type FL_THIN_DOWN_BOX.
- void **fl_thin_down_frame** (int x, int y, int w, int h, [Fl_Color](#))
Draws a frame of type FL_THIN_DOWN_FRAME.

- void **fl_thin_up_box** (int x, int y, int w, int h, [Fl_Color](#) c)
Draws a box of type FL_THIN_UP_BOX.
- void **fl_thin_up_frame** (int x, int y, int w, int h, [Fl_Color](#))
Draws a frame of type FL_THIN_UP_FRAME.
- void **fl_up_box** (int x, int y, int w, int h, [Fl_Color](#) c)
Draws a box of type FL_UP_BOX.
- void **fl_up_frame** (int x, int y, int w, int h, [Fl_Color](#))
Draws a frame of type FL_UP_FRAME.

12.211.1 Detailed Description

Drawing code for common box types.

12.211.2 Function Documentation

12.211.2.1 fl_internal_boxtype()

```
void fl_internal_boxtype (
    Fl\_Boxtype t,
    Fl\_Box\_Draw\_F * f,
    Fl\_Box\_Draw\_Focus\_F * ff)
```

Sets the drawing function for a given box type.

Parameters

in	<i>t</i>	box type
in	<i>f</i>	box drawing function
in	<i>ff</i>	optional box focus rectangle drawing function

12.211.2.2 fl_rectbound()

```
void fl_rectbound (
    int x,
    int y,
    int w,
    int h,
    Fl\_Color bgcolor)
```

Draws a bounded rectangle with a given position, size and color.
Equivalent to drawing a box of type FL_BORDER_BOX.

12.212 fl_cmap.h

```
00001 //
00002 // DO NOT EDIT THIS FILE !
00003 //
00004 // This file must be generated by "util/cmap.cxx".
00005 // See instructions in this file.
00006 //
00007 // Copyright 1998-2022 by Bill Spitzak and others.
00008 //
00009 // This library is free software. Distribution and use rights are outlined in
00010 // the file "COPYING" which should have been included with this file. If this
00011 // file is missing or damaged, see the license at:
00012 //
00013 //     https://www.fltk.org/COPYING.php
00014 //
00015 // Please see the following page on how to report bugs and issues:
00016 //
00017 //     https://www.fltk.org/bugs.php
00018 //
00019 // 0x00000000, // 0
00020 // 0xff000000, // 1
```

```
00021 0x00ff0000, // 2
00022 0xffff0000, // 3
00023 0x0000ff00, // 4
00024 0xff00ff00, // 5
00025 0x00ffff00, // 6
00026 0xffffffff00, // 7
00027 0x55555500, // 8
00028 0xc6717100, // 9
00029 0x71c67100, // 10
00030 0x8e8e3800, // 11
00031 0x7171c600, // 12
00032 0x8e388e00, // 13
00033 0x388e8e00, // 14
00034 0x00008000, // 15
00035 0xa8a89800, // 16
00036 0xe8e8d800, // 17
00037 0x68685800, // 18
00038 0x98a8a800, // 19
00039 0xd8e8e800, // 20
00040 0x58686800, // 21
00041 0x9c9ca800, // 22
00042 0xdcdce800, // 23
00043 0x5c5c6800, // 24
00044 0x9ca89c00, // 25
00045 0xdce8dc00, // 26
00046 0x5c685c00, // 27
00047 0x90909000, // 28
00048 0xc0c0c000, // 29
00049 0x50505000, // 30
00050 0xa0a0a000, // 31
00051 0x00000000, // 32
00052 0x0d0d0d00, // 33
00053 0x1a1a1a00, // 34
00054 0x26262600, // 35
00055 0x31313100, // 36
00056 0x3d3d3d00, // 37
00057 0x48484800, // 38
00058 0x55555500, // 39
00059 0x5f5f5f00, // 40
00060 0x6a6a6a00, // 41
00061 0x75757500, // 42
00062 0x80808000, // 43
00063 0x8a8a8a00, // 44
00064 0x95959500, // 45
00065 0xa0a0a000, // 46
00066 0xaaaaaa00, // 47
00067 0xb5b5b500, // 48
00068 0xc0c0c000, // 49
00069 0xcbcbcb00, // 50
00070 0xd5d5d500, // 51
00071 0xe0e0e000, // 52
00072 0xeaeaea00, // 53
00073 0xf5f5f500, // 54
00074 0xffffffff00, // 55
00075 0x00000000, // 56
00076 0x00240000, // 57
00077 0x00480000, // 58
00078 0x006d0000, // 59
00079 0x00910000, // 60
00080 0x00b60000, // 61
00081 0x00da0000, // 62
00082 0x00ff0000, // 63
00083 0x3f000000, // 64
00084 0x3f240000, // 65
00085 0x3f480000, // 66
00086 0x3f6d0000, // 67
00087 0x3f910000, // 68
00088 0x3fb60000, // 69
00089 0x3fda0000, // 70
00090 0x3fff0000, // 71
00091 0x7f000000, // 72
00092 0x7f240000, // 73
00093 0x7f480000, // 74
00094 0x7f6d0000, // 75
00095 0x7f910000, // 76
00096 0x7fb60000, // 77
00097 0x7fda0000, // 78
00098 0x7fff0000, // 79
00099 0xbf000000, // 80
00100 0xbf240000, // 81
00101 0xbf480000, // 82
00102 0xbf6d0000, // 83
00103 0xbf910000, // 84
00104 0xbfb60000, // 85
00105 0xbfda0000, // 86
00106 0xbfff0000, // 87
00107 0xff000000, // 88
```

```
00108      0xff240000, // 89
00109      0xff480000, // 90
00110      0xff6d0000, // 91
00111      0xff910000, // 92
00112      0xffb60000, // 93
00113      0xffda0000, // 94
00114      0xffff0000, // 95
00115      0x00003f00, // 96
00116      0x00243f00, // 97
00117      0x00483f00, // 98
00118      0x006d3f00, // 99
00119      0x00913f00, // 100
00120      0x00b63f00, // 101
00121      0x00da3f00, // 102
00122      0x00ff3f00, // 103
00123      0x3f003f00, // 104
00124      0x3f243f00, // 105
00125      0x3f483f00, // 106
00126      0x3f6d3f00, // 107
00127      0x3f913f00, // 108
00128      0x3fb63f00, // 109
00129      0x3fda3f00, // 110
00130      0x3fff3f00, // 111
00131      0x7f003f00, // 112
00132      0x7f243f00, // 113
00133      0x7f483f00, // 114
00134      0x7f6d3f00, // 115
00135      0x7f913f00, // 116
00136      0x7fb63f00, // 117
00137      0x7fda3f00, // 118
00138      0x7fff3f00, // 119
00139      0xbf003f00, // 120
00140      0xbf243f00, // 121
00141      0xbf483f00, // 122
00142      0xbf6d3f00, // 123
00143      0xbf913f00, // 124
00144      0xbfb63f00, // 125
00145      0xbfda3f00, // 126
00146      0xbfff3f00, // 127
00147      0xff003f00, // 128
00148      0xff243f00, // 129
00149      0xff483f00, // 130
00150      0xff6d3f00, // 131
00151      0xff913f00, // 132
00152      0xffb63f00, // 133
00153      0xffda3f00, // 134
00154      0xffff3f00, // 135
00155      0x00007f00, // 136
00156      0x00247f00, // 137
00157      0x00487f00, // 138
00158      0x006d7f00, // 139
00159      0x00917f00, // 140
00160      0x00b67f00, // 141
00161      0x00da7f00, // 142
00162      0x00ff7f00, // 143
00163      0x3f007f00, // 144
00164      0x3f247f00, // 145
00165      0x3f487f00, // 146
00166      0x3f6d7f00, // 147
00167      0x3f917f00, // 148
00168      0x3fb67f00, // 149
00169      0x3fda7f00, // 150
00170      0x3fff7f00, // 151
00171      0x7f007f00, // 152
00172      0x7f247f00, // 153
00173      0x7f487f00, // 154
00174      0x7f6d7f00, // 155
00175      0x7f917f00, // 156
00176      0x7fb67f00, // 157
00177      0x7fda7f00, // 158
00178      0x7fff7f00, // 159
00179      0xbf007f00, // 160
00180      0xbf247f00, // 161
00181      0xbf487f00, // 162
00182      0xbf6d7f00, // 163
00183      0xbf917f00, // 164
00184      0xbfb67f00, // 165
00185      0xbfda7f00, // 166
00186      0xbfff7f00, // 167
00187      0xff007f00, // 168
00188      0xff247f00, // 169
00189      0xff487f00, // 170
00190      0xff6d7f00, // 171
00191      0xff917f00, // 172
00192      0xffb67f00, // 173
00193      0xffda7f00, // 174
00194      0xffff7f00, // 175
```

```
00195     0x0000bf00, // 176
00196     0x0024bf00, // 177
00197     0x0048bf00, // 178
00198     0x006dbf00, // 179
00199     0x0091bf00, // 180
00200     0x00b6bf00, // 181
00201     0x00dabf00, // 182
00202     0x00ffbf00, // 183
00203     0x3f00bf00, // 184
00204     0x3f24bf00, // 185
00205     0x3f48bf00, // 186
00206     0x3f6dbf00, // 187
00207     0x3f91bf00, // 188
00208     0x3fb6bf00, // 189
00209     0x3fdabf00, // 190
00210     0x3fffbf00, // 191
00211     0x7f00bf00, // 192
00212     0x7f24bf00, // 193
00213     0x7f48bf00, // 194
00214     0x7f6dbf00, // 195
00215     0x7f91bf00, // 196
00216     0x7fb6bf00, // 197
00217     0x7fdabf00, // 198
00218     0x7fffbf00, // 199
00219     0xbf00bf00, // 200
00220     0xbf24bf00, // 201
00221     0xbf48bf00, // 202
00222     0xbf6dbf00, // 203
00223     0xbf91bf00, // 204
00224     0xbfb6bf00, // 205
00225     0xbfdabf00, // 206
00226     0xbfffbf00, // 207
00227     0xff00bf00, // 208
00228     0xff24bf00, // 209
00229     0xff48bf00, // 210
00230     0xff6dbf00, // 211
00231     0xff91bf00, // 212
00232     0xffb6bf00, // 213
00233     0xffdabf00, // 214
00234     0xffffbf00, // 215
00235     0x0000ff00, // 216
00236     0x0024ff00, // 217
00237     0x0048ff00, // 218
00238     0x006dff00, // 219
00239     0x0091ff00, // 220
00240     0x00b6ff00, // 221
00241     0x00daff00, // 222
00242     0x00ffff00, // 223
00243     0x3f00ff00, // 224
00244     0x3f24ff00, // 225
00245     0x3f48ff00, // 226
00246     0x3f6dff00, // 227
00247     0x3f91ff00, // 228
00248     0x3fb6ff00, // 229
00249     0x3fdaff00, // 230
00250     0x3fffff00, // 231
00251     0x7f00ff00, // 232
00252     0x7f24ff00, // 233
00253     0x7f48ff00, // 234
00254     0x7f6dff00, // 235
00255     0x7f91ff00, // 236
00256     0x7fb6ff00, // 237
00257     0x7fdaff00, // 238
00258     0x7fffff00, // 239
00259     0xbf00ff00, // 240
00260     0xbf24ff00, // 241
00261     0xbf48ff00, // 242
00262     0xbf6dff00, // 243
00263     0xbf91ff00, // 244
00264     0xbfb6ff00, // 245
00265     0xbfdaff00, // 246
00266     0xbfffff00, // 247
00267     0xff00ff00, // 248
00268     0xff24ff00, // 249
00269     0xff48ff00, // 250
00270     0xff6dff00, // 251
00271     0xff91ff00, // 252
00272     0xffb6ff00, // 253
00273     0xffdaff00, // 254
00274     0xffffff00 // 255
00275 //
00276 // End of fl_cmap.h - generated by cmap.cxx
00277 //
```


12.213 fl_color.cxx File Reference

Color handling.

```
#include <FL/Fl.H>
#include <FL/Fl_Device.H>
#include <FL/Fl_Graphics_Driver.H>
#include "fl_cmap.h"
```

Functions

- [Fl_Color fl_color_average](#) ([Fl_Color](#) color1, [Fl_Color](#) color2, float weight)
Returns the weighted average color between the two given colors.
- [Fl_Color fl_inactive](#) ([Fl_Color](#) c)
Returns the inactive, dimmed version of the given color.

Variables

- unsigned [fl_cmap](#) [256]

12.213.1 Detailed Description

Color handling.

12.214 Fl_compose.cxx File Reference

Utility functions to support text input.

```
#include <FL/Fl.H>
#include "Fl_Screen_Driver.H"
```

12.214.1 Detailed Description

Utility functions to support text input.

12.215 fl_contrast.cxx File Reference

Color contrast handling.

```
#include <FL/Fl.H>
#include <math.h>
```

Macros

- `#define DEBUG\_CONTRAST\_LEGACY 0`

Functions

- [Fl_Color fl_contrast](#) ([Fl_Color](#) fg, [Fl_Color](#) bg, int context, int size)
Returns a color that contrasts with the background color.
- void [fl_contrast_function](#) ([Fl_Contrast_Function](#) *f)
Register a custom contrast function.
- int [fl_contrast_level](#) ()
Get the contrast level (sensitivity) of the [fl_contrast\(\)](#) method.
- void [fl_contrast_level](#) (int level)
Set the contrast level (sensitivity) of the [fl_contrast\(\)](#) method.

- `int fl_contrast_mode ()`
Return the current contrast algorithm (mode).
- `void fl_contrast_mode (int mode)`
Set the contrast algorithm (mode).
- `double fl_lightness (FL_Color color)`
Return the perceived lightness of a color.
- `double fl_luminance (FL_Color color)`
Return the raw / physical luminance of a color.
- `unsigned get_color (FL_Color i)`

Variables

- `unsigned fl_cmap [256]`

12.215.1 Detailed Description

Color contrast handling.

Implementation of `fl_contrast()` and its variants.

12.216 fl_curve.cxx File Reference

Utility for drawing Bézier curves, adding the points to the current `fl_begin/fl_vertex/fl_end` path.

```
#include <FL/fl_draw.H>
#include <math.h>
```

12.216.1 Detailed Description

Utility for drawing Bézier curves, adding the points to the current `fl_begin/fl_vertex/fl_end` path.

Incremental math implementation: I very much doubt this is optimal! From Foley/vanDam page 511. If anybody has a better algorithm, please send it!

12.217 Fl_Double_Window.cxx File Reference

`Fl_Double_Window` implementation.

```
#include <FL/Fl.H>
#include <FL/platform.H>
#include <FL/Fl_Double_Window.H>
#include <FL/fl_draw.H>
#include "Fl_Window_Driver.H"
```

12.217.1 Detailed Description

`Fl_Double_Window` implementation.

12.218 Fl_get_system_colors.cxx File Reference

System color support.

```
#include <FL/Fl.H>
#include "Fl_Screen_Driver.H"
#include "Fl_System_Driver.H"
#include <FL/fl_draw.H>
#include <FL/platform.H>
#include <FL/math.h>
```

```
#include <FL/fl_utf8.h>
#include <FL/fl_string_functions.h>
#include "flstring.h"
#include <stdio.h>
#include <stdlib.h>
#include <FL/Fl_Pixmap.H>
#include <FL/Fl_Tiled_Image.H>
#include "tile.xpm"
```

Macros

- `#define D1 BORDER_WIDTH`
- `#define D2 (BORDER_WIDTH+BORDER_WIDTH)`

Functions

- void **fl_down_box** (int, int, int, int, [Fl_Color](#))
Draws a box of type FL_DOWN_BOX.
- void **fl_down_frame** (int, int, int, int, [Fl_Color](#))
Draws a frame of type FL_DOWN_FRAME.
- int **fl_parse_color** (const char *p, [uchar](#) &r, [uchar](#) &g, [uchar](#) &b)
Parse a string containing a description of a color and write r, g, and b.
- void **fl_round_down_box** (int, int, int, int, [Fl_Color](#))
- void **fl_round_focus** ([Fl_Boxtype](#), int, int, int, int, [Fl_Color](#), [Fl_Color](#))
- void **fl_round_up_box** (int, int, int, int, [Fl_Color](#))
- void **fl_thin_down_box** (int, int, int, int, [Fl_Color](#))
Draws a box of type FL_THIN_DOWN_BOX.
- void **fl_thin_down_frame** (int, int, int, int, [Fl_Color](#))
Draws a frame of type FL_THIN_DOWN_FRAME.
- void **fl_thin_up_box** (int, int, int, int, [Fl_Color](#))
Draws a box of type FL_THIN_UP_BOX.
- void **fl_thin_up_frame** (int, int, int, int, [Fl_Color](#))
Draws a frame of type FL_THIN_UP_FRAME.
- void **fl_up_box** (int, int, int, int, [Fl_Color](#))
Draws a box of type FL_UP_BOX.
- void **fl_up_frame** (int, int, int, int, [Fl_Color](#))
Draws a frame of type FL_UP_FRAME.

Variables

- const char * **fl_bg** = NULL
- const char * **fl_bg2** = NULL
- const char * **fl_fg** = NULL

12.218.1 Detailed Description

System color support.

12.218.2 Function Documentation

12.218.2.1 fl_parse_color()

```
int fl_parse_color (
    const char * p,
    uchar & r,
    uchar & g,
    uchar & b)
```

Parse a string containing a description of a color and write r, g, and b.

This call is used by the Pixmap file format interpreter and by the command line arguments parser to set UI colors. RGB color triplets usually start with a '#' character, but it can be omitted if it does not conflict with the later rules. Color components are defined in hexadecimal notation with 1, 2, 3, or four hex digits per component, making color triplets 3, 6, 9, or 12 characters long. The interpreter is case insensitive. Valid code examples include "FF0000" for red, "#0F0" for green, and "000000004444" for a dark blue.

On the X11 platform, color values can also be given a color name like "red". The list of accepted color names is provided by the X11 server.

If none of the color interpretations work, `fl_parse_color` returns 0. The Pixmap reader interprets those as transparent, and are usually written as "None", "#transparent", or "bg".

Parameters

in	<i>p</i>	a C-string describing the color
out	<i>r,g,b</i>	the color components in the 0...255 range

Returns

0 if the color cannot be interpreted, 1 otherwise

12.219 Fl_Gl_Choice.H

```
00001 //
00002 // OpenGL definitions for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2018 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016 //
00017 // Internal interface to set up OpenGL.
00018 //
00019 // A "Fl_Gl_Choice" is created from an OpenGL mode and holds information
00020 // necessary to create a window (on X) and to create an OpenGL "context"
00021 // (on both X and Win32).
00022 //
00023 // create_gl_context takes a window (necessary only on Win32) and an
00024 // Fl_Gl_Choice and returns a new OpenGL context. All contexts share
00025 // display lists with each other.
00026 //
00027 // On X another create_gl_context is provided to create it for any
00028 // X visual.
00029 //
00030 // set_gl_context makes the given OpenGL context current and makes
00031 // it draw into the passed window. It tracks the current one context
00032 // to avoid calling the context switching code when the same context
00033 // is used, though it is a mystery to me why the GLX/WGL libraries
00034 // don't do this themselves...
00035 //
00036 // delete_gl_context destroys the context.
00037 //
00038 // This code is used by Fl_Gl_Window, gl_start(), and gl_visual()
00039 //
00040 #ifndef Fl_Gl_Choice_H
00041 #define Fl_Gl_Choice_H
```

```

00042
00043 // Describes the platform-independent part of data needed to create a GLContext.
00044 class Fl_Gl_Choice {
00045     friend class Fl_Gl_Window_Driver;
00046     int mode_;
00047     const int *alist;
00048     Fl_Gl_Choice *next;
00049 public:
00050     Fl_Gl_Choice(int m, const int *alistp, Fl_Gl_Choice *n) : mode(m), alist(alistp), next(n) {}
00051 };
00052
00053 #endif // Fl_Gl_Choice_H

```

12.220 Fl_Gl_Window_Driver.H

```

00001 //
00002 // Definition of class Fl_Gl_Window_Driver, and of its platform-specific derived classes
00003 // for the Fast Light Tool Kit (FLTK).
00004 //
00005 // Copyright 2016-2018 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00024 #ifndef Fl_Gl_Window_Driver_H
00025 #define Fl_Gl_Window_Driver_H
00026
00027 #include <FL/Fl_Gl_Window.H>
00028 #include <FL/gl.h> // for GLint
00029
00030 class Fl_Gl_Choice;
00031 class Fl_Font_Descriptor;
00032
00033 /* The constructor of each Fl_Gl_Window object creates also an object from a
00034 platform-specific derived class from this class.
00035 */
00036 class Fl_Gl_Window_Driver {
00037 protected:
00038     GLint current_prog;
00039     Fl_Gl_Window *pWindow;
00040 public:
00041     static Fl_Window* cached_window;
00042     static int nContext;
00043     static GLContext *context_list;
00044     static Fl_Gl_Choice *first;
00045     static int copy;
00046     static float gl_scale;
00047     static GLContext gl_start_context;
00048     Fl_Gl_Choice* g() {return pWindow->g;}
00049     void g(Fl_Gl_Choice *c) {pWindow->g = c;}
00050     int mode() {return pWindow->mode_;}
00051     void mode(int m) { pWindow->mode_ = m;}
00052     const int *alist() {return pWindow->alist;}
00053     void alist(const int *l) { pWindow->alist = l;}
00054     void* overlay() {return pWindow->overlay;}
00055     void draw_overlay() {pWindow->draw_overlay();}
00056
00057     Fl_Gl_Window_Driver(Fl_Gl_Window *win) : pWindow(win) {current_prog=0;}
00058     virtual ~Fl_Gl_Window_Driver() {}
00059     static Fl_Gl_Window_Driver *newGlWindowDriver(Fl_Gl_Window *w);
00060     static Fl_Gl_Window_Driver *global();
00061     virtual float pixels_per_unit() {return 1;}
00062     virtual void before_show(int&) {}
00063     virtual void after_show() {}
00064     virtual void invalidate();
00065     virtual int mode_(int /*m*/, const int * /*a*/) {return 0;}
00066     virtual void make_current_before() {}
00067     virtual void make_current_after() {}
00068     virtual void swap_buffers() {}
00069     virtual void resize(int /*is_a_resize*/, int /*w*/, int /*h*/) {}
00070     virtual char swap_type();
00071     virtual void swap_interval(int) {}
00072     virtual int swap_interval() const { return -1; }
00073     virtual int flush_begin(char&) {return 0;}
00074     virtual void gl_hide_before(void *&) {} // the default implementation may be enough
00075     static Fl_Gl_Choice *find_begin(int m, const int *alistp);
00076     static void add_context(GLContext ctx);

```

```

00077     static void del_context(GLContext ctx);
00078     // Return one of these structures for a given gl mode.
00079     // The second argument is a glX attribute list, and is used if mode is zero.
00080     // This is not supported on Win32:
00081     virtual Fl_Gl_Choice *find(int /*mode*/, const int * /*alistp*/) {return NULL;}
00082     virtual GLContext create_gl_context(Fl_Window*, const Fl_Gl_Choice*) {return 0;}
00083     virtual void set_gl_context(Fl_Window*, GLContext) {}
00084     virtual void delete_gl_context(GLContext) {}
00085     virtual void make_overlay(void* &o);
00086     virtual void hide_overlay() {} // the default implementation may be enough
00087     virtual void make_overlay_current() {}
00088     virtual void redraw_overlay() {}
00089     virtual int can_do_overlay() {return 0;}
00090     virtual void waitGL() {} // support for gl_finish() function
00091     virtual void gl_visual(Fl_Gl_Choice*); // support for Fl::gl_visual() function
00092     virtual void gl_start() {} // support for gl_start() function
00093     virtual void* GetProcAddress(const char *procName); // support for glutGetProcAddress()
00094     virtual void draw_string_legacy(const char* str, int n); // support for gl_draw()
00095     void draw_string_legacy_get_list(const char* str, int n); // support for gl_draw()
00096     static void draw_string_legacy_glut(const char* str, int n); // support for gl_draw()
00097     virtual void get_list(Fl_Font_Descriptor*, int) {} // support for gl_draw() without textures
00098     virtual void gl_bitmap_font(Fl_Font_Descriptor *) {} // support for gl_font() without textures
00099     virtual int overlay_color(Fl_Color) {return 0;} // support for gl_color() with HAVE_GL_OVERLAY
00100     static void draw_string_with_texture(const char* str, int n); // cross-platform
00101     // support for gl_draw(). The cross-platform version may be enough.
00102     virtual char *alpha_mask_for_string(const char *str, int n, int w, int h, Fl_Fonsize fs);
00103     virtual int genlistsize() { return 0; } // support for gl_draw()
00104     virtual Fl_Font_Descriptor** fontnum_to_fontdescriptor(int fnum);
00105     virtual Fl_RGB_Image* capture_gl_rectangle(int x, int y, int w, int h);
00106     static inline Fl_Gl_Window_Driver* driver(const Fl_Gl_Window *win) {return win->pGlWindowDriver;}
00107     // true means the platform uses glScissor() to make sure GL subwindows
00108     // don't leak outside their parent window
00109     virtual bool need_scissor() { return false; }
00110     virtual void switch_to_GL1();
00111     virtual void switch_back();
00112 };
00113
00114 #endif /* Fl_Gl_Window_Driver_H */
00115

```

12.221 Fl_Graphics_Driver.cxx File Reference

Implementation of class Fl_Graphics_Driver.

```

#include <config.h>
#include <FL/Fl_Graphics_Driver.H>

```

Variables

- Fl_Graphics_Driver * **fl_graphics_driver**

Points to the driver that currently receives all graphics requests.

12.221.1 Detailed Description

Implementation of class Fl_Graphics_Driver.

12.222 Fl_Grid.cxx File Reference

Implements the [Fl_Grid](#) container widget.

```

#include <FL/Fl_Grid.H>
#include <FL/fl_draw.H>

```

12.222.1 Detailed Description

Implements the [Fl_Grid](#) container widget.

Since

1.4.0

12.223 Fl_Image_Reader.h

```

00001 //
00002 // Internal (Image) Reader class for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2020-2021 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 /*
00018  This internal (undocumented) class reads data chunks from a file or from
00019  memory in LSB-first byte order.
00020
00021  This class is used in Fl_GIF_Image and Fl_BMP_Image to avoid code
00022  duplication and may be extended to be used in similar cases. Future
00023  options might be to read data in MSB-first byte order or to add more
00024  methods.
00025 */
00026
00027 #ifndef FL_IMAGE_READER_H
00028 #define FL_IMAGE_READER_H
00029
00030 #include <stdio.h>
00031
00032 class Fl_Image_Reader {
00033 public:
00034     // Create the reader.
00035     Fl_Image_Reader()
00036         : is_file_(0)
00037         , is_data_(0)
00038         , file_(0L)
00039         , data_(0L)
00040         , start_(0L)
00041         , end_((const unsigned char *) (-1L))
00042         , name_(0L)
00043         , error_(0) {}
00044
00045     // Initialize the reader to access the file system, filename is copied
00046     // and stored.
00047     int open(const char *filename);
00048
00049     // Initialize the reader for memory access, name is copied and stored
00050     int open(const char *imagename, const unsigned char *data, const size_t datasize);
00051     // Deprecated, DO NOT USE!
00052     int open(const char *imagename, const unsigned char *data);
00053
00054     // Close and destroy the reader
00055     ~Fl_Image_Reader();
00056
00057     // Read a single byte from memory or a file
00058     unsigned char read_byte();
00059
00060     // Read a 16-bit unsigned integer, LSB-first
00061     unsigned short read_word();
00062
00063     // Read a 32-bit unsigned integer, LSB-first
00064     unsigned int read_dword();
00065
00066     // Read a 32-bit signed integer, LSB-first
00067     int read_long() { return (int)read_dword(); }
00068
00069     // Move the current read position to a byte offset from the beginning
00070     // of the file or the original start address in memory
00071     void seek(unsigned int n);
00072
00073     // Get the current file or memory offset from the beginning
00074     // of the file or the original start address in memory
00075     long tell() const;
00076
00077     // Get the current EOF or error status of the file or data block
00078     int error() const { return error_; }
00079

```

```

00080 // return the name or filename for this reader
00081 const char *name() const { return name_; }
00082
00083 // skip a given number of bytes
00084 void skip(unsigned int n) { seek((unsigned int)tell() + n); }
00085
00086 private:
00087 // open() sets this if we read from a file
00088 char is_file_;
00089 // open() sets this if we read from memory
00090 char is_data_;
00091 // a pointer to the opened file
00092 FILE *file_;
00093 // a pointer to the current byte in memory
00094 const unsigned char *data_;
00095 // a pointer to the start of the image data
00096 const unsigned char *start_;
00097 // a pointer to the end of image data if reading from memory, otherwise undefined
00098 // note: currently (const unsigned char *)(-1L) if end of memory is not available
00099 // ... which means "unlimited"
00100 const unsigned char *end_;
00101 // a copy of the name associated with this reader
00102 char *name_;
00103 // a flag to store EOF or error status
00104 int error_;
00105 };
00106
00107 #endif // FL_IMAGE_READER_H

```

12.224 Fl_Int_Vector.H

```

00001 //
00002 // An STL-ish vector without templates for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2002 by Greg Ercolano.
00005 // Copyright 2022-2023 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00018 #ifndef Fl_Int_Vector_H
00019 #define Fl_Int_Vector_H
00020
00021 #include <FL/Fl_Export.H>
00022
00023 class FL_EXPORT Fl_Int_Vector {
00024     int *arr_;
00025     unsigned int size_;
00026
00027     void init() {
00028         arr_ = 0;
00029         size_ = 0;
00030     }
00031     void copy(int *newarr, unsigned int newsize);
00032
00033 public:
00034     Fl_Int_Vector() {
00035         init();
00036     }
00037
00038     ~Fl_Int_Vector();
00039
00040     Fl_Int_Vector(Fl_Int_Vector &o) {
00041         init();
00042         copy(o.arr_, o.size_);
00043     }
00044
00045     Fl_Int_Vector &operator=(Fl_Int_Vector &o) {
00046         init();
00047         copy(o.arr_, o.size_);
00048         return *this;
00049     }
00050
00051     int operator[](int x) const {
00052         return arr_[x];
00053     }
00054 }
00055
00056 int operator[](int x) const {
00057     return arr_[x];
00058 }
00059
00060
00061
00062
00063
00064
00065
00066
00067
00068
00069
00070
00071
00072
00073
00074
00075
00076
00077
00078
00079
00080
00081
00082
00083
00084
00085
00086
00087
00088
00089
00090
00091
00092
00093
00094
00095
00096
00097
00098
00099
00100
00101
00102
00103
00104
00105
00106
00107
00108
00109
00110
00111
00112
00113
00114
00115
00116
00117

```



```

00126     int &operator[](int x) {
00127         return arr_[x];
00128     }
00129
00131     unsigned int size() const {
00132         return size_;
00133     }
00134
00135     void size(unsigned int count);
00136
00143     int pop_back() {
00144         int tmp = arr_[size_ - 1];
00145         size--;
00146         return tmp;
00147     }
00148
00150     void push_back(int val) {
00151         unsigned int x = size_;
00152         size(size_ + 1);
00153         arr_[x] = val;
00154     }
00155
00161     int back() const {
00162         return arr_[size_ - 1];
00163     }
00164
00169     bool empty() const {
00170         return (size_ == 0) ? true : false;
00171     }
00172 };
00173
00179 #endif // Fl_Int_Vector_H

```

12.225 Fl_Message.h

```

00001 //
00002 // Common dialog header file for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef _src_Fl_Message_h_
00018 #define _src_Fl_Message_h_
00019
00020 #include <FL/Fl_Window.H>
00021 #include <FL/Fl_Box.H>
00022 #include <FL/fl_ask.H>
00023
00024 class Fl_Button;
00025 class Fl_Input;
00026
00042 /* Note: Do not FL_EXPORT this class, it's for internal use only */
00043
00044 class Fl_Message_Box : public Fl_Box {
00045 public:
00046     Fl_Message_Box(int X, int Y, int W, int H)
00047         : Fl_Box(X, Y, W, H) {}
00048     int handle(int e) FL_OVERRIDE;
00049 }; // class Fl_Message_Box
00050
00072 /* Note: Do not FL_EXPORT this class, it's for internal use only */
00073
00074 class Fl_Message {
00075
00076     // static variables and methods
00077
00078 private:
00079     static Fl_Box *message_icon_; // returned by Fl_Message::message_icon()
00080
00081     static const char *message_title_;
00082     static const char *message_title_default_;
00083
00084     // icon label for next dialog (STR #2762)
00085     static const char *message_icon_label_;
00086

```

```

00087 // Note: since Fl_Message objects are destroyed before fl_input()
00088 // and fl_password() return their input text, we *need* to store
00089 // the text in an internal (static) buffer. :-(
00090
00091 static char *input_buffer_; // points to the allocated text buffer
00092 static int input_size_;     // size of allocated text buffer
00093
00094 // the callback for all buttons:
00095 static void button_cb_(Fl_Widget *w, void *d);
00096
00097 // the window callback:
00098 static void window_cb_(Fl_Widget *w, void *d);
00099
00100 // resize to make text and buttons fit
00101 void resizeform();
00102
00103 public:
00104 static Fl_Box *message_icon();
00105 static void message_title(const char *title);
00106 static void message_title_default(const char *title);
00107 static void icon_label(const char *str);
00108
00109 static void message_position(const int x, const int y, const int center) {
00110     form_x_ = x;
00111     form_y_ = y;
00112     form_position_ = center ? 2 : 1;
00113 }
00114
00115 static void message_position(Fl_Widget *widget) {
00116     int xo, yo;
00117     Fl_Window *win = widget->top_window_offset(xo, yo);
00118     form_x_ = xo + widget->w() / 2;
00119     form_y_ = yo + widget->h() / 2;
00120     if (win) {
00121         form_x_ += win->x();
00122         form_y_ += win->y();
00123     }
00124     form_position_ = 2;
00125 }
00126
00127 static int message_position(int *x, int *y) {
00128     if (x)
00129         *x = form_position_ ? form_x_ : -1;
00130     if (y)
00131         *y = form_position_ ? form_y_ : -1;
00132     return form_position_;
00133 }
00134
00135 static void message_hotspot(int enable) { enable_hotspot_ = enable ? 1 : 0; }
00136
00137 static int message_hotspot() { return enable_hotspot_; }
00138
00139 int window_closed() const {
00140     return window_closed_;
00141 }
00142
00143 // member variables and methods
00144
00145 private:
00146     Fl_Window *window_;
00147     Fl_Message_Box *message_;
00148     Fl_Box *icon_;
00149     Fl_Button *button_[3];
00150     Fl_Input *input_;
00151     int retval_;
00152     int window_closed_;
00153
00154 // static (private) variables
00155
00156 static int enable_hotspot_;
00157 static int form_x_;
00158 static int form_y_;
00159 static int form_position_;
00160
00161 public:
00162 // Constructor
00163 Fl_Message(const char *iconlabel);
00164 ~Fl_Message() { delete window_; }
00165
00166 int innards(const char *fmt, va_list ap, const char *b0, const char *b1, const char *b2);
00167
00168 const char *input_innards(const char *fmt, va_list ap, const char *defstr, uchar type, int maxchar =
00169 -1, bool str = false);
00170 };
00171
00172 #endif // _src_Fl_Message_h_

```

12.226 Fl_Native_File_Chooser_Kdialog.H

```

00001 //
00002 // FLTK native file chooser widget : KDE version
00003 //
00004 // Copyright 2021-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef FL_KDIALOG_NATIVE_FILE_CHOOSER_H
00018 #define FL_KDIALOG_NATIVE_FILE_CHOOSER_H 1
00019
00020 #include <FL/Fl_Native_File_Chooser.H>
00021 #include <string>
00022
00023 class Fl_Kdialog_Native_File_Chooser_Driver : public Fl_Native_File_Chooser_FLTK_Driver {
00024     friend class Fl_Native_File_Chooser;
00025     friend class Fl_Zenity_Native_File_Chooser_Driver;
00026     struct fnfc_pipe_struct {
00027         char *all_files;
00028         int fd;
00029     };
00030     static void fnfc_fd_cb(int fd, fnfc_pipe_struct *data);
00031     char **_pathnames;
00032     int _tpathnames;
00033     char *_directory;
00034     char *_preset_file;
00035     char *_title;
00036     static bool did_find_kdialog;
00037     static bool have_looked_for_kdialog;
00038     Fl_Kdialog_Native_File_Chooser_Driver(int val);
00039     ~Fl_Kdialog_Native_File_Chooser_Driver();
00040     int count() const FL_OVERRIDE;
00041     const char *filename() const FL_OVERRIDE;
00042     const char *filename(int i) const FL_OVERRIDE;
00043     virtual void build_command(std::string& command);
00044     int show() FL_OVERRIDE;
00045     char *parse_filter(const char *f);
00046     const char *filter() const FL_OVERRIDE;
00047     void filter(const char *f) FL_OVERRIDE;
00048     int filters() const FL_OVERRIDE;
00049     void preset_file(const char *val) FL_OVERRIDE;
00050     const char *preset_file() const FL_OVERRIDE;
00051     void directory(const char *val) FL_OVERRIDE;
00052     const char *directory() const FL_OVERRIDE;
00053     void title(const char *val) FL_OVERRIDE;
00054     const char *title() const FL_OVERRIDE;
00055     void shell_quote(std::string& s);
00056 };
00057
00058 #endif // FL_KDIALOG_NATIVE_FILE_CHOOSER_H

```

12.227 Fl_Native_File_Chooser_Zenity.H

```

00001 //
00002 // FLTK native file chooser widget : Zenity version
00003 //
00004 // Copyright 2021-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file.  If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef FL_ZENITY_NATIVE_FILE_CHOOSER_H
00018 #define FL_ZENITY_NATIVE_FILE_CHOOSER_H 1
00019
00020 #include "Fl_Native_File_Chooser_Kdialog.H"
00021
00022

```

```

00028 class Fl_Zenity_Native_File_Chooser_Driver : public Fl_Kdialog_Native_File_Chooser_Driver {
00029     friend class Fl_Native_File_Chooser;
00030     static bool did_find_zenity;
00031     static bool have_looked_for_zenity;
00032     Fl_Zenity_Native_File_Chooser_Driver(int val);
00033     void append_filter(std::string& command);
00034     void build_command(std::string& command) FL_OVERRIDE;
00035 };
00036
00042 #endif // FL_ZENITY_NATIVE_FILE_CHOOSER_H

```

12.228 fl_oxy.h

```

00001 //
00002 // "Oxy" Scheme drawing routines for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2011 by Dmitrij K. aka "kdiman"
00005 // Copyright 2012-2022 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 // https://www.fltk.org/COPYING.php
00012 //
00013 // Please report all bugs and problems on the following page:
00014 //
00015 // https://www.fltk.org/str.php
00016 //
00017
00018 #ifndef fl_oxy_h
00019 #define fl_oxy_h
00020
00021 #include <FL/Fl.H>
00022
00023 // draw an arrow GUI element for the 'oxy' scheme
00024 //
00025 // bb    bounding box
00026 // t     arrow type
00027 // o     orientation
00028 // c     arrow color
00029
00030 extern FL_EXPORT void oxy_arrow(Fl_Rect bb,
00031                                Fl_Arrow_Type t, Fl_Orientation o,
00032                                Fl_Color col);
00033
00034 #endif // fl_oxy_h

```

12.229 Fl_Paged_Device.cxx File Reference

implementation of class [Fl_Paged_Device](#).

```

#include <FL/Fl_Paged_Device.H>
#include <FL/Fl.H>
#include <FL/fl_draw.H>

```

12.229.1 Detailed Description

implementation of class [Fl_Paged_Device](#).

12.230 fl_rect.cxx File Reference

Drawing and clipping routines for rectangles.

```

#include <FL/platform.H>
#include <FL/Fl_Graphics_Driver.H>

```

12.230.1 Detailed Description

Drawing and clipping routines for rectangles.

12.231 Fl_Screen_Driver.H

```

00001 //
00002 // All screen related calls in a driver style class.
00003 //
00004 // Copyright 1998-2024 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00023 #ifndef FL_SCREEN_DRIVER_H
00024 #define FL_SCREEN_DRIVER_H
00025
00026 #include <FL/fl_types.h>
00027 #include <FL/Fl.H> // for Fl_Timeout_Handler
00028 #include <FL/Fl_Text_Editor.H>
00029
00030
00031 // TODO: add text composition?
00032 // TODO: add Fl::display
00033 // TODO: add copy/paste, drag/drop?
00034 // TODO: get key/get mouse?
00035 // TODO: system colors/colormaps
00036 // TODO: system menu?
00037 // TODO: native filechooser
00038 // TODO: native message boxes
00039 // TODO: read screen to image
00040 // TODO: application shortcuts
00041
00042 class Fl_Window;
00043 class Fl_RGB_Image;
00044 class Fl_Group;
00045 class Fl_Input;
00046 class Fl_System_Driver;
00047
00055 class Fl_Screen_Driver {
00056
00057 protected:
00058     Fl_Screen_Driver();
00059     virtual ~Fl_Screen_Driver();
00060
00061     static const int MAX_SCREEN = 16;
00062
00063     int num_screens;
00064     static float fl_intersection(int x1, int y1, int w1, int h1,
00065                                int x2, int y2, int w2, int h2);
00066
00067 public:
00068     static int keyboard_screen_scaling; // true means ctrl+/-/0/ resize windows
00069     static char bg_set;
00070     static char bg2_set;
00071     static char fg_set;
00072     static Fl_System_Driver *system_driver;
00073     // These flags are useful after calling XParseGeometry(). They indicate which of its
00074     // arguments contain meaningful data upon return.
00075     static const int fl_NoValue;
00076     static const int fl_WidthValue;
00077     static const int fl_HeightValue;
00078     static const int fl_XValue;
00079     static const int fl_YValue;
00080     static const int fl_XNegative;
00081     static const int fl_YNegative;
00082     // Next 2 are used when transient scale windows are implemented as popups
00083     static Fl_Window *transient_scale_parent;
00084     static void del_transient_window(void *);
00085     // key_table and key_table_size are used in fl_shortcut to translate key names
00086     struct Keyname {
00087         unsigned int key;
00088         const char* name;
00089     } *key_table;
00090     int key_table_size;
00091
00092     virtual float scale(int) { return 1; }
00093     virtual void scale(int /*n*/, float /*f*/) {}
00094     static Fl_Screen_Driver *newScreenDriver();
00095     // implement to process the -display argument and support the DISPLAY env var
00096     virtual void display(const char *) {}
00097     // default implementation should be enough
00098     virtual int XParseGeometry(const char* string, int* x, int* y, unsigned int* width, unsigned int*

```

```

height);
00099 // the default implementation is most probably enough
00100 virtual void own_colormap() {}
00101 // the default implementation of shortcut_add_key_name() is in src/fl_shortcut.cxx
00102 virtual const char *shortcut_add_key_name(unsigned key, char *p, char *buf, const char **);
00103 // whether a platform uses additional code in Fl_Menu::handle_part1(int e)
00104 virtual int need_menu_handle_part1_extra() {return 0;}
00105 // whether a platform uses additional code in Fl_Menu::handle(int e)
00106 virtual int need_menu_handle_part2() {return 0;}
00107 // implement functions telling whether a key is pressed
00108 virtual int event_key(int) {return 0;}
00109 virtual int get_key(int) {return 0;}
00110 virtual int visual(int flags);
00111 // --- screen configuration
00112 virtual void init() {}
00113 virtual int x() { return 0; }
00114 virtual int y() { return 0; }
00115 virtual int w() { return 800; } // default, FL_OVERRIDE in driver!
00116 virtual int h() { return 600; } // default, FL_OVERRIDE in driver!
00117 virtual int screen_count();
00118 void screen_xywh(int &X, int &Y, int &W, int &H, int mx, int my);
00119 virtual void screen_xywh(int &X, int &Y, int &W, int &H, int /*n*/) {
00120     X = 0;
00121     Y = 0;
00122     W = 800;
00123     H = 600;
00124 }
00125 void screen_xywh(int &X, int &Y, int &W, int &H, int mx, int my, int mw, int mh);
00126 virtual bool screen_boundaries_known() { return true; }
00127 virtual int screen_num(int x, int y);
00128 virtual int screen_num(int x, int y, int w, int h);
00129 virtual void screen_dpi(float &h, float &v, int n = 0) { // FL_OVERRIDE in driver!
00130     h = 72;
00131     v = 72;
00132     (void)n;
00133 }
00134 void screen_work_area(int &X, int &Y, int &W, int &H, int mx, int my);
00135 virtual void screen_work_area(int &X, int &Y, int &W, int &H, int n) {
00136     screen_xywh(X, Y, W, H, n);
00137 }
00138 // --- audible output
00139 virtual void beep(int) {}
00140 // --- global events
00141 virtual void flush() {} // must FL_OVERRIDE
00142 virtual void grab(Fl_Window *) {}
00143 // --- global colors
00144 /* the default implementation of parse_color() may be enough */
00145 virtual int parse_color(const char *p, uchar &r, uchar &g, uchar &b);
00146 virtual void get_system_colors();
00147 /* the default implementation of get_system_scheme() may be enough */
00148 virtual const char *get_system_scheme();
00149
00150 static int secret_input_character;
00151 /* Implement to indicate whether complex text input may involve marked text.
00152    When it does, has_marked_text returns non zero.
00153 */
00154 virtual int has_marked_text() const { return 0; }
00155 // implement so text-editing widgets support dead keys
00156 virtual int compose(int &del) {
00157     del = 0;
00158     return 0;
00159 }
00160 // default implementation may be enough
00161 virtual void compose_reset();
00162 // implement to support drag-n-drop. use_selection = 1 means the GUI is welcome to display
00163 // the selected text during the D&D operation
00164 virtual int dnd(int use_selection = 0) { (void)use_selection; return 0; }
00165 // null means no platform-specific key bindings for Fl_Text_Editor
00166 Fl_Text_Editor::Key_Binding *text_editor_extra_key_bindings;
00167 // default implementation may be enough
00168 virtual int text_display_can_leak() const { return 0; }
00169
00170 // if no keyboard is connected on a touch or pen device, the system on-screen keyboard is
00171 // requested
00172 virtual void request_keyboard() {}
00173 // we no longer need the on-screen keyboard; it's up to the system to hide it
00174 virtual void release_keyboard() {}
00175
00176 /* Member function read_win_rectangle() supports public functions
00177    fl_read_image() and fl_capture_window() which capture pixel data from
00178    a window (or also from an offscreen buffer with fl_read_image).
00179
00180    If 'may_capture_subwins' is true, an implementation may or may not capture
00181    also the content of subwindows embedded in 'win'. If subwindows were captured,
00182    *'did_capture_subwins' is returned set to true. If read_win_rectangle()
00183    is called with 'may_capture_subwins' set to true, 'did_capture_subwins' should
00184    be set before the call to the address of a boolean set to false.

```

```

00185     The implementation of this virtual function for the macOS platform has the
00186     capability of capturing subwindows when asked for.
00187
00188     A platform may also use its read_win_rectangle() implementation to capture
00189     window decorations (e.g., title bar). In that case, it is called by
00190     Fl_XXX_Window_Driver::capture_titlebar_and_borders().
00191
00192     win is the window to capture from, or NULL to capture from the current offscreen
00193     */
00194     virtual Fl_RGB_Image *read_win_rectangle(int /*X*/, int /*Y*/, int /*w*/, int /*h*/, Fl_Window *,
00195                                             bool may_capture_subwins = false,
00196                                             bool *did_capture_subwins = NULL) {
00197         (void)may_capture_subwins;
00198         (void)did_capture_subwins;
00199         return NULL;
00200     }
00201     static void write_image_inside(Fl_RGB_Image *to, Fl_RGB_Image *from, int to_x, int to_y);
00202     static Fl_RGB_Image *traverse_to_gl_subwindows(Fl_Group *g, int x, int y, int w, int h,
00203                                                    Fl_RGB_Image *full_img);
00204     static size_t convert_crlf(char *s, size_t len);
00205     // optional platform-specific key handling for Fl_Input widget
00206     // the default implementation may be enough
00207     virtual int input_widget_handle_key(int key, unsigned mods, unsigned shift, Fl_Input *input);
00208     // implement to support Fl::get_mouse()
00209     virtual int get_mouse(int /*x*/, int /*y*/) { return 0; }
00210     // optional methods to enable/disable input methods for complex scripts
00211     virtual void enable_im() {}
00212     virtual void disable_im() {}
00213     // calls open_display_platform() and then does platform-independent work
00214     void open_display();
00215     // implement to open access to the display
00216     virtual void open_display_platform() {}
00217     // optional method to close display access
00218     virtual void close_display() {}
00219     // compute dimensions of an Fl_Offscreen
00220     virtual void offscreen_size(Fl_Offscreen, int /*width*/, int /*height*/) {}
00221
00222     void rescale_all_windows_from_screen(int screen, float f, float old_f);
00223     static void transient_scale_display(float f, int nscreen);
00224     // need export to fltk_gl.so because used in glut_compatibility.cxx
00225     static FL_EXPORT int scale_handler(int event);
00226     virtual void desktop_scale_factor() {}
00227     void use_startup_scale_factor();
00228     enum APP_SCALING_CAPABILITY {
00229         NO_APP_SCALING = 0,
00230         SYSTEMWIDE_APP_SCALING,
00231         PER_SCREEN_APP_SCALING
00232     };
00233     virtual APP_SCALING_CAPABILITY rescalable() { return NO_APP_SCALING; }
00234     // supports Fl_Window::default_icons()
00235     virtual void default_icons(const Fl_RGB_Image *icons[], int count);
00236     // implement to support copy-to-clipboard
00237     virtual void copy(const char * /*stuff*/, int /*len*/, int /*clipboard*/, const char * /*type*/) {}
00238     // implement to support paste-from-clipboard
00239     virtual void paste(Fl_Widget &, int /*clipboard*/, const char * /*type*/) {}
00240     // implement to support paste-from-clipboard
00241     virtual int clipboard_contains(const char * /*type*/) {return 0;}
00242     // implement to support paste-from-clipboard
00243     virtual void clipboard_notify_change() {}
00244     // next 3 are related to Input Methods
00245     virtual void set_spot(int font, int size, int X, int Y, int W, int H, Fl_Window *win);
00246     virtual void reset_spot();
00247     virtual void set_status(int X, int Y, int W, int H);
00248     virtual float base_scale(int numscreen);
00249
00250 #endif // !FL_SCREEN_DRIVER_H
00251

```

12.232 Fl_String.H

```

00001 //
00002 // Basic Fl_String header for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 2021-2023 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php

```

```

00015 //
00016
00017 #ifndef _FL_Fl_String_H_
00018 #define _FL_Fl_String_H_
00019
00030 #include <FL/Fl_Export.H>
00031
00032 #include <string>
00033
00034 #if 1
00035
00036 // Since fltk 1.5.0, Fl_String is no longer needed and now maps to std::string.
00037 // Fl_String is not part of the public API. All occurrences of Fl_String in the
00038 // core library and in Fluid should be replaced with std::string. When done,
00039 // src/Fl_String.H and src/Fl_String.cxx can be deleted.
00040 using Fl_String = std::string;
00041
00042 #else
00043
00044 // See: https://en.cppreference.com/w/cpp/string/basic\_string/basic\_string
00045
00072 class FL_EXPORT Fl_String {
00073 private:
00074     /*
00075      * FLTK does no small string optimization.
00076      * If the string is empty and capacity is not set, buffer_ will be NULL.
00077      */
00078     char *buffer_;
00079     int size_;
00080     int capacity_;
00081
00082     void init_();
00083     void grow_(int n);
00084     void shrink_(int n);
00085     Fl_String &replace_(int at, int n_del, const char *src, int n_ins);
00086
00087 protected:
00088     static const char NUL;
00089
00090 public:
00091     static const int npos;
00092
00093     // ---- Assignment
00094     Fl_String();
00095     Fl_String(const Fl_String &str);
00096     Fl_String(const char *cstr);
00097     Fl_String(const char *str, int size);
00098     ~Fl_String();
00099     Fl_String& operator=(const Fl_String &str);
00100     Fl_String& operator=(const char *cstr);
00101     Fl_String &assign(const Fl_String &str);
00102     Fl_String &assign(const char *cstr);
00103     Fl_String &assign(const char *str, int size);
00104
00105     // ---- Element Access
00106     char at(int pos) const;
00107     char operator[](int n) const;
00108     char &operator[](int n);
00109     const char *data() const;
00110     char *data();
00111     const char *c_str() const;
00112
00113     // ---- Capacity
00114     bool empty() const;
00115     int size() const;
00116     void reserve(int n);
00117     int capacity() const;
00118     void shrink_to_fit();
00119
00120     // --- Operations
00121     void clear();
00122     Fl_String &insert(int at, const char *src, int n_ins=npow);
00123     Fl_String &insert(int at, const Fl_String &src);
00124     Fl_String &erase(int at, int n_del);
00125     void push_back(char c);
00126     void pop_back();
00127     Fl_String &append(const char *src, int n_ins=npow);
00128     Fl_String &append(const Fl_String &src);
00129     Fl_String &append(char c);
00130     Fl_String &operator+=(const char *src);
00131     Fl_String &operator+=(const Fl_String &src);
00132     Fl_String &operator+=(char c);
00133     int find(const Fl_String &needle, int start_pos=0) const;
00134     Fl_String &replace(int at, int n_del, const char *src, int n_ins=npow);
00135     Fl_String &replace(int at, int n_del, const Fl_String &src);
00136     Fl_String substr(int pos=0, int n=npow) const;

```



```

00138 void resize(int n);
00139
00140 // --- Non Standard
00141 int strlen() const;
00142 void debug(const char *info = 0) const;
00143 void hexdump(const char *info = 0) const;
00144 }; // class Fl_String
00145
00146 // ---- Non-member functions
00147 FL_EXPORT Fl_String operator+(const Fl_String& lhs, const Fl_String& rhs);
00148 FL_EXPORT Fl_String operator+(const Fl_String& lhs, const char* rhs);
00149 FL_EXPORT bool operator==(const Fl_String & lhs, const Fl_String & rhs);
00150 FL_EXPORT bool operator!=(const Fl_String & lhs, const Fl_String & rhs);
00151
00152 #endif
00153
00159 #endif // _FL_Fl_String_H_

```

12.233 Fl_Sys_Menu_Bar_Driver.H

```

00001 //
00002 // system menu bar widget for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2017 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef Fl_Sys_Menu_Bar_Driver_H
00018 #define Fl_Sys_Menu_Bar_Driver_H
00019
00020 #if !defined(FL_DOXYGEN)
00021
00022 #include <FL/Fl_Sys_Menu_Bar.H>
00023
00024 class Fl_Sys_Menu_Bar_Driver {
00025     friend class Fl_Sys_Menu_Bar;
00026 public:
00027     static Fl_Sys_Menu_Bar::window_menu_style_enum window_menu_style_;
00028     static Fl_Sys_Menu_Bar_Driver *driver_; // to be assigned with a unique object of this class or of a
00029     derived class
00030     Fl_Sys_Menu_Bar *bar;
00031     Fl_Sys_Menu_Bar_Driver();
00032     virtual ~Fl_Sys_Menu_Bar_Driver();
00033     virtual void update() {}
00034     virtual void draw() { bar->Fl_Menu_Bar::draw(); }
00035     virtual void about(Fl_Callback *, void *) {}
00036     virtual int add(const char* label, int shortcut, Fl_Callback *cb, void *user_data, int flags) {
00037         return bar->Fl_Menu_Bar::add(label, shortcut, cb, user_data, flags);
00038     }
00039     virtual int add(const char* str) { return bar->Fl_Menu_Bar::add(str); }
00040     virtual int insert(int index, const char* label, int shortcut, Fl_Callback *cb, void *user_data, int
00041     flags) {
00042         return bar->Fl_Menu_Bar::insert(index, label, shortcut, cb, user_data, flags);
00043     }
00044     virtual void menu(const Fl_Menu_Item *m) { bar->Fl_Menu_Bar::menu(m); }
00045     virtual void shortcut(int i, int s) { bar->Fl_Menu_Bar::shortcut(i, s); }
00046     virtual void setonly(Fl_Menu_Item *item) { bar->Fl_Menu_Bar::setonly(item); }
00047     virtual void clear() { bar->Fl_Menu_Bar::clear(); }
00048     virtual int clear_submenu(int index) { return bar->Fl_Menu_Bar::clear_submenu(index); }
00049     virtual void remove(int index) { bar->Fl_Menu_Bar::remove(index); }
00050     virtual void replace(int index, const char *name) { bar->Fl_Menu_Bar::replace(index, name); }
00051     virtual void mode(int i, int fl) { bar->Fl_Menu_Bar::mode(i, fl); }
00052     virtual void create_window_menu() {}
00053     virtual void play_menu(const Fl_Menu_Item *) {}
00054     static Fl_Sys_Menu_Bar::window_menu_style_enum window_menu_style() { return window_menu_style_; }
00055     static void window_menu_style(Fl_Sys_Menu_Bar::window_menu_style_enum style) { window_menu_style_ =
00056     style; }
00057 };
00058 #endif // !defined(FL_DOXYGEN)
00059 #endif // Fl_Sys_Menu_Bar_Driver_H

```

12.234 Fl_System_Driver.H

```

00001 //
00002 // A base class for platform specific system calls
00003 // for the Fast Light Tool Kit (FLTK).
00004 //
00005 // Copyright 2010-2022 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file.  If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //      https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //      https://www.fltk.org/bugs.php
00016 //
00017
00018 /* Class hierarchy
00019 + Fl_System_Driver
00020 | + Fl_Posix_System_Driver
00021 | | + Fl_Unix_System_Driver
00022 | | + Fl_Darwin_System_Driver
00023 | + Fl_WinAPI_System_Driver
00024 */
00025
00026 #ifndef FL_SYSTEM_DRIVER_H
00027 #define FL_SYSTEM_DRIVER_H
00028
00029 #include <FL/Fl.H>
00030 #include <FL/Fl_Export.H>
00031 #include <FL/filename.H>
00032 #include <FL/Fl_Preferences.H>
00033 #include <stdio.h>
00034 #include <stdlib.h>
00035 #include <stdarg.h>
00036 #include <string.h>
00037 #include <time.h>
00038
00039 class Fl_File_Icon;
00040 class Fl_File_Browser;
00041 class Fl_Pixmap;
00042 class Fl_Widget;
00043 class Fl_Sys_Menu_Bar_Driver;
00044
00045 class Fl_System_Driver {
00046     friend class Fl;
00047 protected:
00048     // implement once for each platform
00049     static Fl_System_Driver *newSystemDriver();
00050     Fl_System_Driver();
00051     static bool awake_ring_empty();
00052     int filename_relative(char *to, int tolen, const char *from, const char *base, bool
case_sensitive);
00053 public:
00054     virtual ~Fl_System_Driver();
00055     static int command_key;
00056     static int control_key;
00057
00058     // implement if the system adds unwanted program argument(s)
00059     virtual int single_arg(const char *) { return 0; }
00060     // implement if the system adds unwanted program argument pair(s)
00061     virtual int arg_and_value(const char * /*name*/, const char * /*value*/) { return 0; }
00062     static void warning(const char* format, ...);
00063     // implement to set the default effect of Fl::warning()
00064     virtual void warning(const char* format, va_list args);
00065     static void error(const char* format, ...);
00066     // implement to set the default effect of Fl::error()
00067     virtual void error(const char* format, va_list args);
00068     static void fatal(const char* format, ...);
00069     // implement to set the default effect of Fl::error()
00070     virtual void fatal(const char* format, va_list args);
00071
00072     // implement these to support cross-platform file operations
00073     virtual char *utf2mbcs(const char *s) {return (char*)s;}
00074     virtual char *getenv(const char*) {return NULL;}
00075     virtual int putenv(const char *) {return -1;}
00076     virtual int open(const char* /*f*/, int /*oflags*/, int /*pmode*/) {return -1;}
00077
00078     // implement these to support cross-platform string operations
00079     virtual char *strdup(const char *) {return NULL;}
00080
00081     // Note: the default implementation ignores the 'binary' argument.
00082     // Some platforms (notably Windows) may use this argument.
00083     virtual int open_ext(const char* f, int /*binary*/, int oflags, int pmode) {
00084         return this->open(f, oflags, pmode);
00085     }
00086

```

```

00100     }
00101     virtual FILE *fopen(const char* f, const char *mode);
00102     virtual int system(const char*) {return -1;}
00103     virtual int execvp(const char * /*file*/, char *const * /*argv*/) {return -1;}
00104     virtual int chmod(const char* /*f*/, int /*mode*/) {return -1;}
00105     virtual int access(const char* /*f*/, int /*mode*/) {return -1;}
00106     virtual int flstat(const char* /*f*/, struct stat *) {return -1;}
00107     virtual char *getcwd(char* /*b*/, int /*l*/) {return NULL;}
00108     virtual int chdir(const char*) {return -1;}
00109     virtual int unlink(const char*) {return -1;}
00110     virtual int mkdir(const char* /*f*/, int /*mode*/) {return -1;}
00111     virtual int rmdir(const char*) {return -1;}
00112     virtual int rename(const char* /*f*/, const char * /*n*/) {return -1;}
00113
00114     // Windows commandline argument conversion to UTF-8.
00115     // Default implementation: no-op, overridden only on Windows
00116     virtual int args_to_utf8(int argc, char ** &argv) {return argc;}
00117
00118     // the default implementation of these utf8... functions should be enough
00119     virtual unsigned utf8towc(const char* src, unsigned srclen, wchar_t* dst, unsigned dstlen);
00120     virtual unsigned utf8fromwc(char* dst, unsigned dstlen, const wchar_t* src, unsigned srclen);
00121     virtual int utf8locale() {return 1;}
00122     virtual unsigned utf8to_mb(const char* src, unsigned srclen, char* dst, unsigned dstlen);
00123     virtual unsigned utf8from_mb(char* dst, unsigned dstlen, const char* src, unsigned srclen);
00124     // implement to shield fprintf() from locale changes in decimal point
00125     virtual int clocale_vprintf(FILE *output, const char *format, va_list args);
00126     virtual int clocale_vsnprintf(char *output, size_t output_size, const char *format, va_list args);
00127     virtual int clocale_vsscanf(const char *input, const char *format, va_list args);
00128     // implement scandir-like function
00129     virtual int filename_list(const char * /*d*/, dirent ***,
00130                             int (* /*sort*/)(struct dirent **, struct dirent **),
00131                             char *errmsg=NULL, int errmsg_sz=0) {
00132         (void)errmsg; (void)errmsg_sz;
00133         return -1;
00134     }
00135     // the default implementation of filename_expand() may be enough
00136     virtual int filename_expand(char *to, int tolen, const char *from);
00137     // to implement
00138     virtual const char *getpwnam(const char *) {return NULL;}
00139     // the default implementation of filename_relative() is in src/filename_absolute.cxx and may be
00140     // enough
00141     virtual int filename_relative(char *to, int tolen, const char *from, const char *base);
00142     // the default implementation of filename_absolute() is in src/filename_absolute.cxx and may be
00143     // enough
00144     virtual int filename_absolute(char *to, int tolen, const char *from, const char *base);
00145     // the default implementation of filename_isdir() is in src/filename_isdir.cxx and may be enough
00146     virtual int filename_isdir(const char* n);
00147     // the default implementation of filename_isdir_quick() is in src/filename_isdir.cxx and may be
00148     // enough
00149     virtual int filename_isdir_quick(const char* n);
00150     // the default implementation of filename_ext() is in src/filename_ext.cxx and may be enough
00151     virtual const char *filename_ext(const char *buf);
00152     // implement to support fl_filename_name()
00153     virtual const char *filename_name(const char *buf) {return buf;}
00154     // implement to support fl_open_uri()
00155     virtual int open_uri(const char * /*uri*/, char * /*msg*/, int /*msglen*/) {return 0;}
00156     // the default implementation of use_tooltip_timeout_condition() may be enough
00157     virtual int use_tooltip_timeout_condition() {return 0;}
00158     // the default implementation of use_recent_tooltip_fix() may be enough
00159     virtual int use_recent_tooltip_fix() {return 0;}
00160     // the default implementation of need_test_shortcut_extra() may be enough
00161     virtual int need_test_shortcut_extra() {return 0;}
00162     // implement to support Fl_File_Browser::load()
00163     virtual int file_browser_load_filesystem(Fl_File_Browser *, char * /*filename*/, int /*lname*/,
00164                                             Fl_File_Icon *) {return 0;}
00165     // the default implementation of file_browser_load_directory() should be enough
00166     virtual int file_browser_load_directory(const char *directory, char *filename, size_t name_size,
00167                                             dirent ***pfiles, Fl_File_Sort_F *sort,
00168                                             char *errmsg=NULL, int errmsg_sz=0);
00169     // implement to support Fl_Preferences
00170     virtual void newUUID(char *uuidBuffer) { uuidBuffer[0] = 0; }
00171     // implement to support Fl_Preferences
00172     virtual char *preference_rootnode(Fl_Preferences *, Fl_Preferences::Root,
00173                                       const char * /*vendor*/,
00174                                       const char * /*application*/) {return NULL;}
00175     // the default implementation of preferences_need_protection_check() may be enough
00176     virtual int preferences_need_protection_check() {return 0;}
00177     // implement to support Fl_Plugin_Manager::load()
00178     virtual void *load(const char *) {return NULL;}
00179     // the default implementation is most probably enough
00180     virtual void png_extra_rgba_processing(unsigned char * /*array*/, int /*w*/, int /*h*/) {}
00181     // the default implementation is most probably enough
00182     virtual const char *next_dir_sep(const char *start) {return strchr(start, '/');}
00183     // implement to support threading
00184     virtual void awake(void*) {}
00185     virtual int lock() {return 1;}
00186     virtual void unlock() {}

```

```

00183 virtual void* thread_message() {return NULL;}
00184 // implement to support Fl_File_Icon
00185 virtual int file_type(const char *filename);
00186 // implement to return the user's home directory name
00187 virtual const char *home_directory_name() { return ""; }
00188 // the default implementation is most probably enough
00189 virtual const char *filesystems_label() { return "File Systems"; }
00190 // return TRUE means \ same as / in file names
00191 virtual int backslash_as_slash() {return 0;}
00192 // return TRUE means : indicates a drive letter in file names
00193 virtual int colon_is_drive() {return 0;}
00194 // return TRUE means that files whose name begins with dot are hidden
00195 virtual int dot_file_hidden() {return 0;}
00196 // return TRUE when file names are case insensitive
00197 virtual int case_insensitive_filenames() {return 0;}
00198 // the implementations of local_to_latin1() and latin1_to_local() are in fl_encoding_latin1.cxx
00199 virtual const char *local_to_latin1(const char *t, int n);
00200 virtual const char *latin1_to_local(const char *t, int n);
00201 // the implementations of local_to_mac_roman() and mac_roman_to_local() are in
    fl_encoding_mac_roman.cxx
00202 virtual const char *local_to_mac_roman(const char *t, int n);
00203 virtual const char *mac_roman_to_local(const char *t, int n);
00204 // draw default tree view expando button
00205 virtual void tree_draw_expando_button(int x, int y, bool state, bool active);
00206 // the default implementation of tree_connector_style() is in Fl_Tree_Prefs.cxx and can be enough
00207 virtual int tree_connector_style();
00208 virtual void add_fd(int fd, int when, Fl_FD_Handler cb, void* = 0);
00209 virtual void add_fd(int fd, Fl_FD_Handler cb, void* = 0);
00210 virtual void remove_fd(int, int when);
00211 virtual void remove_fd(int);
00212 // the default implementation of open_callback() may be enough
00213 virtual void open_callback(void (*) (const char *));
00214 // The default implementation may be enough.
00215 virtual void gettime(time_t *sec, int *usec);
00216 // The default implementation of the next 4 functions may be enough.
00217 virtual const char *shift_name() { return "Shift"; }
00218 virtual const char *meta_name() { return "Meta"; }
00219 virtual const char *alt_name() { return "Alt"; }
00220 virtual const char *control_name() { return "Ctrl"; }
00221 virtual Fl_Sys_Menu_Bar_Driver *sys_menu_bar_driver() { return NULL; }
00222 virtual void lock_ring() {}
00223 virtual void unlock_ring() {}
00224 virtual double wait(double); // must FL_OVERRIDE
00225 virtual int ready() { return 0; } // must FL_OVERRIDE
00226 virtual int close_fd(int) {return -1;} // to close a file descriptor
00227 };
00228
00229 #endif // FL_SYSTEM_DRIVER_H
00230

```

12.235 Fl_Timeout.cxx File Reference

```

#include <config.h>
#include "Fl_Timeout.h"
#include "Fl_System_Driver.H"
#include <stdio.h>
#include <math.h>

```

12.236 Fl_Timeout.h File Reference

[Fl_Timeout](#) handling.

```
#include <FL/Fl.H>
```

Classes

- class [Fl_Timeout](#)

The internal class [Fl_Timeout](#) handles all timeout related functions.

Macros

- #define `FL_TIMEOUT_DEBUG 0`

12.236.1 Detailed Description

[Fl_Timeout](#) handling.

This file contains implementations of:

- [Fl::add_timeout\(\)](#)
- [Fl::repeat_timeout\(\)](#)
- [Fl::has_timeout\(\)](#)
- [Fl::remove_timeout\(\)](#)
- [Fl::remove_next_timeout\(\)](#)

and related methods of class [Fl_Timeout](#).

12.237 Fl_Timeout.h

[Go to the documentation of this file.](#)

```
00001 //
00002 // Header for timeout support functions for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Author: Albrecht Schlosser
00005 // Copyright 2021-2024 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //
00015 //     https://www.fltk.org/bugs.php
00016 //
00017
00018 #ifndef _src_Fl_Timeout_h_
00019 #define _src_Fl_Timeout_h_
00020
00021 #include <FL/Fl.H>
00022
00023 #define FL_TIMEOUT_DEBUG 0          // 1 = include debugging features, 0 = no
00024
00056 class Fl_Timeout {
00057 protected:
00058
00059     Fl_Timeout *next;                // ** Link to next timeout
00060     Fl_Timeout_Handler callback;    // the user's callback
00061     void *data;                     // the user's callback data
00062     double time;                     // delay until timeout
00063     int skip;                        // skip "new" (inserted) timers (issue #450)
00064
00065     // constructor
00066     Fl_Timeout() {
00067         next = 0;
00068         callback = 0;
00069         data = 0;
00070         time = 0;
00071         skip = 0;
00072     }
00073
00074     // destructor
00075     ~Fl_Timeout() {}
00076
00077     // get a new timer entry from the pool or allocate a new one
00078     static Fl_Timeout *get(double time, Fl_Timeout_Handler cb, void *data);
00079
00080     // insert this timer into the active timer queue, sorted by expiration time
00081     void insert();
00082
00083     // remove this timer from the active timer queue and
00084     // add it to the "current" timer stack
00085     void make_current();
00086
00087     // remove this timer from the current timer stack and
00088     // add it to the list of free timers
00089     void release();
00090
00091 }
```

```

00093     double delay() {
00094         return time;
00095     }
00096
00097     void delay(double t) {
00098         time = t;
00099     }
00100
00101
00102 public:
00103     // Returns whether the given timeout is active.
00104     static int has_timeout(Fl_Timeout_Handler cb, void *data);
00105
00106     // Add or remove timeouts
00107
00108     static void add_timeout(double time, Fl_Timeout_Handler cb, void *data);
00109     static void repeat_timeout(double time, Fl_Timeout_Handler cb, void *data);
00110     static void remove_timeout(Fl_Timeout_Handler cb, void *data);
00111     static int remove_next_timeout(Fl_Timeout_Handler cb, void *data = NULL, void **data_return = NULL);
00112
00113     // Elapse timeouts, i.e. calculate new delay time of all timers.
00114     // This does not call the timer callbacks.
00115     static void elapse_timeouts();
00116
00117     // Elapse timeouts and call timer callbacks.
00118     static void do_timeouts();
00119
00120     // Return the delay in seconds until the next timer expires.
00121     static double time_to_wait(double ttw);
00122
00123 #if FL_TIMEOUT_DEBUG
00124     // Write some statistics to stdout
00125     static void debug(int level = 1);
00126 #endif
00127
00128 protected:
00129
00130     static Fl_Timeout *current();
00131
00141     static Fl_Timeout *first_timeout;
00142
00147     static Fl_Timeout *free_timeout;
00148
00165     static Fl_Timeout *current_timeout;    // list of "current" timeouts
00166
00167 }; // class Fl_Timeout
00168
00169 #endif // _src_Fl_Timeout_h_

```

12.238 fl_vertex.cxx File Reference

Portable drawing code for drawing arbitrary shapes with simple 2D transformations.

```

#include <FL/Fl_Graphics_Driver.H>
#include <FL/Fl.H>
#include <FL/math.h>
#include <stdlib.h>

```

12.238.1 Detailed Description

Portable drawing code for drawing arbitrary shapes with simple 2D transformations.

12.239 Fl_Window_Driver.H

```

00001 //
00002 // A base class for platform specific window handling code
00003 // for the Fast Light Tool Kit (FLTK).
00004 //
00005 // Copyright 2010-2025 by Bill Spitzak and others.
00006 //
00007 // This library is free software. Distribution and use rights are outlined in
00008 // the file "COPYING" which should have been included with this file. If this
00009 // file is missing or damaged, see the license at:
00010 //
00011 //     https://www.fltk.org/COPYING.php
00012 //
00013 // Please see the following page on how to report bugs and issues:
00014 //

```

```

00015 //      https://www.fltk.org/bugs.php
00016 //
00017
00028 #ifndef FL_WINDOW_DRIVER_H
00029 #define FL_WINDOW_DRIVER_H
00030
00031 #include <FL/Fl_Export.H>
00032 #include <FL/Fl_Window.H>
00033 #include <FL/Fl_Overlay_Window.H>
00034
00035 #include <stdlib.h>
00036
00037 class Fl_X;
00038 class Fl_Image;
00039 class Fl_RGB_Image;
00040 class Fl_Image_Surface;
00041
00052 class Fl_Window_Driver
00053 {
00054     friend class Fl_Window;
00055 private:
00056     static bool is_a_rescale_; // true when a top-level window is being rescaled
00057 protected:
00058     Fl_Window *pWindow;
00059     int screen_num_; // number of screen where window is mapped
00060 public:
00061     Fl_Window_Driver(Fl_Window *);
00062     virtual ~Fl_Window_Driver();
00063     static Fl_Window_Driver *newWindowDriver(Fl_Window *);
00064     static fl_uintptr_t xid(const Fl_Window *win);
00065     static Fl_Window *find(fl_uintptr_t xid);
00066     int wait_for_expose_value;
00067     Fl_Image_Surface *other_xid; // offscreen bitmap (overlay and double-buffered windows)
00068     int screen_num();
00069     void screen_num(int n) { screen_num_ = n; }
00070
00071 // --- frequently used accessors to public window data
00072
00073 int x() const { return pWindow->x(); }
00074 int y() const { return pWindow->y(); }
00075 int w() const { return pWindow->w(); }
00076 int h() const { return pWindow->h(); }
00077 int border() const { return pWindow->border(); }
00078 int visible() const { return pWindow->visible(); }
00079 int visible_r() const { return pWindow->visible_r(); }
00080 int shown() const { return pWindow->shown(); }
00081 Fl_Group *parent() const { return pWindow->parent(); }
00082
00083 // --- accessors to private window data
00084 int is_resizable() const { return pWindow->is_resizable(); }
00085 void is_a_rescale(bool b) { is_a_rescale_ = b; }
00086 int fullscreen_screen_top();
00087 int fullscreen_screen_bottom();
00088 int fullscreen_screen_left();
00089 int fullscreen_screen_right();
00090 int* no_fullscreen_x() const { return &pWindow->no_fullscreen_x; }
00091 int* no_fullscreen_y() const { return &pWindow->no_fullscreen_y; }
00092 int* no_fullscreen_w() const { return &pWindow->no_fullscreen_w; }
00093 int* no_fullscreen_h() const { return &pWindow->no_fullscreen_h; }
00094 int force_position();
00095 void force_position(int c);
00096 void x(int X);
00097 void y(int Y);
00098 void current(Fl_Window *c);
00099 char show_iconic() const { return Fl_Window::show_next_window_iconic(); }
00100 void show_iconic(char c) { Fl_Window::show_next_window_iconic(c); }
00101 void flx(Fl_X *x) { pWindow->flx_ = x; }
00102 Fl_Cursor cursor_default() const { return pWindow->cursor_default; }
00103 void destroy_double_buffer();
00104 Fl_Window *overlay() const {
00105     return pWindow->as_overlay_window() ? pWindow->as_overlay_window()->overlay_ : NULL;
00106 }
00107 void overlay(Fl_Window *o) const {
00108     if (pWindow->as_overlay_window()) pWindow->as_overlay_window()->overlay_ = o;
00109 }
00110
00111 void resize_after_scale_change(int ns, float old_f, float new_f);
00112 void set_popup_window() const { pWindow->set_flag(Fl_Window::POPUP); }
00113 bool popup_window() const { return pWindow->flags() & Fl_Window::POPUP; }
00114
00115 // --- window data
00116 virtual int decorated_w() const { return w(); } // default, should be overridden by driver
00117 virtual int decorated_h() const { return h(); }
00118 virtual const Fl_Image* shape() const { return NULL; }
00119
00120 // --- window management

```

```

00133     virtual void take_focus();
00134     virtual void flush(); // the default implementation may be enough
00135     virtual void flush_double();
00136     virtual void flush_overlay();
00138     virtual void draw_begin();
00140     virtual void draw_end();
00141     void draw();
00142     virtual void make_current();
00143     virtual void label(const char *name, const char *mininame);
00144
00145     virtual void makeWindow() {}
00146     virtual void wait_for_expose();
00147     virtual void show();
00148     virtual void resize(int /*X*/, int /*Y*/, int /*W*/, int /*H*/) {}
00149     virtual void hide() {}
00150     int hide_common();
00151     virtual void map() {}
00152     virtual void unmap() {}
00153     virtual void fullscreen_on() {}
00154     virtual void fullscreen_off(int /*X*/, int /*Y*/, int /*W*/, int /*H*/) {}
00155     virtual void fullscreen_screens(bool /*on_off*/) {}
00156     virtual void maximize();
00157     virtual void un_maximize();
00158     virtual bool maximize_needs_hide() { return false; }
00159     void is_maximized(bool b) { pWindow->is_maximized_(b); }
00160     virtual void use_border();
00161     virtual void size_range();
00162     virtual void iconize() {}
00163     virtual void decoration_sizes(int *top, int *left, int *right, int *bottom) {
00164         *top = *left = *right = *bottom = 0;
00165     }
00166     virtual void show_with_args_begin() {}
00167     virtual void show_with_args_end(int /*argc*/, char ** /*argv*/) {}
00168     virtual int can_do_overlay();
00169     virtual void redraw_overlay();
00170
00171     // --- window cursor stuff
00172     virtual int set_cursor(Fl_Cursor);
00173     virtual int set_cursor(const Fl_RGB_Image*, int, int);
00174
00175     // --- window shape stuff
00176     void shape_pixmap(Fl_Image* pixmap); // platform-independent, support function
00177     virtual void shape(const Fl_Image*) {}
00178     virtual void shape_alpha_(Fl_Image*, int /*offset*/) {}
00179
00180     // --- window icon stuff
00181     virtual void icons(const Fl_RGB_Image* /*icons*/[], int /*count*/) {}
00182     virtual const void *icon() const {return NULL;}
00183     virtual void icon(const void *) {}
00184     virtual void free_icons() {}
00185
00186     // --- window printing/drawing helper
00187     virtual void capture_titlebar_and_borders(Fl_RGB_Image*& top, Fl_RGB_Image*& left,
00188                                             Fl_RGB_Image*& bottom, Fl_RGB_Image*& right);
00189     virtual int scroll(int /*src_x*/, int /*src_y*/, int /*src_w*/, int /*src_h*/,
00190                      int /*dest_x*/, int /*dest_y*/,
00191                      void (*) (void*, int,int,int,int), void*) { return 0; }
00192     static inline Fl_Window_Driver* driver(const Fl_Window *win) {return win->pWindowDriver;}
00193
00194     // --- support for menu windows
00195     // The default implementation of next 2 virtual members is enough if the
00196     // position of a window in a screen is known. Next static members may be useful
00197     // when that's not the case, as with Wayland.
00198     virtual void reposition_menu_window(int x, int y);
00199     virtual void menu_window_area(int &X, int &Y, int &W, int &H, int nscreen = -1);
00200     static Fl_Window *menu_parent(int *display_height = NULL);
00201     static Fl_Window *menu_leftorigin(Fl_Window*);
00202     static Fl_Window *menu_title(Fl_Window*);
00203     static int menu_itemheight(Fl_Window*);
00204     static int menu_barheight(Fl_Window*);
00205     static int menu_selected(Fl_Window*);
00206     static int *menu_offset_y(Fl_Window*);
00207     static bool is_floating_title(Fl_Window *);
00208     static void scroll_to_selected_item(Fl_Window *);
00209
00210     virtual fl_uintptr_t os_id() { return 0; }
00211     virtual void allow_expand_outside_parent() {}
00212 };
00213
00214 #endif // FL_WINDOW_DRIVER_H
00215

```


12.240 fl_write_png.cxx File Reference

PNG image support functions.

```
#include <config.h>
#include <FL/Fl_PNG_Image.H>
#include <FL/Fl_RGB_Image.H>
#include <FL/fl_string_functions.h>
#include <FL/fl_utf8.h>
#include <stdio.h>
#include <time.h>
```

Functions

- `int fl_write_png (const char *filename, const char *pixels, int w, int h, int d, int ld)`
Write raw image data to a PNG image file.
- `int fl_write_png (const char *filename, const unsigned char *pixels, int w, int h, int d, int ld)`
Write raw image data to a PNG image file.
- `int fl_write_png (const char *filename, FL_RGB_Image *img)`
Write an RGB(A) image to a PNG image file.

12.240.1 Detailed Description

PNG image support functions.

12.240.2 Function Documentation

12.240.2.1 fl_write_png() [1/3]

```
int fl_write_png (
    const char * filename,
    const char * pixels,
    int w,
    int h,
    int d,
    int ld)
```

Write raw image data to a PNG image file.

This is a very basic and restricted function to create a PNG image file from raw image data, e.g. a screenshot.

The image data must be aligned w/o gaps after each row ($ld = 0$ or $ld = w * d$) or ld must be the total length of each row, i.e. $w * d + \text{gapsize}$. If $ld == 0$ then $ld = w * d$ is assumed.

The total data size must be $(w * d + \text{gapsize}) * h = ld' * h$ where $ld' = w * d$ if $ld == 0$.

For further restrictions and return values please see `fl_write_png(const char *filename, FL_RGB_Image *img)`.

Parameters

in	<i>filename</i>	Output filename, extension should be '.png'
in	<i>pixels</i>	Image data
in	<i>w</i>	Image data width
in	<i>h</i>	Image data height
in	<i>d</i>	Image depth: 1 = GRAY, 2 = GRAY + alpha, 3 = RGB, 4 = RGBA
in	<i>ld</i>	Line delta: default (0) = $w * d$

Returns

success (0) or error code, see ...

See also

[fl_write_png\(const char *filename, Fl_RGB_Image *img\)](#)

12.240.2.2 fl_write_png() [2/3]

```
int fl_write_png (
    const char * filename,
    const unsigned char * pixels,
    int w,
    int h,
    int d,
    int ld)
```

Write raw image data to a PNG image file.

See also

[fl_write_png\(const char *filename, const char *pixels, int w, int h, int d, int ld\)](#)

12.240.2.3 fl_write_png() [3/3]

```
int fl_write_png (
    const char * filename,
    Fl_RGB_Image * img)
```

Write an RGB(A) image to a PNG image file.

This is a very basic and restricted function to create a PNG image file from an RGB image ([Fl_RGB_Image](#)).

The image data must be aligned w/o gaps, i.e. `ld()` **MUST** be zero or equal to `data_w() * data_h()`.

The image file is always written with the original image size `data_w()` and `data_h()`, even if the image has been scaled.

Image depth 1 (gray), 2 (gray + alpha channel), 3 (RGB) and 4 (RGBA) are supported.

Note

Currently there is no error handling except for errors when opening the file. This may be changed in the future.

Parameters

in	<i>filename</i>	Output filename, extension should be '.png'
in	<i>img</i>	RGB image to be written

Returns

success (0) or error code: negative values are errors

Return values

0	success, file has been written
-1	png or zlib library not available
-2	file open error

See also

[fl_write_png\(const char *, int, int, int, const unsigned char *\)](#)

12.241 Fl_XColor.H

```

00001 //
00002 // X-specific color definitions for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef FL_DOXYGEN
00018 #include <config.h>
00019 #include <FL/Enumerations.H>
00020
00021 // one of these for each color in fltk's "colormap":
00022 // if overlays are enabled, another one for the overlay
00023 struct Fl_XColor {
00024     unsigned char r,g,b; // actual color used by X
00025     unsigned char mapped; // true when XAllocColor done
00026     unsigned long pixel; // the X pixel to use
00027 };
00028 extern Fl_XColor fl_xmap[/*overlay*/][256];
00029
00030 // mask & shifts to produce xcolor for truecolor visuals:
00031 extern unsigned char fl_redmask, fl_greenmask, fl_bluemask;
00032 extern int fl_redshift, fl_greenshift, fl_blueshift, fl_extrashift;
00033 #endif // FL_DOXYGEN

```

12.242 flstring.h

```

00001 /*
00002  * Common string header file for the Fast Light Tool Kit (FLTK).
00003  * Internal use only (see "important note" below).
00004  *
00005  * Copyright 1998-2020 by Bill Spitzak and others.
00006  *
00007  * This library is free software. Distribution and use rights are outlined in
00008  * the file "COPYING" which should have been included with this file. If this
00009  * file is missing or damaged, see the license at:
00010  *
00011  *     https://www.fltk.org/COPYING.php
00012  *
00013  * Please see the following page on how to report bugs and issues:
00014  *
00015  *     https://www.fltk.org/bugs.php
00016  */
00017
00018 /*
00019  * Important note: this header file includes '<config.h>' !
00020  *
00021  * This header MUST NOT be included in public headers (i.e. in 'FL/') and
00022  * SHOULD NOT be included in test and demo programs (i.e. in 'test/' or
00023  * 'examples/') because it includes '<config.h>'.
00024  */
00025
00026 #ifndef flstring_h
00027 #   define flstring_h
00028
00029 #   include <FL/Fl_Export.H>
00030 #   include <config.h>
00031 #   include <stdio.h>
00032 #   include <stdarg.h>
00033 #   include <string.h>
00034 #   ifdef HAVE_STRINGS_H
00035 #       include <strings.h>
00036 #   endif /* HAVE_STRINGS_H */
00037 #   include <ctype.h>
00038 #   include <FL/fl_string_functions.h>
00039
00040 /*
00041  * Apparently Unixware defines "index" to strchr (!) rather than
00042  * providing a proper entry point or not providing the (obsolete)
00043  * BSD function. Make sure index is not defined...
00044  */
00045
00046 #   ifdef index

```

```

00047 #    undef index
00048 #    endif /* index */
00049
00050 /*
00051  * Visual C++ 2005 incorrectly displays a warning about the use of
00052  * POSIX APIs on Windows, which is supposed to be POSIX compliant...
00053  * Some of these functions are also defined in ISO C99...
00054  */
00055
00056 #    if defined(_WIN32) && !defined(__CYGWIN__) && !defined(__MINGW32__)
00057 #        define strcasecmp(s,t)    _stricmp((s), (t))
00058 #        define strncasecmp(s,t,n) _strnicmp((s), (t), (n))
00059 #    endif /* _WIN32 && ... */
00060
00061 #    ifdef __cplusplus
00062 extern "C" {
00063 #    endif /* __cplusplus */
00064
00065 FL_EXPORT extern int fl_snprintf(char *, size_t, const char *, ...);
00066 #    ifndef HAVE_SNPRINTF
00067 #        define snprintf fl_snprintf
00068 #    endif /* !HAVE_SNPRINTF */
00069
00070 FL_EXPORT extern int fl_vsnprintf(char *, size_t, const char *, va_list ap);
00071 #    ifndef HAVE_VSNPRINTF
00072 #        define vsnprintf fl_vsnprintf
00073 #    endif /* !HAVE_VSNPRINTF */
00074
00075 /*
00076  * strcpy() and strcat() are some really useful BSD string functions
00077  * that work the way strncpy() and strncat() *should* have worked.
00078  */
00079
00080 FL_EXPORT extern size_t fl_strlcat(char *, const char *, size_t);
00081 #    ifndef HAVE_STRLCAT
00082 #        define strlcat fl_strlcat
00083 #    endif /* !HAVE_STRLCAT */
00084
00085 /* promoted to <FL/fl_string_functions.h> */
00086 /* FL_EXPORT extern size_t fl_strlcpy(char *, const char *, size_t); */
00087 #    ifndef HAVE_STRLCPY
00088 #        define strlcpy fl_strlcpy
00089 #    endif /* !HAVE_STRLCPY */
00090
00091 /*
00092  * Locale independent ASCII string compare function,
00093  * does not introduce locale issues as with strcasecmp()
00094  */
00095 FL_EXPORT extern int fl_ascii_strcasecmp(const char *s, const char *t);
00096
00097 #    ifdef __cplusplus
00098 }
00099 #    endif /* __cplusplus */
00100
00101 #endif /* !flstring_h */

```

12.243 freeglut_teapot_data.h

```

00001 /*
00002  * freeglut_teapot_data.h
00003  *
00004  * The freeglut library teapot data include file.
00005  *
00006  * Permission is hereby granted, free of charge, to any person obtaining a
00007  * copy of this software and associated documentation files (the "Software"),
00008  * to deal in the Software without restriction, including without limitation
00009  * the rights to use, copy, modify, merge, publish, distribute, sublicense,
00010  * and/or sell copies of the Software, and to permit persons to whom the
00011  * Software is furnished to do so, subject to the following conditions:
00012  *
00013  * The above copyright notice and this permission notice shall be included
00014  * in all copies or substantial portions of the Software.
00015  *
00016  * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS
00017  * OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
00018  * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
00019  * PAWEL W. OLSZTA BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
00020  * IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
00021  * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
00022  */
00023
00024 #ifndef FREEGLUT_TEAPOT_DATA_H
00025 #define FREEGLUT_TEAPOT_DATA_H
00026
00027 /*

```

```

00028  * Original teapot code copyright follows:
00029  */
00030
00031 /*
00032  * (c) Copyright 1993, Silicon Graphics, Inc.
00033  *
00034  * ALL RIGHTS RESERVED
00035  *
00036  * Permission to use, copy, modify, and distribute this software
00037  * for any purpose and without fee is hereby granted, provided
00038  * that the above copyright notice appear in all copies and that
00039  * both the copyright notice and this permission notice appear in
00040  * supporting documentation, and that the name of Silicon
00041  * Graphics, Inc. not be used in advertising or publicity
00042  * pertaining to distribution of the software without specific,
00043  * written prior permission.
00044  *
00045  * THE MATERIAL EMBODIED ON THIS SOFTWARE IS PROVIDED TO YOU
00046  * "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR
00047  * OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF
00048  * MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO
00049  * EVENT SHALL SILICON GRAPHICS, INC. BE LIABLE TO YOU OR ANYONE
00050  * ELSE FOR ANY DIRECT, SPECIAL, INCIDENTAL, INDIRECT OR
00051  * CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER,
00052  * INCLUDING WITHOUT LIMITATION, LOSS OF PROFIT, LOSS OF USE,
00053  * SAVINGS OR REVENUE, OR THE CLAIMS OF THIRD PARTIES, WHETHER OR
00054  * NOT SILICON GRAPHICS, INC. HAS BEEN ADVISED OF THE POSSIBILITY
00055  * OF SUCH LOSS, HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
00056  * ARISING OUT OF OR IN CONNECTION WITH THE POSSESSION, USE OR
00057  * PERFORMANCE OF THIS SOFTWARE.
00058  *
00059  * US Government Users Restricted Rights
00060  *
00061  * Use, duplication, or disclosure by the Government is subject to
00062  * restrictions set forth in FAR 52.227.19(c)(2) or subparagraph
00063  * (c)(1)(ii) of the Rights in Technical Data and Computer
00064  * Software clause at DFARS 252.227-7013 and/or in similar or
00065  * successor clauses in the FAR or the DOD or NASA FAR
00066  * Supplement. Unpublished-- rights reserved under the copyright
00067  * laws of the United States. Contractor/manufacturer is Silicon
00068  * Graphics, Inc., 2011 N. Shoreline Blvd., Mountain View, CA
00069  * 94039-7311.
00070  *
00071  * OpenGL(TM) is a trademark of Silicon Graphics, Inc.
00072  */
00073
00074 /*
00075  * Rim, body, lid, and bottom data must be reflected in x and y;
00076  * handle and spout data across the y axis only.
00077  */
00078 static int patchdata[][16] =
00079 {
00080     { 102, 103, 104, 105, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 }, /* rim */
00081     { 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27 }, /* body */
00082     { 24, 25, 26, 27, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40 },
00083     { 96, 96, 96, 96, 97, 98, 99, 100, 101, 101, 101, 101, 0, 1, 2, 3 }, /* lid */
00084     { 0, 1, 2, 3, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117 },
00085     { 118, 118, 118, 118, 124, 122, 119, 121, 123, 126, 125, 120, 40, 39, 38, 37 }, /* bottom */
00086     { 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56 }, /* handle */
00087     { 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 28, 65, 66, 67 },
00088     { 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83 }, /* spout */
00089     { 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95 }
00090 };
00091
00092 static double cpdata[][3] =
00093 {
00094     { 0.2, 0, 2.7 }, { 0.2, -0.112, 2.7 }, { 0.112, -0.2, 2.7 }, { 0,
00095     -0.2, 2.7 }, { 1.3375, 0, 2.53125 }, { 1.3375, -0.749, 2.53125 },
00096     { 0.749, -1.3375, 2.53125 }, { 0, -1.3375, 2.53125 }, { 1.4375,
00097     0, 2.53125 }, { 1.4375, -0.805, 2.53125 }, { 0.805, -1.4375,
00098     2.53125 }, { 0, -1.4375, 2.53125 }, { 1.5, 0, 2.4 }, { 1.5, -0.84,
00099     2.4 }, { 0.84, -1.5, 2.4 }, { 0, -1.5, 2.4 }, { 1.75, 0, 1.875 },
00100     { 1.75, -0.98, 1.875 }, { 0.98, -1.75, 1.875 }, { 0, -1.75,
00101     1.875 }, { 2, 0, 1.35 }, { 2, -1.12, 1.35 }, { 1.12, -2, 1.35 },
00102     { 0, -2, 1.35 }, { 2, 0, 0.9 }, { 2, -1.12, 0.9 }, { 1.12, -2,
00103     0.9 }, { 0, -2, 0.9 }, { -2, 0, 0.9 }, { 2, 0, 0.45 }, { 2, -1.12,
00104     0.45 }, { 1.12, -2, 0.45 }, { 0, -2, 0.45 }, { 1.5, 0, 0.225 },
00105     { 1.5, -0.84, 0.225 }, { 0.84, -1.5, 0.225 }, { 0, -1.5, 0.225 },
00106     { 1.5, 0, 0.15 }, { 1.5, -0.84, 0.15 }, { 0.84, -1.5, 0.15 }, { 0,
00107     -1.5, 0.15 }, { -1.6, 0, 2.025 }, { -1.6, -0.3, 2.025 }, { -1.5,
00108     -0.3, 2.25 }, { -1.5, 0, 2.25 }, { -2.3, 0, 2.025 }, { -2.3, -0.3,
00109     2.025 }, { -2.5, -0.3, 2.25 }, { -2.5, 0, 2.25 }, { -2.7, 0,
00110     2.025 }, { -2.7, -0.3, 2.025 }, { -3, -0.3, 2.25 }, { -3, 0,
00111     2.25 }, { -2.7, 0, 1.8 }, { -2.7, -0.3, 1.8 }, { -3, -0.3, 1.8 },
00112     { -3, 0, 1.8 }, { -2.7, 0, 1.575 }, { -2.7, -0.3, 1.575 }, { -3,
00113     -0.3, 1.35 }, { -3, 0, 1.35 }, { -2.5, 0, 1.125 }, { -2.5, -0.3,
00114     1.125 }, { -2.65, -0.3, 0.9375 }, { -2.65, 0, 0.9375 }, { -2,

```

```

00115     -0.3, 0.9}, {-1.9, -0.3, 0.6}, {-1.9, 0, 0.6}, {1.7, 0,
00116     1.425}, {1.7, -0.66, 1.425}, {1.7, -0.66, 0.6}, {1.7, 0,
00117     0.6}, {2.6, 0, 1.425}, {2.6, -0.66, 1.425}, {3.1, -0.66,
00118     0.825}, {3.1, 0, 0.825}, {2.3, 0, 2.1}, {2.3, -0.25, 2.1},
00119     {2.4, -0.25, 2.025}, {2.4, 0, 2.025}, {2.7, 0, 2.4}, {2.7,
00120     -0.25, 2.4}, {3.3, -0.25, 2.4}, {3.3, 0, 2.4}, {2.8, 0,
00121     2.475}, {2.8, -0.25, 2.475}, {3.525, -0.25, 2.49375},
00122     {3.525, 0, 2.49375}, {2.9, 0, 2.475}, {2.9, -0.15, 2.475},
00123     {3.45, -0.15, 2.5125}, {3.45, 0, 2.5125}, {2.8, 0, 2.4},
00124     {2.8, -0.15, 2.4}, {3.2, -0.15, 2.4}, {3.2, 0, 2.4}, {0, 0,
00125     3.15}, {0.8, 0, 3.15}, {0.8, -0.45, 3.15}, {0.45, -0.8,
00126     3.15}, {0, -0.8, 3.15}, {0, 0, 2.85}, {1.4, 0, 2.4}, {1.4,
00127     -0.784, 2.4}, {0.784, -1.4, 2.4}, {0, -1.4, 2.4}, {0.4, 0,
00128     2.55}, {0.4, -0.224, 2.55}, {0.224, -0.4, 2.55}, {0, -0.4,
00129     2.55}, {1.3, 0, 2.55}, {1.3, -0.728, 2.55}, {0.728, -1.3,
00130     2.55}, {0, -1.3, 2.55}, {1.3, 0, 2.4}, {1.3, -0.728, 2.4},
00131     {0.728, -1.3, 2.4}, {0, -1.3, 2.4}, {0, 0, 0}, {1.425,
00132     -0.798, 0}, {1.5, 0, 0.075}, {1.425, 0, 0}, {0.798, -1.425,
00133     0}, {0, -1.5, 0.075}, {0, -1.425, 0}, {1.5, -0.84, 0.075},
00134     {0.84, -1.5, 0.075}
00135 };
00136
00137 static double tex[2][2][2] =
00138 {
00139     { {0.0, 0.0}, {1.0, 0.0} },
00140     { {0.0, 1.0}, {1.0, 1.0} }
00141 };
00142
00143
00144 #endif /* FREEGLUT_TEAPOT_DATA_H */
00145

```

12.244 mediumarrow.h

```

00001 #define mediumarrow_width 16
00002 #define mediumarrow_height 16
00003 static const unsigned char mediumarrow_bits[] = {
00004     0x40, 0x00, 0x60, 0x00, 0x70, 0x00, 0x78, 0x00, 0xfc, 0x3f, 0x78, 0x00,
00005     0x70, 0x00, 0x60, 0x02, 0x40, 0x06, 0x00, 0x0e, 0x00, 0x1e, 0xfc, 0x3f,
00006     0x00, 0x1e, 0x00, 0x0e, 0x00, 0x06, 0x00, 0x02};

```

12.245 numeric sort.c File Reference

```

#include <ctype.h>
#include <stdlib.h>
#include <string.h>
#include <FL/platform_types.h>
#include <FL/filename.H>
#include <FL/fl_utf8.h>

```

Functions

- `int fl_casenumersort` (struct dirent **A, struct dirent **B)
Compares directory entries alphanumerically (case-insensitive).
- `int fl_numersort` (struct dirent **A, struct dirent **B)
Compares directory entries alphanumerically (case-sensitive).

12.245.1 Function Documentation

12.245.1.1 fl_casenumersort()

```

int fl_casenumersort (
    struct dirent ** A,
    struct dirent ** B)

```

Compares directory entries alphanumerically (case-insensitive).

Note

This comparison is UTF-8 aware.

See also

[fl_numeric sort\(\)](#)

12.245.1.2 fl_numeric sort()

```
int fl_numeric sort (
    struct dirent ** A,
    struct dirent ** B)
```

Compares directory entries alphanumerically (case-sensitive).

Numbers are compared without sign, i.e. "-" is not taken as a sign of following numerical values. The following list of files would be in ascending order (examples are ASCII and numbers only for simplicity):

1. 1zzz.txt
2. 2xxx.txt
3. 19uuu.txt
4. 100aaa.txt
5. file1z.txt
6. file5a.txt
7. file5z.txt
8. file30z.txt
9. file200a.txt
10. temp+5.txt ('+' is lexically lower than '-')
11. temp-5.txt ('-' is not a sign)
12. temp-100.txt (100 is bigger than 5, no sign)

Parameters

in	A	first directory entry
in	B	second directory entry

Returns

comparison result (-1, 0, or +1)

Return values

-1	A < B
0	A == B
+1	A > B

Note

This comparison is UTF-8 aware.

See also

[fl_casenum eric sort\(\)](#)

12.246 print_button.h

```
00001 //
00002 // Header for "Print Window" functions for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2022 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 #ifndef _SRC_FL_PRINT_BUTTON_H_
00018 #define _SRC_FL_PRINT_BUTTON_H_
00019
00020 #include <FL/Fl_Window.H>
00021
00022 // These are all internal functions, do not FL_EXPORT these functions!
00023 // These functions are mplemented in src/print_button.cxx
00024
00025 // Create and initialize the "Print/copy front window" dialog window
00026
00027 int fl_create_print_window();
00028
00029 // Print a window or copy its contents to the clipboard.
00030
00031 int fl_print_or_copy_window(Fl_Window *win, bool grab_decoration, int mode);
00032
00033 #endif // _SRC_FL_PRINT_BUTTON_H_
```

12.247 print_panel.h

```
00001 //
00002 // Print panel for the Fast Light Tool Kit (FLTK).
00003 //
00004 // Copyright 1998-2010 by Bill Spitzak and others.
00005 //
00006 // This library is free software. Distribution and use rights are outlined in
00007 // the file "COPYING" which should have been included with this file. If this
00008 // file is missing or damaged, see the license at:
00009 //
00010 //     https://www.fltk.org/COPYING.php
00011 //
00012 // Please see the following page on how to report bugs and issues:
00013 //
00014 //     https://www.fltk.org/bugs.php
00015 //
00016
00017 //
00018 // This is a temporary file. It is only for development and will
00019 // probably be removed later.
00020 //
00021
00022 #ifndef print_panel_h
00023 #define print_panel_h
00024 #include <FL/Fl.H>
00025 #include <FL/Fl_Double_Window.H>
00026 #include <FL/Fl_Group.H>
00027 #include <FL/Fl_Choice.H>
00028 #include <FL/Fl_Button.H>
00029 #include <FL/Fl_Box.H>
00030 #include <FL/Fl_Round_Button.H>
00031 #include <FL/Fl_Input.H>
00032 #include <FL/Fl_Spinner.H>
00033 #include <FL/Fl_Check_Button.H>
00034 #include <FL/Fl_Return_Button.H>
00035 #include <FL/Fl_Progress.H>
00036 enum printing_style {SystemV, BSD};
00037 static Fl_Double_Window* make_print_panel();
00038 static void print_cb(Fl_Return_Button *, void *);
00039 static printing_style print_load();
00040 static void print_update_status();
00041 #endif
```


12.248 slowarrow.h

```
00001 #define slowarrow_width 16
00002 #define slowarrow_height 16
00003 static const unsigned char slowarrow_bits[] = {
00004     0x40, 0x00, 0x40, 0x00, 0x60, 0x00, 0x60, 0x00, 0xf0, 0x0f, 0x60, 0x00,
00005     0x60, 0x00, 0x40, 0x02, 0x40, 0x02, 0x00, 0x06, 0x00, 0x06, 0xf0, 0x0f,
00006     0x00, 0x06, 0x00, 0x06, 0x00, 0x02, 0x00, 0x02};
```

12.249 utf8_internal.h

```
00001 /*
00002  * Internal UTF-8 header file for the Fast Light Tool Kit (FLTK).
00003  *
00004  * Copyright 1998-2016 by Bill Spitzak and others.
00005  *
00006  * This library is free software. Distribution and use rights are outlined in
00007  * the file "COPYING" which should have been included with this file. If this
00008  * file is missing or damaged, see the license at:
00009  *
00010  *     https://www.fltk.org/COPYING.php
00011  *
00012  * Please see the following page on how to report bugs and issues:
00013  *
00014  *     https://www.fltk.org/bugs.php
00015  */
00016
00017 /*
00018  *-----
00019  * Note to editors:
00020  *-----
00021  *
00022  * This file may only contain common, platform-independent function
00023  * declarations used internally in FLTK. It may be #included everywhere
00024  * in source files in the library, but not in public header files.
00025  */
00026
00027 #ifndef _SRC__FL_UTF8_H
00028 #define _SRC__FL_UTF8_H
00029
00030 # ifdef __cplusplus
00031 extern "C" {
00032 # endif
00033
00034 unsigned short
00035 XUtf8IsNonSpacing(
00036     unsigned int ucs);
00037
00038 unsigned short
00039 XUtf8IsRightToLeft(
00040     unsigned int ucs);
00041
00042
00043 int
00044 XUtf8Tolower(
00045     int ucs);
00046
00047 int
00048 XUtf8Toupper(
00049     int ucs);
00050
00051
00052 # ifdef __cplusplus
00053 }
00054 # endif
00055
00056 #endif /* _SRC__FL_UTF8_H */
```

12.250 vsnprintf.c File Reference

Portable vsnprintf() implementation.

```
#include <stdio.h>
```

```
#include "flstring.h"
```

Functions

- `int fl_snprintf (char *str, size_t size, const char *fmt,...)`

- `int fl_vsnprintf` (char *buffer, size_t bufsz, const char *format, va_list ap)
FLTK's platform independent wrapper for the vsnprintf() C library function.

12.250.1 Detailed Description

Portable vsnprintf() implementation.

12.250.2 Function Documentation

12.250.2.1 fl_vsnprintf()

```
int fl_vsnprintf (
    char * buffer,
    size_t bufsz,
    const char * format,
    va_list ap)
```

FLTK's platform independent wrapper for the vsnprintf() C library function.

This function guarantees:

- access to vsnprintf(), even on systems that don't have it (FLTK's own built-in code is used)
- Guarantees NUL termination. Even if string expands larger than the buffer, a terminating NUL is included, unlike some implementations of vsnprintf(), notably Microsoft Visual Studio (pre-2015), which can leave the string unterminated when truncated.

If the build environment for FLTK has vsnprintf(), `fl_vsnprintf()` is just a wrapper around the compiler's provided function. Otherwise, if the function is NOT available, FLTK's own built-in version is provided.

The FLTK built in provides these style options:

- `%[ -+#]`
- `*` – padding width
- `.*` – precision width
- Data types: h, l, ll, L
- Floating point formats: E, G, e, f, g
- Integer formats: B, X, b, d, i, o, u, x
- Pointer format: p
- String/char: c, s, n

12.251 Xutf8.h

```
00001 /*
00002  * Author: Jean-Marc Lienher ( http://oksid.ch )
00003  * Copyright 2000-2010 by O'ksi'D.
00004  *
00005  * This library is free software. Distribution and use rights are outlined in
00006  * the file "COPYING" which should have been included with this file. If this
00007  * file is missing or damaged, see the license at:
00008  *
00009  *     https://www.fltk.org/COPYING.php
00010  *
00011  * Please see the following page on how to report bugs and issues:
00012  *
00013  *     https://www.fltk.org/bugs.php
00014  */
00015
00016 #if ! ( defined(_Xutf8_h) || defined(FL_DOXYGEN) )
00017 #define _Xutf8_h
00018
00019 # ifdef __cplusplus
00020 extern "C" {
00021 # endif
00022
```

```

00023 #include <X11/X.h>
00024 #include <X11/Xlib.h>
00025 #include <X11/Xlocale.h>
00026 #include <X11/Xutil.h>
00027 #include <FL/Fl_Export.H>
00028
00029 typedef struct {
00030     int nb_font;
00031     char **font_name_list;
00032     int *encodings;
00033     XFontStruct **fonts;
00034     Font fid;
00035     int ascent;
00036     int descent;
00037     int *ranges;
00038 } XUtf8FontStruct;
00039
00040 XUtf8FontStruct *
00041 XCreateUtf8FontStruct (
00042     Display          *dpy,
00043     const char       *base_font_name_list);
00044
00045 void
00046 XUtf8DrawString(
00047     Display          *display,
00048     Drawable         d,
00049     XUtf8FontStruct *font_set,
00050     GC               gc,
00051     int              x,
00052     int              y,
00053     const char       *string,
00054     int              num_bytes);
00055
00056 void
00057 XUtf8_measure_extents(
00058     Display          *display,
00059     Drawable         d,
00060     XUtf8FontStruct *font_set,
00061     GC               gc,
00062     int              *xx,
00063     int              *yy,
00064     int              *ww,
00065     int              *hh,
00066     const char       *string,
00067     int              num_bytes);
00068
00069 void
00070 XUtf8DrawRtlString(
00071     Display          *display,
00072     Drawable         d,
00073     XUtf8FontStruct *font_set,
00074     GC               gc,
00075     int              x,
00076     int              y,
00077     const char       *string,
00078     int              num_bytes);
00079
00080 void
00081 XUtf8DrawImageString(
00082     Display          *display,
00083     Drawable         d,
00084     XUtf8FontStruct *font_set,
00085     GC               gc,
00086     int              x,
00087     int              y,
00088     const char       *string,
00089     int              num_bytes);
00090
00091 int
00092 XUtf8TextWidth(
00093     XUtf8FontStruct *font_set,
00094     const char       *string,
00095     int              num_bytes);
00096
00097 int
00098 XUtf8UcsWidth(
00099     XUtf8FontStruct *font_set,
00100     unsigned int     ucs);
00101
00102 FL_EXPORT int
00103 fl_XGetUtf8FontAndGlyph(
00104     XUtf8FontStruct *font_set,
00105     unsigned int     ucs,
00106     XFontStruct      **fnt,
00107     unsigned short    *id);
00108
00109 void
00110 XFreeUtf8FontStruct (

```

```

00110         Display                *dpy,
00111         XUtf8FontStruct          *font_set);
00112
00113
00114     int
00115     XConvertUtf8ToUcs(
00116         const unsigned char      *buf,
00117         int                      len,
00118         unsigned int             *ucs);
00119
00120     int
00121     XConvertUcsToUtf8(
00122         unsigned int             ucs,
00123         char                    *buf);
00124
00125     int
00126     XUtf8CharByteLen(
00127         const unsigned char      *buf,
00128         int                      len);
00129
00130     int
00131     XCountUtf8Char(
00132         const unsigned char *buf,
00133         int len);
00134
00135     int
00136     XFastConvertUtf8ToUcs(
00137         const unsigned char      *buf,
00138         int                      len,
00139         unsigned int             *ucs);
00140
00141     long
00142     XKeysymToUcs(
00143         KeySym keysym);
00144
00145     #ifdef X_HAVE_UTF8_STRING
00146     #define XUtf8LookupString Xutf8LookupString
00147     #else
00148     int
00149     XUtf8LookupString(
00150         XIC ic,
00151         XKeyPressedEvent* event,
00152         char* buffer_return,
00153         int bytes_buffer,
00154         KeySym* keysym,
00155         Status* status_return);
00156     #endif
00157
00158     # ifdef __cplusplus
00159     }
00160     # endif
00161
00162     #endif

```

12.252 case.h

```

00001 /* spacing */
00002
00003 static const unsigned short ucs_table_0041[] = {
00004     /* U+0041 */ 0x0061,
00005     /* U+0042 */ 0x0062,
00006     /* U+0043 */ 0x0063,
00007     /* U+0044 */ 0x0064,
00008     /* U+0045 */ 0x0065,
00009     /* U+0046 */ 0x0066,
00010     /* U+0047 */ 0x0067,
00011     /* U+0048 */ 0x0068,
00012     /* U+0049 */ 0x0069,
00013     /* U+004A */ 0x006A,
00014     /* U+004B */ 0x006B,
00015     /* U+004C */ 0x006C,
00016     /* U+004D */ 0x006D,
00017     /* U+004E */ 0x006E,
00018     /* U+004F */ 0x006F,
00019     /* U+0050 */ 0x0070,
00020     /* U+0051 */ 0x0071,
00021     /* U+0052 */ 0x0072,
00022     /* U+0053 */ 0x0073,
00023     /* U+0054 */ 0x0074,
00024     /* U+0055 */ 0x0075,
00025     /* U+0056 */ 0x0076,
00026     /* U+0057 */ 0x0077,
00027     /* U+0058 */ 0x0078,
00028     /* U+0059 */ 0x0079,
00029     /* U+005A */ 0x007A,

```

```
00030 0x00,
00031 0x00,
00032 0x00,
00033 0x00,
00034 0x00,
00035 0x00,
00036 0x00,
00037 0x00,
00038 0x00,
00039 0x00,
00040 0x00,
00041 0x00,
00042 0x00,
00043 0x00,
00044 0x00,
00045 0x00,
00046 0x00,
00047 0x00,
00048 0x00,
00049 0x00,
00050 0x00,
00051 0x00,
00052 0x00,
00053 0x00,
00054 0x00,
00055 0x00,
00056 0x00,
00057 0x00,
00058 0x00,
00059 0x00,
00060 0x00,
00061 0x00,
00062 0x00,
00063 0x00,
00064 0x00,
00065 0x00,
00066 0x00,
00067 0x00,
00068 0x00,
00069 0x00,
00070 0x00,
00071 0x00,
00072 0x00,
00073 0x00,
00074 0x00,
00075 0x00,
00076 0x00,
00077 0x00,
00078 0x00,
00079 0x00,
00080 0x00,
00081 0x00,
00082 0x00,
00083 0x00,
00084 0x00,
00085 0x00,
00086 0x00,
00087 0x00,
00088 0x00,
00089 0x00,
00090 0x00,
00091 0x00,
00092 0x00,
00093 0x00,
00094 0x00,
00095 0x00,
00096 0x00,
00097 0x00,
00098 0x00,
00099 0x00,
00100 0x00,
00101 0x00,
00102 0x00,
00103 0x00,
00104 0x00,
00105 0x00,
00106 0x00,
00107 0x00,
00108 0x00,
00109 0x00,
00110 0x00,
00111 0x00,
00112 0x00,
00113 0x00,
00114 0x00,
00115 0x00,
00116 0x00,
```

```
00117 0x00,
00118 0x00,
00119 0x00,
00120 0x00,
00121 0x00,
00122 0x00,
00123 0x00,
00124 0x00,
00125 0x00,
00126 0x00,
00127 0x00,
00128 0x00,
00129 0x00,
00130 0x00,
00131 /* U+00C0 */ 0x00E0,
00132 /* U+00C1 */ 0x00E1,
00133 /* U+00C2 */ 0x00E2,
00134 /* U+00C3 */ 0x00E3,
00135 /* U+00C4 */ 0x00E4,
00136 /* U+00C5 */ 0x00E5,
00137 /* U+00C6 */ 0x00E6,
00138 /* U+00C7 */ 0x00E7,
00139 /* U+00C8 */ 0x00E8,
00140 /* U+00C9 */ 0x00E9,
00141 /* U+00CA */ 0x00EA,
00142 /* U+00CB */ 0x00EB,
00143 /* U+00CC */ 0x00EC,
00144 /* U+00CD */ 0x00ED,
00145 /* U+00CE */ 0x00EE,
00146 /* U+00CF */ 0x00EF,
00147 /* U+00D0 */ 0x00F0,
00148 /* U+00D1 */ 0x00F1,
00149 /* U+00D2 */ 0x00F2,
00150 /* U+00D3 */ 0x00F3,
00151 /* U+00D4 */ 0x00F4,
00152 /* U+00D5 */ 0x00F5,
00153 /* U+00D6 */ 0x00F6,
00154 0x00,
00155 /* U+00D8 */ 0x00F8,
00156 /* U+00D9 */ 0x00F9,
00157 /* U+00DA */ 0x00FA,
00158 /* U+00DB */ 0x00FB,
00159 /* U+00DC */ 0x00FC,
00160 /* U+00DD */ 0x00FD,
00161 /* U+00DE */ 0x00FE,
00162 0x00,
00163 0x00,
00164 0x00,
00165 0x00,
00166 0x00,
00167 0x00,
00168 0x00,
00169 0x00,
00170 0x00,
00171 0x00,
00172 0x00,
00173 0x00,
00174 0x00,
00175 0x00,
00176 0x00,
00177 0x00,
00178 0x00,
00179 0x00,
00180 0x00,
00181 0x00,
00182 0x00,
00183 0x00,
00184 0x00,
00185 0x00,
00186 0x00,
00187 0x00,
00188 0x00,
00189 0x00,
00190 0x00,
00191 0x00,
00192 0x00,
00193 0x00,
00194 0x00,
00195 /* U+0100 */ 0x0101,
00196 0x00,
00197 /* U+0102 */ 0x0103,
00198 0x00,
00199 /* U+0104 */ 0x0105,
00200 0x00,
00201 /* U+0106 */ 0x0107,
00202 0x00,
00203 /* U+0108 */ 0x0109,
```

```
00204 0x00,
00205 /* U+010A */ 0x010B,
00206 0x00,
00207 /* U+010C */ 0x010D,
00208 0x00,
00209 /* U+010E */ 0x010F,
00210 0x00,
00211 /* U+0110 */ 0x0111,
00212 0x00,
00213 /* U+0112 */ 0x0113,
00214 0x00,
00215 /* U+0114 */ 0x0115,
00216 0x00,
00217 /* U+0116 */ 0x0117,
00218 0x00,
00219 /* U+0118 */ 0x0119,
00220 0x00,
00221 /* U+011A */ 0x011B,
00222 0x00,
00223 /* U+011C */ 0x011D,
00224 0x00,
00225 /* U+011E */ 0x011F,
00226 0x00,
00227 /* U+0120 */ 0x0121,
00228 0x00,
00229 /* U+0122 */ 0x0123,
00230 0x00,
00231 /* U+0124 */ 0x0125,
00232 0x00,
00233 /* U+0126 */ 0x0127,
00234 0x00,
00235 /* U+0128 */ 0x0129,
00236 0x00,
00237 /* U+012A */ 0x012B,
00238 0x00,
00239 /* U+012C */ 0x012D,
00240 0x00,
00241 /* U+012E */ 0x012F,
00242 0x00,
00243 /* U+0130 */ 0x0,
00244 0x00,
00245 /* U+0132 */ 0x0133,
00246 0x00,
00247 /* U+0134 */ 0x0135,
00248 0x00,
00249 /* U+0136 */ 0x0137,
00250 0x00,
00251 0x00,
00252 /* U+0139 */ 0x013A,
00253 0x00,
00254 /* U+013B */ 0x013C,
00255 0x00,
00256 /* U+013D */ 0x013E,
00257 0x00,
00258 /* U+013F */ 0x0140,
00259 0x00,
00260 /* U+0141 */ 0x0142,
00261 0x00,
00262 /* U+0143 */ 0x0144,
00263 0x00,
00264 /* U+0145 */ 0x0146,
00265 0x00,
00266 /* U+0147 */ 0x0148,
00267 0x00,
00268 0x00,
00269 /* U+014A */ 0x014B,
00270 0x00,
00271 /* U+014C */ 0x014D,
00272 0x00,
00273 /* U+014E */ 0x014F,
00274 0x00,
00275 /* U+0150 */ 0x0151,
00276 0x00,
00277 /* U+0152 */ 0x0153,
00278 0x00,
00279 /* U+0154 */ 0x0155,
00280 0x00,
00281 /* U+0156 */ 0x0157,
00282 0x00,
00283 /* U+0158 */ 0x0159,
00284 0x00,
00285 /* U+015A */ 0x015B,
00286 0x00,
00287 /* U+015C */ 0x015D,
00288 0x00,
00289 /* U+015E */ 0x015F,
00290 0x00,
```

```
00291 /* U+0160 */ 0x0161,
00292 0x00,
00293 /* U+0162 */ 0x0163,
00294 0x00,
00295 /* U+0164 */ 0x0165,
00296 0x00,
00297 /* U+0166 */ 0x0167,
00298 0x00,
00299 /* U+0168 */ 0x0169,
00300 0x00,
00301 /* U+016A */ 0x016B,
00302 0x00,
00303 /* U+016C */ 0x016D,
00304 0x00,
00305 /* U+016E */ 0x016F,
00306 0x00,
00307 /* U+0170 */ 0x0171,
00308 0x00,
00309 /* U+0172 */ 0x0173,
00310 0x00,
00311 /* U+0174 */ 0x0175,
00312 0x00,
00313 /* U+0176 */ 0x0177,
00314 0x00,
00315 /* U+0178 */ 0x00FF,
00316 /* U+0179 */ 0x017A,
00317 0x00,
00318 /* U+017B */ 0x017C,
00319 0x00,
00320 /* U+017D */ 0x017E,
00321 0x00,
00322 0x00,
00323 0x00,
00324 /* U+0181 */ 0x0253,
00325 /* U+0182 */ 0x0183,
00326 0x00,
00327 /* U+0184 */ 0x0185,
00328 0x00,
00329 /* U+0186 */ 0x0254,
00330 /* U+0187 */ 0x0188,
00331 0x00,
00332 /* U+0189 */ 0x0,
00333 /* U+018A */ 0x0257,
00334 /* U+018B */ 0x018C,
00335 0x00,
00336 0x00,
00337 /* U+018E */ 0x0258,
00338 /* U+018F */ 0x0259,
00339 /* U+0190 */ 0x025B,
00340 /* U+0191 */ 0x0192,
00341 0x00,
00342 /* U+0193 */ 0x0260,
00343 /* U+0194 */ 0x0263,
00344 0x00,
00345 /* U+0196 */ 0x0269,
00346 /* U+0197 */ 0x0268,
00347 /* U+0198 */ 0x0199,
00348 0x00,
00349 0x00,
00350 0x00,
00351 /* U+019C */ 0x026F,
00352 /* U+019D */ 0x0272,
00353 0x00,
00354 /* U+019F */ 0x0,
00355 /* U+01A0 */ 0x01A1,
00356 0x00,
00357 /* U+01A2 */ 0x01A3,
00358 0x00,
00359 /* U+01A4 */ 0x01A5,
00360 0x00,
00361 0x00,
00362 /* U+01A7 */ 0x01A8,
00363 0x00,
00364 /* U+01A9 */ 0x0283,
00365 0x00,
00366 0x00,
00367 /* U+01AC */ 0x01AD,
00368 0x00,
00369 /* U+01AE */ 0x0288,
00370 /* U+01AF */ 0x01B0,
00371 0x00,
00372 /* U+01B1 */ 0x028A,
00373 /* U+01B2 */ 0x028B,
00374 /* U+01B3 */ 0x01B4,
00375 0x00,
00376 /* U+01B5 */ 0x01B6,
00377 0x00,
```



```
00378 /* U+01B7 */ 0x0292,
00379 /* U+01B8 */ 0x01B9,
00380 0x00,
00381 0x00,
00382 0x00,
00383 /* U+01BC */ 0x01BD,
00384 0x00,
00385 0x00,
00386 0x00,
00387 0x00,
00388 0x00,
00389 0x00,
00390 0x00,
00391 /* U+01C4 */ 0x01C6,
00392 /* U+01C5 */ 0x0,
00393 0x00,
00394 /* U+01C7 */ 0x01C9,
00395 /* U+01C8 */ 0x0,
00396 0x00,
00397 /* U+01CA */ 0x01CC,
00398 /* U+01CB */ 0x0,
00399 0x00,
00400 /* U+01CD */ 0x01CE,
00401 0x00,
00402 /* U+01CF */ 0x01D0,
00403 0x00,
00404 /* U+01D1 */ 0x01D2,
00405 0x00,
00406 /* U+01D3 */ 0x01D4,
00407 0x00,
00408 /* U+01D5 */ 0x01D6,
00409 0x00,
00410 /* U+01D7 */ 0x01D8,
00411 0x00,
00412 /* U+01D9 */ 0x01DA,
00413 0x00,
00414 /* U+01DB */ 0x01DC,
00415 0x00,
00416 0x00,
00417 /* U+01DE */ 0x01DF,
00418 0x00,
00419 /* U+01E0 */ 0x01E1,
00420 0x00,
00421 /* U+01E2 */ 0x01E3,
00422 0x00,
00423 /* U+01E4 */ 0x01E5,
00424 0x00,
00425 /* U+01E6 */ 0x01E7,
00426 0x00,
00427 /* U+01E8 */ 0x01E9,
00428 0x00,
00429 /* U+01EA */ 0x01EB,
00430 0x00,
00431 /* U+01EC */ 0x01ED,
00432 0x00,
00433 /* U+01EE */ 0x01EF,
00434 0x00,
00435 0x00,
00436 /* U+01F1 */ 0x01F3,
00437 /* U+01F2 */ 0x0,
00438 0x00,
00439 /* U+01F4 */ 0x01F5,
00440 0x00,
00441 0x00,
00442 0x00,
00443 0x00,
00444 0x00,
00445 /* U+01FA */ 0x01FB,
00446 0x00,
00447 /* U+01FC */ 0x01FD,
00448 0x00,
00449 /* U+01FE */ 0x01FF,
00450 0x00,
00451 /* U+0200 */ 0x0201,
00452 0x00,
00453 /* U+0202 */ 0x0203,
00454 0x00,
00455 /* U+0204 */ 0x0205,
00456 0x00,
00457 /* U+0206 */ 0x0207,
00458 0x00,
00459 /* U+0208 */ 0x0209,
00460 0x00,
00461 /* U+020A */ 0x020B,
00462 0x00,
00463 /* U+020C */ 0x020D,
00464 0x00,
```

```
00465 /* U+020E */ 0x020F,
00466 0x00,
00467 /* U+0210 */ 0x0211,
00468 0x00,
00469 /* U+0212 */ 0x0213,
00470 0x00,
00471 /* U+0214 */ 0x0215,
00472 0x00,
00473 /* U+0216 */ 0x0217,
00474 0x00,
00475 0x00,
00476 0x00,
00477 0x00,
00478 0x00,
00479 0x00,
00480 0x00,
00481 0x00,
00482 0x00,
00483 0x00,
00484 0x00,
00485 0x00,
00486 0x00,
00487 0x00,
00488 0x00,
00489 0x00,
00490 0x00,
00491 0x00,
00492 0x00,
00493 0x00,
00494 0x00,
00495 0x00,
00496 0x00,
00497 0x00,
00498 0x00,
00499 0x00,
00500 0x00,
00501 0x00,
00502 0x00,
00503 0x00,
00504 0x00,
00505 0x00,
00506 0x00,
00507 0x00,
00508 0x00,
00509 0x00,
00510 0x00,
00511 0x00,
00512 0x00,
00513 0x00,
00514 0x00,
00515 0x00,
00516 0x00,
00517 0x00,
00518 0x00,
00519 0x00,
00520 0x00,
00521 0x00,
00522 0x00,
00523 0x00,
00524 0x00,
00525 0x00,
00526 0x00,
00527 0x00,
00528 0x00,
00529 0x00,
00530 0x00,
00531 0x00,
00532 0x00,
00533 0x00,
00534 0x00,
00535 0x00,
00536 0x00,
00537 0x00,
00538 0x00,
00539 0x00,
00540 0x00,
00541 0x00,
00542 0x00,
00543 0x00,
00544 0x00,
00545 0x00,
00546 0x00,
00547 0x00,
00548 0x00,
00549 /* U+0262 */ 0x0,
00550 0x00,
00551 0x00,
```

```
00552 0x00,
00553 0x00,
00554 0x00,
00555 0x00,
00556 0x00,
00557 /* U+026A */ 0x0,
00558 0x00,
00559 0x00,
00560 0x00,
00561 0x00,
00562 0x00,
00563 0x00,
00564 0x00,
00565 0x00,
00566 0x00,
00567 /* U+0274 */ 0x0,
00568 0x00,
00569 /* U+0276 */ 0x0,
00570 0x00,
00571 0x00,
00572 0x00,
00573 0x00,
00574 0x00,
00575 0x00,
00576 0x00,
00577 0x00,
00578 0x00,
00579 /* U+0280 */ 0x0,
00580 /* U+0281 */ 0x0,
00581 0x00,
00582 0x00,
00583 0x00,
00584 0x00,
00585 0x00,
00586 0x00,
00587 0x00,
00588 0x00,
00589 0x00,
00590 0x00,
00591 0x00,
00592 0x00,
00593 0x00,
00594 /* U+028F */ 0x0,
00595 0x00,
00596 0x00,
00597 0x00,
00598 0x00,
00599 0x00,
00600 0x00,
00601 0x00,
00602 0x00,
00603 0x00,
00604 /* U+0299 */ 0x0,
00605 0x00,
00606 /* U+029B */ 0x0,
00607 /* U+029C */ 0x0,
00608 0x00,
00609 0x00,
00610 /* U+029F */ 0x0,
00611 0x00,
00612 0x00,
00613 0x00,
00614 0x00,
00615 0x00,
00616 0x00,
00617 0x00,
00618 0x00,
00619 0x00,
00620 0x00,
00621 0x00,
00622 0x00,
00623 0x00,
00624 0x00,
00625 0x00,
00626 0x00,
00627 0x00,
00628 0x00,
00629 0x00,
00630 0x00,
00631 0x00,
00632 0x00,
00633 /* U+02B6 */ 0x0,
00634 };
00635
00636 static const unsigned short ucs_table_0386[] = {
00637 /* U+0386 */ 0x03AC,
00638 0x00,
```

```
00639 /* U+0388 */ 0x03AD,
00640 /* U+0389 */ 0x03AE,
00641 /* U+038A */ 0x03AF,
00642 0x00,
00643 /* U+038C */ 0x03CC,
00644 0x00,
00645 /* U+038E */ 0x03CD,
00646 /* U+038F */ 0x03CE,
00647 0x00,
00648 /* U+0391 */ 0x03B1,
00649 /* U+0392 */ 0x03B2,
00650 /* U+0393 */ 0x03B3,
00651 /* U+0394 */ 0x03B4,
00652 /* U+0395 */ 0x03B5,
00653 /* U+0396 */ 0x03B6,
00654 /* U+0397 */ 0x03B7,
00655 /* U+0398 */ 0x03B8,
00656 /* U+0399 */ 0x03B9,
00657 /* U+039A */ 0x03BA,
00658 /* U+039B */ 0x03BB,
00659 /* U+039C */ 0x03BC,
00660 /* U+039D */ 0x03BD,
00661 /* U+039E */ 0x03BE,
00662 /* U+039F */ 0x03BF,
00663 /* U+03A0 */ 0x03C0,
00664 /* U+03A1 */ 0x03C1,
00665 0x00,
00666 /* U+03A3 */ 0x03C3,
00667 /* U+03A4 */ 0x03C4,
00668 /* U+03A5 */ 0x03C5,
00669 /* U+03A6 */ 0x03C6,
00670 /* U+03A7 */ 0x03C7,
00671 /* U+03A8 */ 0x03C8,
00672 /* U+03A9 */ 0x03C9,
00673 /* U+03AA */ 0x03CA,
00674 /* U+03AB */ 0x03CB,
00675 0x00,
00676 0x00,
00677 0x00,
00678 0x00,
00679 0x00,
00680 0x00,
00681 0x00,
00682 0x00,
00683 0x00,
00684 0x00,
00685 0x00,
00686 0x00,
00687 0x00,
00688 0x00,
00689 0x00,
00690 0x00,
00691 0x00,
00692 0x00,
00693 0x00,
00694 0x00,
00695 0x00,
00696 0x00,
00697 0x00,
00698 0x00,
00699 0x00,
00700 0x00,
00701 0x00,
00702 0x00,
00703 0x00,
00704 0x00,
00705 0x00,
00706 0x00,
00707 0x00,
00708 0x00,
00709 0x00,
00710 0x00,
00711 0x00,
00712 0x00,
00713 /* U+03D2 */ 0x03D2,
00714 /* U+03D3 */ 0x03D3,
00715 /* U+03D4 */ 0x03D4,
00716 0x00,
00717 0x00,
00718 0x00,
00719 0x00,
00720 0x00,
00721 /* U+03DA */ 0x03DA,
00722 0x00,
00723 /* U+03DC */ 0x03DC,
00724 0x00,
00725 /* U+03DE */ 0x03DE,
```

```
00726 0x00,
00727 /* U+03E0 */ 0x03E0,
00728 0x00,
00729 /* U+03E2 */ 0x03E3,
00730 0x00,
00731 /* U+03E4 */ 0x03E5,
00732 0x00,
00733 /* U+03E6 */ 0x03E7,
00734 0x00,
00735 /* U+03E8 */ 0x03E9,
00736 0x00,
00737 /* U+03EA */ 0x03EB,
00738 0x00,
00739 /* U+03EC */ 0x03ED,
00740 0x00,
00741 /* U+03EE */ 0x03EF,
00742 0x00,
00743 0x00,
00744 0x00,
00745 0x00,
00746 0x00,
00747 0x00,
00748 0x00,
00749 0x00,
00750 0x00,
00751 0x00,
00752 0x00,
00753 0x00,
00754 0x00,
00755 0x00,
00756 0x00,
00757 0x00,
00758 0x00,
00759 0x00,
00760 /* U+0401 */ 0x0451,
00761 /* U+0402 */ 0x0452,
00762 /* U+0403 */ 0x0453,
00763 /* U+0404 */ 0x0454,
00764 /* U+0405 */ 0x0455,
00765 /* U+0406 */ 0x0456,
00766 /* U+0407 */ 0x0457,
00767 /* U+0408 */ 0x0458,
00768 /* U+0409 */ 0x0459,
00769 /* U+040A */ 0x045A,
00770 /* U+040B */ 0x045B,
00771 /* U+040C */ 0x045C,
00772 0x00,
00773 /* U+040E */ 0x045E,
00774 /* U+040F */ 0x045F,
00775 /* U+0410 */ 0x0430,
00776 /* U+0411 */ 0x0431,
00777 /* U+0412 */ 0x0432,
00778 /* U+0413 */ 0x0433,
00779 /* U+0414 */ 0x0434,
00780 /* U+0415 */ 0x0435,
00781 /* U+0416 */ 0x0436,
00782 /* U+0417 */ 0x0437,
00783 /* U+0418 */ 0x0438,
00784 /* U+0419 */ 0x0439,
00785 /* U+041A */ 0x043A,
00786 /* U+041B */ 0x043B,
00787 /* U+041C */ 0x043C,
00788 /* U+041D */ 0x043D,
00789 /* U+041E */ 0x043E,
00790 /* U+041F */ 0x043F,
00791 /* U+0420 */ 0x0440,
00792 /* U+0421 */ 0x0441,
00793 /* U+0422 */ 0x0442,
00794 /* U+0423 */ 0x0443,
00795 /* U+0424 */ 0x0444,
00796 /* U+0425 */ 0x0445,
00797 /* U+0426 */ 0x0446,
00798 /* U+0427 */ 0x0447,
00799 /* U+0428 */ 0x0448,
00800 /* U+0429 */ 0x0449,
00801 /* U+042A */ 0x044A,
00802 /* U+042B */ 0x044B,
00803 /* U+042C */ 0x044C,
00804 /* U+042D */ 0x044D,
00805 /* U+042E */ 0x044E,
00806 /* U+042F */ 0x044F,
00807 0x00,
00808 0x00,
00809 0x00,
00810 0x00,
00811 0x00,
00812 0x00,
```

```
00813 0x00,
00814 0x00,
00815 0x00,
00816 0x00,
00817 0x00,
00818 0x00,
00819 0x00,
00820 0x00,
00821 0x00,
00822 0x00,
00823 0x00,
00824 0x00,
00825 0x00,
00826 0x00,
00827 0x00,
00828 0x00,
00829 0x00,
00830 0x00,
00831 0x00,
00832 0x00,
00833 0x00,
00834 0x00,
00835 0x00,
00836 0x00,
00837 0x00,
00838 0x00,
00839 0x00,
00840 0x00,
00841 0x00,
00842 0x00,
00843 0x00,
00844 0x00,
00845 0x00,
00846 0x00,
00847 0x00,
00848 0x00,
00849 0x00,
00850 0x00,
00851 0x00,
00852 0x00,
00853 0x00,
00854 0x00,
00855 /* U+0460 */ 0x0461,
00856 0x00,
00857 /* U+0462 */ 0x0463,
00858 0x00,
00859 /* U+0464 */ 0x0465,
00860 0x00,
00861 /* U+0466 */ 0x0467,
00862 0x00,
00863 /* U+0468 */ 0x0469,
00864 0x00,
00865 /* U+046A */ 0x046B,
00866 0x00,
00867 /* U+046C */ 0x046D,
00868 0x00,
00869 /* U+046E */ 0x046F,
00870 0x00,
00871 /* U+0470 */ 0x0471,
00872 0x00,
00873 /* U+0472 */ 0x0473,
00874 0x00,
00875 /* U+0474 */ 0x0475,
00876 0x00,
00877 /* U+0476 */ 0x0477,
00878 0x00,
00879 /* U+0478 */ 0x0479,
00880 0x00,
00881 /* U+047A */ 0x047B,
00882 0x00,
00883 /* U+047C */ 0x047D,
00884 0x00,
00885 /* U+047E */ 0x047F,
00886 0x00,
00887 /* U+0480 */ 0x0481,
00888 0x00,
00889 0x00,
00890 0x00,
00891 0x00,
00892 0x00,
00893 0x00,
00894 0x00,
00895 0x00,
00896 0x00,
00897 0x00,
00898 0x00,
00899 0x00,
```

```
00900 0x00,
00901 0x00,
00902 0x00,
00903 /* U+0490 */ 0x0491,
00904 0x00,
00905 /* U+0492 */ 0x0493,
00906 0x00,
00907 /* U+0494 */ 0x0495,
00908 0x00,
00909 /* U+0496 */ 0x0497,
00910 0x00,
00911 /* U+0498 */ 0x0499,
00912 0x00,
00913 /* U+049A */ 0x049B,
00914 0x00,
00915 /* U+049C */ 0x049D,
00916 0x00,
00917 /* U+049E */ 0x049F,
00918 0x00,
00919 /* U+04A0 */ 0x04A1,
00920 0x00,
00921 /* U+04A2 */ 0x04A3,
00922 0x00,
00923 /* U+04A4 */ 0x04A5,
00924 0x00,
00925 /* U+04A6 */ 0x04A7,
00926 0x00,
00927 /* U+04A8 */ 0x04A9,
00928 0x00,
00929 /* U+04AA */ 0x04AB,
00930 0x00,
00931 /* U+04AC */ 0x04AD,
00932 0x00,
00933 /* U+04AE */ 0x04AF,
00934 0x00,
00935 /* U+04B0 */ 0x04B1,
00936 0x00,
00937 /* U+04B2 */ 0x04B3,
00938 0x00,
00939 /* U+04B4 */ 0x04B5,
00940 0x00,
00941 /* U+04B6 */ 0x04B7,
00942 0x00,
00943 /* U+04B8 */ 0x04B9,
00944 0x00,
00945 /* U+04BA */ 0x04BB,
00946 0x00,
00947 /* U+04BC */ 0x04BD,
00948 0x00,
00949 /* U+04BE */ 0x04BF,
00950 0x00,
00951 0x00,
00952 /* U+04C1 */ 0x04C2,
00953 0x00,
00954 /* U+04C3 */ 0x04C4,
00955 0x00,
00956 0x00,
00957 0x00,
00958 /* U+04C7 */ 0x04C8,
00959 0x00,
00960 0x00,
00961 0x00,
00962 /* U+04CB */ 0x04CC,
00963 0x00,
00964 0x00,
00965 0x00,
00966 0x00,
00967 /* U+04D0 */ 0x04D1,
00968 0x00,
00969 /* U+04D2 */ 0x04D3,
00970 0x00,
00971 /* U+04D4 */ 0x04D5,
00972 0x00,
00973 /* U+04D6 */ 0x04D7,
00974 0x00,
00975 /* U+04D8 */ 0x04D9,
00976 0x00,
00977 /* U+04DA */ 0x04DB,
00978 0x00,
00979 /* U+04DC */ 0x04DD,
00980 0x00,
00981 /* U+04DE */ 0x04DF,
00982 0x00,
00983 /* U+04E0 */ 0x04E1,
00984 0x00,
00985 /* U+04E2 */ 0x04E3,
00986 0x00,
```

```
00987 /* U+04E4 */ 0x04E5,
00988 0x00,
00989 /* U+04E6 */ 0x04E7,
00990 0x00,
00991 /* U+04E8 */ 0x04E9,
00992 0x00,
00993 /* U+04EA */ 0x04EB,
00994 0x00,
00995 0x00,
00996 0x00,
00997 /* U+04EE */ 0x04EF,
00998 0x00,
00999 /* U+04F0 */ 0x04F1,
01000 0x00,
01001 /* U+04F2 */ 0x04F3,
01002 0x00,
01003 /* U+04F4 */ 0x04F5,
01004 0x00,
01005 0x00,
01006 0x00,
01007 /* U+04F8 */ 0x04F9,
01008 0x00,
01009 0x00,
01010 0x00,
01011 0x00,
01012 0x00,
01013 0x00,
01014 0x00,
01015 0x00,
01016 0x00,
01017 0x00,
01018 0x00,
01019 0x00,
01020 0x00,
01021 0x00,
01022 0x00,
01023 0x00,
01024 0x00,
01025 0x00,
01026 0x00,
01027 0x00,
01028 0x00,
01029 0x00,
01030 0x00,
01031 0x00,
01032 0x00,
01033 0x00,
01034 0x00,
01035 0x00,
01036 0x00,
01037 0x00,
01038 0x00,
01039 0x00,
01040 0x00,
01041 0x00,
01042 0x00,
01043 0x00,
01044 0x00,
01045 0x00,
01046 0x00,
01047 0x00,
01048 0x00,
01049 0x00,
01050 0x00,
01051 0x00,
01052 0x00,
01053 0x00,
01054 0x00,
01055 0x00,
01056 0x00,
01057 0x00,
01058 0x00,
01059 0x00,
01060 0x00,
01061 0x00,
01062 0x00,
01063 0x00,
01064 /* U+0531 */ 0x0561,
01065 /* U+0532 */ 0x0562,
01066 /* U+0533 */ 0x0563,
01067 /* U+0534 */ 0x0564,
01068 /* U+0535 */ 0x0565,
01069 /* U+0536 */ 0x0566,
01070 /* U+0537 */ 0x0567,
01071 /* U+0538 */ 0x0568,
01072 /* U+0539 */ 0x0569,
01073 /* U+053A */ 0x056A,
```



```
01074 /* U+053B */ 0x056B,
01075 /* U+053C */ 0x056C,
01076 /* U+053D */ 0x056D,
01077 /* U+053E */ 0x056E,
01078 /* U+053F */ 0x056F,
01079 /* U+0540 */ 0x0570,
01080 /* U+0541 */ 0x0571,
01081 /* U+0542 */ 0x0572,
01082 /* U+0543 */ 0x0573,
01083 /* U+0544 */ 0x0574,
01084 /* U+0545 */ 0x0575,
01085 /* U+0546 */ 0x0576,
01086 /* U+0547 */ 0x0577,
01087 /* U+0548 */ 0x0578,
01088 /* U+0549 */ 0x0579,
01089 /* U+054A */ 0x057A,
01090 /* U+054B */ 0x057B,
01091 /* U+054C */ 0x057C,
01092 /* U+054D */ 0x057D,
01093 /* U+054E */ 0x057E,
01094 /* U+054F */ 0x057F,
01095 /* U+0550 */ 0x0580,
01096 /* U+0551 */ 0x0581,
01097 /* U+0552 */ 0x0582,
01098 /* U+0553 */ 0x0583,
01099 /* U+0554 */ 0x0584,
01100 /* U+0555 */ 0x0585,
01101 /* U+0556 */ 0x0586,
01102 };
01103
01104 static const unsigned short ucs_table_10A0[] = {
01105 /* U+10A0 */ 0x10D0,
01106 /* U+10A1 */ 0x10D1,
01107 /* U+10A2 */ 0x10D2,
01108 /* U+10A3 */ 0x10D3,
01109 /* U+10A4 */ 0x10D4,
01110 /* U+10A5 */ 0x10D5,
01111 /* U+10A6 */ 0x10D6,
01112 /* U+10A7 */ 0x10D7,
01113 /* U+10A8 */ 0x10D8,
01114 /* U+10A9 */ 0x10D9,
01115 /* U+10AA */ 0x10DA,
01116 /* U+10AB */ 0x10DB,
01117 /* U+10AC */ 0x10DC,
01118 /* U+10AD */ 0x10DD,
01119 /* U+10AE */ 0x10DE,
01120 /* U+10AF */ 0x10DF,
01121 /* U+10B0 */ 0x10E0,
01122 /* U+10B1 */ 0x10E1,
01123 /* U+10B2 */ 0x10E2,
01124 /* U+10B3 */ 0x10E3,
01125 /* U+10B4 */ 0x10E4,
01126 /* U+10B5 */ 0x10E5,
01127 /* U+10B6 */ 0x10E6,
01128 /* U+10B7 */ 0x10E7,
01129 /* U+10B8 */ 0x10E8,
01130 /* U+10B9 */ 0x10E9,
01131 /* U+10BA */ 0x10EA,
01132 /* U+10BB */ 0x10EB,
01133 /* U+10BC */ 0x10EC,
01134 /* U+10BD */ 0x10ED,
01135 /* U+10BE */ 0x10EE,
01136 /* U+10BF */ 0x10EF,
01137 /* U+10C0 */ 0x10F0,
01138 /* U+10C1 */ 0x10F1,
01139 /* U+10C2 */ 0x10F2,
01140 /* U+10C3 */ 0x10F3,
01141 /* U+10C4 */ 0x10F4,
01142 /* U+10C5 */ 0x10F5,
01143 };
01144
01145 static const unsigned short ucs_table_1E00[] = {
01146 /* U+1E00 */ 0x1E01,
01147 0x00,
01148 /* U+1E02 */ 0x1E03,
01149 0x00,
01150 /* U+1E04 */ 0x1E05,
01151 0x00,
01152 /* U+1E06 */ 0x1E07,
01153 0x00,
01154 /* U+1E08 */ 0x1E09,
01155 0x00,
01156 /* U+1E0A */ 0x1E0B,
01157 0x00,
01158 /* U+1E0C */ 0x1E0D,
01159 0x00,
01160 /* U+1E0E */ 0x1E0F,
```

```
01161 0x00,
01162 /* U+1E10 */ 0x1E11,
01163 0x00,
01164 /* U+1E12 */ 0x1E13,
01165 0x00,
01166 /* U+1E14 */ 0x1E15,
01167 0x00,
01168 /* U+1E16 */ 0x1E17,
01169 0x00,
01170 /* U+1E18 */ 0x1E19,
01171 0x00,
01172 /* U+1E1A */ 0x1E1B,
01173 0x00,
01174 /* U+1E1C */ 0x1E1D,
01175 0x00,
01176 /* U+1E1E */ 0x1E1F,
01177 0x00,
01178 /* U+1E20 */ 0x1E21,
01179 0x00,
01180 /* U+1E22 */ 0x1E23,
01181 0x00,
01182 /* U+1E24 */ 0x1E25,
01183 0x00,
01184 /* U+1E26 */ 0x1E27,
01185 0x00,
01186 /* U+1E28 */ 0x1E29,
01187 0x00,
01188 /* U+1E2A */ 0x1E2B,
01189 0x00,
01190 /* U+1E2C */ 0x1E2D,
01191 0x00,
01192 /* U+1E2E */ 0x1E2F,
01193 0x00,
01194 /* U+1E30 */ 0x1E31,
01195 0x00,
01196 /* U+1E32 */ 0x1E33,
01197 0x00,
01198 /* U+1E34 */ 0x1E35,
01199 0x00,
01200 /* U+1E36 */ 0x1E37,
01201 0x00,
01202 /* U+1E38 */ 0x1E39,
01203 0x00,
01204 /* U+1E3A */ 0x1E3B,
01205 0x00,
01206 /* U+1E3C */ 0x1E3D,
01207 0x00,
01208 /* U+1E3E */ 0x1E3F,
01209 0x00,
01210 /* U+1E40 */ 0x1E41,
01211 0x00,
01212 /* U+1E42 */ 0x1E43,
01213 0x00,
01214 /* U+1E44 */ 0x1E45,
01215 0x00,
01216 /* U+1E46 */ 0x1E47,
01217 0x00,
01218 /* U+1E48 */ 0x1E49,
01219 0x00,
01220 /* U+1E4A */ 0x1E4B,
01221 0x00,
01222 /* U+1E4C */ 0x1E4D,
01223 0x00,
01224 /* U+1E4E */ 0x1E4F,
01225 0x00,
01226 /* U+1E50 */ 0x1E51,
01227 0x00,
01228 /* U+1E52 */ 0x1E53,
01229 0x00,
01230 /* U+1E54 */ 0x1E55,
01231 0x00,
01232 /* U+1E56 */ 0x1E57,
01233 0x00,
01234 /* U+1E58 */ 0x1E59,
01235 0x00,
01236 /* U+1E5A */ 0x1E5B,
01237 0x00,
01238 /* U+1E5C */ 0x1E5D,
01239 0x00,
01240 /* U+1E5E */ 0x1E5F,
01241 0x00,
01242 /* U+1E60 */ 0x1E61,
01243 0x00,
01244 /* U+1E62 */ 0x1E63,
01245 0x00,
01246 /* U+1E64 */ 0x1E65,
01247 0x00,
```

```
01248 /* U+1E66 */ 0x1E67,
01249 0x00,
01250 /* U+1E68 */ 0x1E69,
01251 0x00,
01252 /* U+1E6A */ 0x1E6B,
01253 0x00,
01254 /* U+1E6C */ 0x1E6D,
01255 0x00,
01256 /* U+1E6E */ 0x1E6F,
01257 0x00,
01258 /* U+1E70 */ 0x1E71,
01259 0x00,
01260 /* U+1E72 */ 0x1E73,
01261 0x00,
01262 /* U+1E74 */ 0x1E75,
01263 0x00,
01264 /* U+1E76 */ 0x1E77,
01265 0x00,
01266 /* U+1E78 */ 0x1E79,
01267 0x00,
01268 /* U+1E7A */ 0x1E7B,
01269 0x00,
01270 /* U+1E7C */ 0x1E7D,
01271 0x00,
01272 /* U+1E7E */ 0x1E7F,
01273 0x00,
01274 /* U+1E80 */ 0x1E81,
01275 0x00,
01276 /* U+1E82 */ 0x1E83,
01277 0x00,
01278 /* U+1E84 */ 0x1E85,
01279 0x00,
01280 /* U+1E86 */ 0x1E87,
01281 0x00,
01282 /* U+1E88 */ 0x1E89,
01283 0x00,
01284 /* U+1E8A */ 0x1E8B,
01285 0x00,
01286 /* U+1E8C */ 0x1E8D,
01287 0x00,
01288 /* U+1E8E */ 0x1E8F,
01289 0x00,
01290 /* U+1E90 */ 0x1E91,
01291 0x00,
01292 /* U+1E92 */ 0x1E93,
01293 0x00,
01294 /* U+1E94 */ 0x1E95,
01295 0x00,
01296 0x00,
01297 0x00,
01298 0x00,
01299 0x00,
01300 0x00,
01301 0x00,
01302 0x00,
01303 0x00,
01304 0x00,
01305 0x00,
01306 /* U+1EA0 */ 0x1EA1,
01307 0x00,
01308 /* U+1EA2 */ 0x1EA3,
01309 0x00,
01310 /* U+1EA4 */ 0x1EA5,
01311 0x00,
01312 /* U+1EA6 */ 0x1EA7,
01313 0x00,
01314 /* U+1EA8 */ 0x1EA9,
01315 0x00,
01316 /* U+1EAA */ 0x1EAB,
01317 0x00,
01318 /* U+1EAC */ 0x1EAD,
01319 0x00,
01320 /* U+1EAE */ 0x1EAF,
01321 0x00,
01322 /* U+1EB0 */ 0x1EB1,
01323 0x00,
01324 /* U+1EB2 */ 0x1EB3,
01325 0x00,
01326 /* U+1EB4 */ 0x1EB5,
01327 0x00,
01328 /* U+1EB6 */ 0x1EB7,
01329 0x00,
01330 /* U+1EB8 */ 0x1EB9,
01331 0x00,
01332 /* U+1EBA */ 0x1EBB,
01333 0x00,
01334 /* U+1EBC */ 0x1EBD,
```

```
01335 0x00,
01336 /* U+1EBE */ 0x1EBF,
01337 0x00,
01338 /* U+1EC0 */ 0x1EC1,
01339 0x00,
01340 /* U+1EC2 */ 0x1EC3,
01341 0x00,
01342 /* U+1EC4 */ 0x1EC5,
01343 0x00,
01344 /* U+1EC6 */ 0x1EC7,
01345 0x00,
01346 /* U+1EC8 */ 0x1EC9,
01347 0x00,
01348 /* U+1ECA */ 0x1ECB,
01349 0x00,
01350 /* U+1ECC */ 0x1ECD,
01351 0x00,
01352 /* U+1ECE */ 0x1ECF,
01353 0x00,
01354 /* U+1ED0 */ 0x1ED1,
01355 0x00,
01356 /* U+1ED2 */ 0x1ED3,
01357 0x00,
01358 /* U+1ED4 */ 0x1ED5,
01359 0x00,
01360 /* U+1ED6 */ 0x1ED7,
01361 0x00,
01362 /* U+1ED8 */ 0x1ED9,
01363 0x00,
01364 /* U+1EDA */ 0x1EDB,
01365 0x00,
01366 /* U+1EDC */ 0x1EDD,
01367 0x00,
01368 /* U+1EDE */ 0x1EDF,
01369 0x00,
01370 /* U+1EE0 */ 0x1EE1,
01371 0x00,
01372 /* U+1EE2 */ 0x1EE3,
01373 0x00,
01374 /* U+1EE4 */ 0x1EE5,
01375 0x00,
01376 /* U+1EE6 */ 0x1EE7,
01377 0x00,
01378 /* U+1EE8 */ 0x1EE9,
01379 0x00,
01380 /* U+1EEA */ 0x1EEB,
01381 0x00,
01382 /* U+1EEC */ 0x1EED,
01383 0x00,
01384 /* U+1EEE */ 0x1EEF,
01385 0x00,
01386 /* U+1EF0 */ 0x1EF1,
01387 0x00,
01388 /* U+1EF2 */ 0x1EF3,
01389 0x00,
01390 /* U+1EF4 */ 0x1EF5,
01391 0x00,
01392 /* U+1EF6 */ 0x1EF7,
01393 0x00,
01394 /* U+1EF8 */ 0x1EF9,
01395 0x00,
01396 0x00,
01397 0x00,
01398 0x00,
01399 0x00,
01400 0x00,
01401 0x00,
01402 0x00,
01403 0x00,
01404 0x00,
01405 0x00,
01406 0x00,
01407 0x00,
01408 0x00,
01409 0x00,
01410 /* U+1F08 */ 0x1F00,
01411 /* U+1F09 */ 0x1F01,
01412 /* U+1F0A */ 0x1F02,
01413 /* U+1F0B */ 0x1F03,
01414 /* U+1F0C */ 0x1F04,
01415 /* U+1F0D */ 0x1F05,
01416 /* U+1F0E */ 0x1F06,
01417 /* U+1F0F */ 0x1F07,
01418 0x00,
01419 0x00,
01420 0x00,
01421 0x00,
```

```
01422 0x00,
01423 0x00,
01424 0x00,
01425 0x00,
01426 /* U+1F18 */ 0x1F10,
01427 /* U+1F19 */ 0x1F11,
01428 /* U+1F1A */ 0x1F12,
01429 /* U+1F1B */ 0x1F13,
01430 /* U+1F1C */ 0x1F14,
01431 /* U+1F1D */ 0x1F15,
01432 0x00,
01433 0x00,
01434 0x00,
01435 0x00,
01436 0x00,
01437 0x00,
01438 0x00,
01439 0x00,
01440 0x00,
01441 0x00,
01442 /* U+1F28 */ 0x1F20,
01443 /* U+1F29 */ 0x1F21,
01444 /* U+1F2A */ 0x1F22,
01445 /* U+1F2B */ 0x1F23,
01446 /* U+1F2C */ 0x1F24,
01447 /* U+1F2D */ 0x1F25,
01448 /* U+1F2E */ 0x1F26,
01449 /* U+1F2F */ 0x1F27,
01450 0x00,
01451 0x00,
01452 0x00,
01453 0x00,
01454 0x00,
01455 0x00,
01456 0x00,
01457 0x00,
01458 /* U+1F38 */ 0x1F30,
01459 /* U+1F39 */ 0x1F31,
01460 /* U+1F3A */ 0x1F32,
01461 /* U+1F3B */ 0x1F33,
01462 /* U+1F3C */ 0x1F34,
01463 /* U+1F3D */ 0x1F35,
01464 /* U+1F3E */ 0x1F36,
01465 /* U+1F3F */ 0x1F37,
01466 0x00,
01467 0x00,
01468 0x00,
01469 0x00,
01470 0x00,
01471 0x00,
01472 0x00,
01473 0x00,
01474 /* U+1F48 */ 0x1F40,
01475 /* U+1F49 */ 0x1F41,
01476 /* U+1F4A */ 0x1F42,
01477 /* U+1F4B */ 0x1F43,
01478 /* U+1F4C */ 0x1F44,
01479 /* U+1F4D */ 0x1F45,
01480 0x00,
01481 0x00,
01482 0x00,
01483 0x00,
01484 0x00,
01485 0x00,
01486 0x00,
01487 0x00,
01488 0x00,
01489 0x00,
01490 0x00,
01491 /* U+1F59 */ 0x1F51,
01492 0x00,
01493 /* U+1F5B */ 0x1F53,
01494 0x00,
01495 /* U+1F5D */ 0x1F55,
01496 0x00,
01497 /* U+1F5F */ 0x1F57,
01498 0x00,
01499 0x00,
01500 0x00,
01501 0x00,
01502 0x00,
01503 0x00,
01504 0x00,
01505 0x00,
01506 /* U+1F68 */ 0x1F60,
01507 /* U+1F69 */ 0x1F61,
01508 /* U+1F6A */ 0x1F62,
```

```
01509 /* U+1F6B */ 0x1F63,
01510 /* U+1F6C */ 0x1F64,
01511 /* U+1F6D */ 0x1F65,
01512 /* U+1F6E */ 0x1F66,
01513 /* U+1F6F */ 0x1F67,
01514 0x00,
01515 0x00,
01516 0x00,
01517 0x00,
01518 0x00,
01519 0x00,
01520 0x00,
01521 0x00,
01522 0x00,
01523 0x00,
01524 0x00,
01525 0x00,
01526 0x00,
01527 0x00,
01528 0x00,
01529 0x00,
01530 0x00,
01531 0x00,
01532 0x00,
01533 0x00,
01534 0x00,
01535 0x00,
01536 0x00,
01537 0x00,
01538 /* U+1F88 */ 0x0,
01539 /* U+1F89 */ 0x0,
01540 /* U+1F8A */ 0x0,
01541 /* U+1F8B */ 0x0,
01542 /* U+1F8C */ 0x0,
01543 /* U+1F8D */ 0x0,
01544 /* U+1F8E */ 0x0,
01545 /* U+1F8F */ 0x0,
01546 0x00,
01547 0x00,
01548 0x00,
01549 0x00,
01550 0x00,
01551 0x00,
01552 0x00,
01553 0x00,
01554 /* U+1F98 */ 0x0,
01555 /* U+1F99 */ 0x0,
01556 /* U+1F9A */ 0x0,
01557 /* U+1F9B */ 0x0,
01558 /* U+1F9C */ 0x0,
01559 /* U+1F9D */ 0x0,
01560 /* U+1F9E */ 0x0,
01561 /* U+1F9F */ 0x0,
01562 0x00,
01563 0x00,
01564 0x00,
01565 0x00,
01566 0x00,
01567 0x00,
01568 0x00,
01569 0x00,
01570 /* U+1FA8 */ 0x0,
01571 /* U+1FA9 */ 0x0,
01572 /* U+1FAA */ 0x0,
01573 /* U+1FAB */ 0x0,
01574 /* U+1FAC */ 0x0,
01575 /* U+1FAD */ 0x0,
01576 /* U+1FAE */ 0x0,
01577 /* U+1FAF */ 0x0,
01578 0x00,
01579 0x00,
01580 0x00,
01581 0x00,
01582 0x00,
01583 0x00,
01584 0x00,
01585 0x00,
01586 /* U+1FB8 */ 0x1FB0,
01587 /* U+1FB9 */ 0x1FB1,
01588 /* U+1FBA */ 0x1F70,
01589 /* U+1FBB */ 0x1F71,
01590 /* U+1FBC */ 0x0,
01591 0x00,
01592 0x00,
01593 0x00,
01594 0x00,
01595 0x00,
```

```
01596 0x00,
01597 0x00,
01598 0x00,
01599 0x00,
01600 0x00,
01601 0x00,
01602 /* U+1FC8 */ 0x1F72,
01603 /* U+1FC9 */ 0x1F73,
01604 /* U+1FCA */ 0x1F74,
01605 /* U+1FCB */ 0x1F75,
01606 /* U+1FCC */ 0x0,
01607 0x00,
01608 0x00,
01609 0x00,
01610 0x00,
01611 0x00,
01612 0x00,
01613 0x00,
01614 0x00,
01615 0x00,
01616 0x00,
01617 0x00,
01618 /* U+1FD8 */ 0x1FD0,
01619 /* U+1FD9 */ 0x1FD1,
01620 /* U+1FDA */ 0x1F76,
01621 /* U+1FDB */ 0x1F77,
01622 0x00,
01623 0x00,
01624 0x00,
01625 0x00,
01626 0x00,
01627 0x00,
01628 0x00,
01629 0x00,
01630 0x00,
01631 0x00,
01632 0x00,
01633 0x00,
01634 /* U+1FE8 */ 0x1FE0,
01635 /* U+1FE9 */ 0x1FE1,
01636 /* U+1FEA */ 0x1F7A,
01637 /* U+1FEB */ 0x1F7B,
01638 /* U+1FEC */ 0x1FE5,
01639 0x00,
01640 0x00,
01641 0x00,
01642 0x00,
01643 0x00,
01644 0x00,
01645 0x00,
01646 0x00,
01647 0x00,
01648 0x00,
01649 0x00,
01650 /* U+1FF8 */ 0x1F78,
01651 /* U+1FF9 */ 0x1F79,
01652 /* U+1FFA */ 0x1F7C,
01653 /* U+1FFB */ 0x1F7D,
01654 /* U+1FFC */ 0x0,
01655 };
01656
01657 static const unsigned short ucs_table_2102[] = {
01658 /* U+2102 */ 0x0,
01659 0x00,
01660 0x00,
01661 0x00,
01662 0x00,
01663 0x00,
01664 0x00,
01665 0x00,
01666 0x00,
01667 /* U+210B */ 0x0,
01668 /* U+210C */ 0x0,
01669 /* U+210D */ 0x0,
01670 0x00,
01671 0x00,
01672 /* U+2110 */ 0x0,
01673 /* U+2111 */ 0x0,
01674 /* U+2112 */ 0x2113,
01675 0x00,
01676 0x00,
01677 /* U+2115 */ 0x0,
01678 0x00,
01679 0x00,
01680 /* U+2118 */ 0x0,
01681 /* U+2119 */ 0x0,
01682 /* U+211A */ 0x0,
```

```
01683 /* U+211B */ 0x0,
01684 /* U+211C */ 0x0,
01685 /* U+211D */ 0x0,
01686 0x00,
01687 0x00,
01688 0x00,
01689 0x00,
01690 0x00,
01691 0x00,
01692 /* U+2124 */ 0x0,
01693 0x00,
01694 0x00,
01695 0x00,
01696 /* U+2128 */ 0x0,
01697 0x00,
01698 0x00,
01699 0x00,
01700 /* U+212C */ 0x0,
01701 /* U+212D */ 0x0,
01702 0x00,
01703 0x00,
01704 /* U+2130 */ 0x212F,
01705 /* U+2131 */ 0x0,
01706 /* U+2132 */ 0x0,
01707 /* U+2133 */ 0x0,
01708 };
01709
01710 static const unsigned short ucs_table_24B6[] = {
01711 /* U+24B6 */ 0x24D0,
01712 /* U+24B7 */ 0x24D1,
01713 /* U+24B8 */ 0x24D2,
01714 /* U+24B9 */ 0x24D3,
01715 /* U+24BA */ 0x24D4,
01716 /* U+24BB */ 0x24D5,
01717 /* U+24BC */ 0x24D6,
01718 /* U+24BD */ 0x24D7,
01719 /* U+24BE */ 0x24D8,
01720 /* U+24BF */ 0x24D9,
01721 /* U+24C0 */ 0x24DA,
01722 /* U+24C1 */ 0x24DB,
01723 /* U+24C2 */ 0x24DC,
01724 /* U+24C3 */ 0x24DD,
01725 /* U+24C4 */ 0x24DE,
01726 /* U+24C5 */ 0x24DF,
01727 /* U+24C6 */ 0x24E0,
01728 /* U+24C7 */ 0x24E1,
01729 /* U+24C8 */ 0x24E2,
01730 /* U+24C9 */ 0x24E3,
01731 /* U+24CA */ 0x24E4,
01732 /* U+24CB */ 0x24E5,
01733 /* U+24CC */ 0x24E6,
01734 /* U+24CD */ 0x24E7,
01735 /* U+24CE */ 0x24E8,
01736 /* U+24CF */ 0x24E9,
01737 };
01738
01739 static const unsigned short ucs_table_33CE[] = {
01740 /* U+33CE */ 0x0,
01741 };
01742
01743 static const unsigned short ucs_table_FF21[] = {
01744 /* U+FF21 */ 0xFF41,
01745 /* U+FF22 */ 0xFF42,
01746 /* U+FF23 */ 0xFF43,
01747 /* U+FF24 */ 0xFF44,
01748 /* U+FF25 */ 0xFF45,
01749 /* U+FF26 */ 0xFF46,
01750 /* U+FF27 */ 0xFF47,
01751 /* U+FF28 */ 0xFF48,
01752 /* U+FF29 */ 0xFF49,
01753 /* U+FF2A */ 0xFF4A,
01754 /* U+FF2B */ 0xFF4B,
01755 /* U+FF2C */ 0xFF4C,
01756 /* U+FF2D */ 0xFF4D,
01757 /* U+FF2E */ 0xFF4E,
01758 /* U+FF2F */ 0xFF4F,
01759 /* U+FF30 */ 0xFF50,
01760 /* U+FF31 */ 0xFF51,
01761 /* U+FF32 */ 0xFF52,
01762 /* U+FF33 */ 0xFF53,
01763 /* U+FF34 */ 0xFF54,
01764 /* U+FF35 */ 0xFF55,
01765 /* U+FF36 */ 0xFF56,
01766 /* U+FF37 */ 0xFF57,
01767 /* U+FF38 */ 0xFF58,
01768 /* U+FF39 */ 0xFF59,
01769 /* U+FF3A */ 0xFF5A,
```



```
01770 };
```

12.253 dingbats_.h

```
00001 /* dingbats */
00002
00003 static const char unicode_to_dingbats_lb_0020[] = {
00004 /* U+0020 */ 0x20,
00005 0x00,
00006 0x00,
00007 0x00,
00008 0x00,
00009 0x00,
00010 0x00,
00011 0x00,
00012 0x00,
00013 0x00,
00014 0x00,
00015 0x00,
00016 0x00,
00017 0x00,
00018 0x00,
00019 0x00,
00020 0x00,
00021 0x00,
00022 0x00,
00023 0x00,
00024 0x00,
00025 0x00,
00026 0x00,
00027 0x00,
00028 0x00,
00029 0x00,
00030 0x00,
00031 0x00,
00032 0x00,
00033 0x00,
00034 0x00,
00035 0x00,
00036 0x00,
00037 0x00,
00038 0x00,
00039 0x00,
00040 0x00,
00041 0x00,
00042 0x00,
00043 0x00,
00044 0x00,
00045 0x00,
00046 0x00,
00047 0x00,
00048 0x00,
00049 0x00,
00050 0x00,
00051 0x00,
00052 0x00,
00053 0x00,
00054 0x00,
00055 0x00,
00056 0x00,
00057 0x00,
00058 0x00,
00059 0x00,
00060 0x00,
00061 0x00,
00062 0x00,
00063 0x00,
00064 0x00,
00065 0x00,
00066 0x00,
00067 0x00,
00068 0x00,
00069 0x00,
00070 0x00,
00071 0x00,
00072 0x00,
00073 0x00,
00074 0x00,
00075 0x00,
00076 0x00,
00077 0x00,
00078 0x00,
00079 0x00,
00080 0x00,
00081 0x00,
```

```
00082 0x00,
00083 0x00,
00084 0x00,
00085 0x00,
00086 0x00,
00087 0x00,
00088 0x00,
00089 0x00,
00090 0x00,
00091 0x00,
00092 0x00,
00093 0x00,
00094 0x00,
00095 0x00,
00096 0x00,
00097 0x00,
00098 0x00,
00099 0x00,
00100 0x00,
00101 0x00,
00102 0x00,
00103 0x00,
00104 0x00,
00105 0x00,
00106 0x00,
00107 0x00,
00108 0x00,
00109 0x00,
00110 0x00,
00111 0x00,
00112 0x00,
00113 0x00,
00114 0x00,
00115 0x00,
00116 0x00,
00117 0x00,
00118 0x00,
00119 0x00,
00120 0x00,
00121 0x00,
00122 0x00,
00123 0x00,
00124 0x00,
00125 0x00,
00126 0x00,
00127 0x00,
00128 0x00,
00129 0x00,
00130 0x00,
00131 0x00,
00132 /* U+00A0 */ 0x20,
00133 };
00134
00135 static const char unicode_to_dingbats_1b_2192[] = {
00136 /* U+2192 */ (char)0xD5,
00137 0x00,
00138 /* U+2194 */ (char)0xD6,
00139 /* U+2195 */ (char)0xD7,
00140 };
00141
00142 static const char unicode_to_dingbats_1b_2460[] = {
00143 /* U+2460 */ (char)0xAC,
00144 /* U+2461 */ (char)0xAD,
00145 /* U+2462 */ (char)0xAE,
00146 /* U+2463 */ (char)0xAF,
00147 /* U+2464 */ (char)0xB0,
00148 /* U+2465 */ (char)0xB1,
00149 /* U+2466 */ (char)0xB2,
00150 /* U+2467 */ (char)0xB3,
00151 /* U+2468 */ (char)0xB4,
00152 /* U+2469 */ (char)0xB5,
00153 };
00154
00155 static const char unicode_to_dingbats_1b_25A0[] = {
00156 /* U+25A0 */ 0x6E,
00157 0x00,
00158 0x00,
00159 0x00,
00160 0x00,
00161 0x00,
00162 0x00,
00163 0x00,
00164 0x00,
00165 0x00,
00166 0x00,
00167 0x00,
00168 0x00,
```

```
00169 0x00,
00170 0x00,
00171 0x00,
00172 0x00,
00173 0x00,
00174 /* U+25B2 */ 0x73,
00175 0x00,
00176 0x00,
00177 0x00,
00178 0x00,
00179 0x00,
00180 0x00,
00181 0x00,
00182 0x00,
00183 0x00,
00184 /* U+25BC */ 0x74,
00185 0x00,
00186 0x00,
00187 0x00,
00188 0x00,
00189 0x00,
00190 0x00,
00191 0x00,
00192 0x00,
00193 0x00,
00194 /* U+25C6 */ 0x75,
00195 0x00,
00196 0x00,
00197 0x00,
00198 0x00,
00199 0x00,
00200 0x00,
00201 0x00,
00202 0x00,
00203 /* U+25CF */ 0x6C,
00204 0x00,
00205 0x00,
00206 0x00,
00207 0x00,
00208 0x00,
00209 0x00,
00210 0x00,
00211 /* U+25D7 */ 0x77,
00212 0x00,
00213 0x00,
00214 0x00,
00215 0x00,
00216 0x00,
00217 0x00,
00218 0x00,
00219 0x00,
00220 0x00,
00221 0x00,
00222 0x00,
00223 0x00,
00224 0x00,
00225 0x00,
00226 0x00,
00227 0x00,
00228 0x00,
00229 0x00,
00230 0x00,
00231 0x00,
00232 0x00,
00233 0x00,
00234 0x00,
00235 0x00,
00236 0x00,
00237 0x00,
00238 0x00,
00239 0x00,
00240 0x00,
00241 0x00,
00242 0x00,
00243 0x00,
00244 0x00,
00245 0x00,
00246 0x00,
00247 0x00,
00248 0x00,
00249 0x00,
00250 0x00,
00251 0x00,
00252 0x00,
00253 0x00,
00254 0x00,
00255 0x00,
```

```
00256 0x00,
00257 /* U+2605 */ 0x48,
00258 0x00,
00259 0x00,
00260 0x00,
00261 0x00,
00262 0x00,
00263 0x00,
00264 0x00,
00265 0x00,
00266 /* U+260E */ 0x25,
00267 0x00,
00268 0x00,
00269 0x00,
00270 0x00,
00271 0x00,
00272 0x00,
00273 0x00,
00274 0x00,
00275 0x00,
00276 0x00,
00277 0x00,
00278 0x00,
00279 /* U+261B */ 0x2A,
00280 0x00,
00281 0x00,
00282 /* U+261E */ 0x2B,
00283 0x00,
00284 0x00,
00285 0x00,
00286 0x00,
00287 0x00,
00288 0x00,
00289 0x00,
00290 0x00,
00291 0x00,
00292 0x00,
00293 0x00,
00294 0x00,
00295 0x00,
00296 0x00,
00297 0x00,
00298 0x00,
00299 0x00,
00300 0x00,
00301 0x00,
00302 0x00,
00303 0x00,
00304 0x00,
00305 0x00,
00306 0x00,
00307 0x00,
00308 0x00,
00309 0x00,
00310 0x00,
00311 0x00,
00312 0x00,
00313 0x00,
00314 0x00,
00315 0x00,
00316 0x00,
00317 0x00,
00318 0x00,
00319 0x00,
00320 0x00,
00321 0x00,
00322 0x00,
00323 0x00,
00324 0x00,
00325 0x00,
00326 0x00,
00327 0x00,
00328 0x00,
00329 0x00,
00330 0x00,
00331 0x00,
00332 0x00,
00333 0x00,
00334 0x00,
00335 0x00,
00336 0x00,
00337 0x00,
00338 0x00,
00339 0x00,
00340 0x00,
00341 0x00,
00342 0x00,
```

```
00343 0x00,
00344 0x00,
00345 0x00,
00346 0x00,
00347 0x00,
00348 /* U+2660 */ (char)0xAB,
00349 0x00,
00350 0x00,
00351 /* U+2663 */ (char)0xA8,
00352 0x00,
00353 /* U+2665 */ (char)0xAA,
00354 /* U+2666 */ (char)0xA9,
00355 };
00356
00357 static const char unicode_to_dingbats_lb_2701[] = {
00358 /* U+2701 */ 0x21,
00359 /* U+2702 */ 0x22,
00360 /* U+2703 */ 0x23,
00361 /* U+2704 */ 0x24,
00362 0x00,
00363 /* U+2706 */ 0x26,
00364 /* U+2707 */ 0x27,
00365 /* U+2708 */ 0x28,
00366 /* U+2709 */ 0x29,
00367 0x00,
00368 0x00,
00369 /* U+270C */ 0x2C,
00370 /* U+270D */ 0x2D,
00371 /* U+270E */ 0x2E,
00372 /* U+270F */ 0x2F,
00373 /* U+2710 */ 0x30,
00374 /* U+2711 */ 0x31,
00375 /* U+2712 */ 0x32,
00376 /* U+2713 */ 0x33,
00377 /* U+2714 */ 0x34,
00378 /* U+2715 */ 0x35,
00379 /* U+2716 */ 0x36,
00380 /* U+2717 */ 0x37,
00381 /* U+2718 */ 0x38,
00382 /* U+2719 */ 0x39,
00383 /* U+271A */ 0x3A,
00384 /* U+271B */ 0x3B,
00385 /* U+271C */ 0x3C,
00386 /* U+271D */ 0x3D,
00387 /* U+271E */ 0x3E,
00388 /* U+271F */ 0x3F,
00389 /* U+2720 */ 0x40,
00390 /* U+2721 */ 0x41,
00391 /* U+2722 */ 0x42,
00392 /* U+2723 */ 0x43,
00393 /* U+2724 */ 0x44,
00394 /* U+2725 */ 0x45,
00395 /* U+2726 */ 0x46,
00396 /* U+2727 */ 0x47,
00397 0x00,
00398 /* U+2729 */ 0x49,
00399 /* U+272A */ 0x4A,
00400 /* U+272B */ 0x4B,
00401 /* U+272C */ 0x4C,
00402 /* U+272D */ 0x4D,
00403 /* U+272E */ 0x4E,
00404 /* U+272F */ 0x4F,
00405 /* U+2730 */ 0x50,
00406 /* U+2731 */ 0x51,
00407 /* U+2732 */ 0x52,
00408 /* U+2733 */ 0x53,
00409 /* U+2734 */ 0x54,
00410 /* U+2735 */ 0x55,
00411 /* U+2736 */ 0x56,
00412 /* U+2737 */ 0x57,
00413 /* U+2738 */ 0x58,
00414 /* U+2739 */ 0x59,
00415 /* U+273A */ 0x5A,
00416 /* U+273B */ 0x5B,
00417 /* U+273C */ 0x5C,
00418 /* U+273D */ 0x5D,
00419 /* U+273E */ 0x5E,
00420 /* U+273F */ 0x5F,
00421 /* U+2740 */ 0x60,
00422 /* U+2741 */ 0x61,
00423 /* U+2742 */ 0x62,
00424 /* U+2743 */ 0x63,
00425 /* U+2744 */ 0x64,
00426 /* U+2745 */ 0x65,
00427 /* U+2746 */ 0x66,
00428 /* U+2747 */ 0x67,
00429 /* U+2748 */ 0x68,
```

```
00430 /* U+2749 */ 0x69,
00431 /* U+274A */ 0x6A,
00432 /* U+274B */ 0x6B,
00433 0x00,
00434 /* U+274D */ 0x6D,
00435 0x00,
00436 /* U+274F */ 0x6F,
00437 /* U+2750 */ 0x70,
00438 /* U+2751 */ 0x71,
00439 /* U+2752 */ 0x72,
00440 0x00,
00441 0x00,
00442 0x00,
00443 /* U+2756 */ 0x76,
00444 0x00,
00445 /* U+2758 */ 0x78,
00446 /* U+2759 */ 0x79,
00447 /* U+275A */ 0x7A,
00448 /* U+275B */ 0x7B,
00449 /* U+275C */ 0x7C,
00450 /* U+275D */ 0x7D,
00451 /* U+275E */ 0x7E,
00452 0x00,
00453 0x00,
00454 /* U+2761 */ (char) 0xA1,
00455 /* U+2762 */ (char) 0xA2,
00456 /* U+2763 */ (char) 0xA3,
00457 /* U+2764 */ (char) 0xA4,
00458 /* U+2765 */ (char) 0xA5,
00459 /* U+2766 */ (char) 0xA6,
00460 /* U+2767 */ (char) 0xA7,
00461 0x00,
00462 0x00,
00463 0x00,
00464 0x00,
00465 0x00,
00466 0x00,
00467 0x00,
00468 0x00,
00469 0x00,
00470 0x00,
00471 0x00,
00472 0x00,
00473 0x00,
00474 0x00,
00475 /* U+2776 */ (char) 0xB6,
00476 /* U+2777 */ (char) 0xB7,
00477 /* U+2778 */ (char) 0xB8,
00478 /* U+2779 */ (char) 0xB9,
00479 /* U+277A */ (char) 0xBA,
00480 /* U+277B */ (char) 0xBB,
00481 /* U+277C */ (char) 0xBC,
00482 /* U+277D */ (char) 0xBD,
00483 /* U+277E */ (char) 0xBE,
00484 /* U+277F */ (char) 0xBF,
00485 /* U+2780 */ (char) 0xC0,
00486 /* U+2781 */ (char) 0xC1,
00487 /* U+2782 */ (char) 0xC2,
00488 /* U+2783 */ (char) 0xC3,
00489 /* U+2784 */ (char) 0xC4,
00490 /* U+2785 */ (char) 0xC5,
00491 /* U+2786 */ (char) 0xC6,
00492 /* U+2787 */ (char) 0xC7,
00493 /* U+2788 */ (char) 0xC8,
00494 /* U+2789 */ (char) 0xC9,
00495 /* U+278A */ (char) 0xCA,
00496 /* U+278B */ (char) 0xCB,
00497 /* U+278C */ (char) 0xCC,
00498 /* U+278D */ (char) 0xCD,
00499 /* U+278E */ (char) 0xCE,
00500 /* U+278F */ (char) 0xCF,
00501 /* U+2790 */ (char) 0xD0,
00502 /* U+2791 */ (char) 0xD1,
00503 /* U+2792 */ (char) 0xD2,
00504 /* U+2793 */ (char) 0xD3,
00505 /* U+2794 */ (char) 0xD4,
00506 0x00,
00507 0x00,
00508 0x00,
00509 /* U+2798 */ (char) 0xD8,
00510 /* U+2799 */ (char) 0xD9,
00511 /* U+279A */ (char) 0xDA,
00512 /* U+279B */ (char) 0xDB,
00513 /* U+279C */ (char) 0xDC,
00514 /* U+279D */ (char) 0xDD,
00515 /* U+279E */ (char) 0xDE,
00516 /* U+279F */ (char) 0xDF,
```

```
00517 /* U+27A0 */ (char)0xE0,
00518 /* U+27A1 */ (char)0xE1,
00519 /* U+27A2 */ (char)0xE2,
00520 /* U+27A3 */ (char)0xE3,
00521 /* U+27A4 */ (char)0xE4,
00522 /* U+27A5 */ (char)0xE5,
00523 /* U+27A6 */ (char)0xE6,
00524 /* U+27A7 */ (char)0xE7,
00525 /* U+27A8 */ (char)0xE8,
00526 /* U+27A9 */ (char)0xE9,
00527 /* U+27AA */ (char)0xEA,
00528 /* U+27AB */ (char)0xEB,
00529 /* U+27AC */ (char)0xEC,
00530 /* U+27AD */ (char)0xED,
00531 /* U+27AE */ (char)0xEE,
00532 /* U+27AF */ (char)0xEF,
00533 0x00,
00534 /* U+27B1 */ (char)0xF1,
00535 /* U+27B2 */ (char)0xF2,
00536 /* U+27B3 */ (char)0xF3,
00537 /* U+27B4 */ (char)0xF4,
00538 /* U+27B5 */ (char)0xF5,
00539 /* U+27B6 */ (char)0xF6,
00540 /* U+27B7 */ (char)0xF7,
00541 /* U+27B8 */ (char)0xF8,
00542 /* U+27B9 */ (char)0xF9,
00543 /* U+27BA */ (char)0xFA,
00544 /* U+27BB */ (char)0xFB,
00545 /* U+27BC */ (char)0xFC,
00546 /* U+27BD */ (char)0xFD,
00547 /* U+27BE */ (char)0xFE,
00548 };
00549
00550 static const char unicode_to_dingbats_lb_F8D7[] = {
00551 /* U+F8D7 */ (char)0x80,
00552 /* U+F8D8 */ (char)0x81,
00553 /* U+F8D9 */ (char)0x82,
00554 /* U+F8DA */ (char)0x83,
00555 /* U+F8DB */ (char)0x84,
00556 /* U+F8DC */ (char)0x85,
00557 /* U+F8DD */ (char)0x86,
00558 /* U+F8DE */ (char)0x87,
00559 /* U+F8DF */ (char)0x88,
00560 /* U+F8E0 */ (char)0x89,
00561 /* U+F8E1 */ (char)0x8A,
00562 /* U+F8E2 */ (char)0x8B,
00563 /* U+F8E3 */ (char)0x8C,
00564 /* U+F8E4 */ (char)0x8D,
00565 };
```

12.254 spacing.h

```
00001 /* spacing */
00002
00003 static const unsigned short ucs_table_0300[] = {
00004 /* U+0300 */ 0x0060,
00005 /* U+0301 */ 0x00B4,
00006 /* U+0302 */ 0x005E,
00007 /* U+0303 */ 0x02DC,
00008 /* U+0304 */ 0x00AF,
00009 /* U+0305 */ 0x203E,
00010 /* U+0306 */ 0x02D8,
00011 /* U+0307 */ 0x02D9,
00012 /* U+0308 */ 0x00A8,
00013 /* U+0309 */ 0x0309,
00014 /* U+030A */ 0x02DA,
00015 /* U+030B */ 0x02DD,
00016 /* U+030C */ 0x030C,
00017 /* U+030D */ 0x030D,
00018 /* U+030E */ 0x030E,
00019 /* U+030F */ 0x030F,
00020 /* U+0310 */ 0x0310,
00021 /* U+0311 */ 0x0311,
00022 /* U+0312 */ 0x0312,
00023 /* U+0313 */ 0x1FBD,
00024 /* U+0314 */ 0x1FFE,
00025 /* U+0315 */ 0x0315,
00026 /* U+0316 */ 0x0316,
00027 /* U+0317 */ 0x0317,
00028 /* U+0318 */ 0x0318,
00029 /* U+0319 */ 0x0319,
00030 /* U+031A */ 0x031A,
00031 /* U+031B */ 0x031B,
00032 /* U+031C */ 0x031C,
00033 /* U+031D */ 0x031D,
```

```
00034 /* U+031E */ 0x031E,
00035 /* U+031F */ 0x031F,
00036 /* U+0320 */ 0x0320,
00037 /* U+0321 */ 0x0321,
00038 /* U+0322 */ 0x0322,
00039 /* U+0323 */ 0x0323,
00040 /* U+0324 */ 0x0324,
00041 /* U+0325 */ 0x0325,
00042 /* U+0326 */ 0x0326,
00043 /* U+0327 */ 0x00B8,
00044 /* U+0328 */ 0x02DB,
00045 /* U+0329 */ 0x0329,
00046 /* U+032A */ 0x032A,
00047 /* U+032B */ 0x032B,
00048 /* U+032C */ 0x032C,
00049 /* U+032D */ 0x032D,
00050 /* U+032E */ 0x032E,
00051 /* U+032F */ 0x032F,
00052 /* U+0330 */ 0x0330,
00053 /* U+0331 */ 0x0331,
00054 /* U+0332 */ 0x005F,
00055 /* U+0333 */ 0x2017,
00056 /* U+0334 */ 0x0334,
00057 /* U+0335 */ 0x0335,
00058 /* U+0336 */ 0x0336,
00059 /* U+0337 */ 0x0337,
00060 /* U+0338 */ 0x0338,
00061 /* U+0339 */ 0x0339,
00062 /* U+033A */ 0x033A,
00063 /* U+033B */ 0x033B,
00064 /* U+033C */ 0x033C,
00065 /* U+033D */ 0x033D,
00066 /* U+033E */ 0x033E,
00067 /* U+033F */ 0x033F,
00068 /* U+0340 */ 0x0340,
00069 /* U+0341 */ 0x0341,
00070 /* U+0342 */ 0x1FC0,
00071 /* U+0343 */ 0x0343,
00072 /* U+0344 */ 0x0344,
00073 /* U+0345 */ 0x037A,
00074 0x00,
00075 0x00,
00076 0x00,
00077 0x00,
00078 0x00,
00079 0x00,
00080 0x00,
00081 0x00,
00082 0x00,
00083 0x00,
00084 0x00,
00085 0x00,
00086 0x00,
00087 0x00,
00088 0x00,
00089 0x00,
00090 0x00,
00091 0x00,
00092 0x00,
00093 0x00,
00094 0x00,
00095 0x00,
00096 0x00,
00097 0x00,
00098 0x00,
00099 0x00,
00100 /* U+0360 */ 0x0360,
00101 /* U+0361 */ 0x0361,
00102 };
00103
00104 static const unsigned short ucs_table_0483[] = {
00105 /* U+0483 */ 0x0483,
00106 /* U+0484 */ 0x0484,
00107 /* U+0485 */ 0x0485,
00108 /* U+0486 */ 0x0486,
00109 };
00110
00111 static const unsigned short ucs_table_0591[] = {
00112 /* U+0591 */ 0x0591,
00113 /* U+0592 */ 0x0592,
00114 /* U+0593 */ 0x0593,
00115 /* U+0594 */ 0x0594,
00116 /* U+0595 */ 0x0595,
00117 /* U+0596 */ 0x0596,
00118 /* U+0597 */ 0x0597,
00119 /* U+0598 */ 0x0598,
00120 /* U+0599 */ 0x0599,
```



```
00121 /* U+059A */ 0x059A,
00122 /* U+059B */ 0x059B,
00123 /* U+059C */ 0x059C,
00124 /* U+059D */ 0x059D,
00125 /* U+059E */ 0x059E,
00126 /* U+059F */ 0x059F,
00127 /* U+05A0 */ 0x05A0,
00128 /* U+05A1 */ 0x05A1,
00129 0x00,
00130 /* U+05A3 */ 0x05A3,
00131 /* U+05A4 */ 0x05A4,
00132 /* U+05A5 */ 0x05A5,
00133 /* U+05A6 */ 0x05A6,
00134 /* U+05A7 */ 0x05A7,
00135 /* U+05A8 */ 0x05A8,
00136 /* U+05A9 */ 0x05A9,
00137 /* U+05AA */ 0x05AA,
00138 /* U+05AB */ 0x05AB,
00139 /* U+05AC */ 0x05AC,
00140 /* U+05AD */ 0x05AD,
00141 /* U+05AE */ 0x05AE,
00142 /* U+05AF */ 0x05AF,
00143 /* U+05B0 */ 0x05B0,
00144 /* U+05B1 */ 0x05B1,
00145 /* U+05B2 */ 0x05B2,
00146 /* U+05B3 */ 0x05B3,
00147 /* U+05B4 */ 0x05B4,
00148 /* U+05B5 */ 0x05B5,
00149 /* U+05B6 */ 0x05B6,
00150 /* U+05B7 */ 0x05B7,
00151 /* U+05B8 */ 0x05B8,
00152 /* U+05B9 */ 0x05B9,
00153 0x00,
00154 /* U+05BB */ 0x05BB,
00155 /* U+05BC */ 0x05BC,
00156 /* U+05BD */ 0x05BD,
00157 0x00,
00158 /* U+05BF */ 0x05BF,
00159 0x00,
00160 /* U+05C1 */ 0x05C1,
00161 /* U+05C2 */ 0x05C2,
00162 0x00,
00163 /* U+05C4 */ 0x05C4,
00164 };
00165
00166 static const unsigned short ucs_table_064B[] = {
00167 /* U+064B */ 0xFE70,
00168 /* U+064C */ 0xFE72,
00169 /* U+064D */ 0xFE74,
00170 /* U+064E */ 0xFE76,
00171 /* U+064F */ 0xFE78,
00172 /* U+0650 */ 0xFE7A,
00173 /* U+0651 */ 0xFE7C,
00174 /* U+0652 */ 0xFE7E,
00175 0x00,
00176 0x00,
00177 0x00,
00178 0x00,
00179 0x00,
00180 0x00,
00181 0x00,
00182 0x00,
00183 0x00,
00184 0x00,
00185 0x00,
00186 0x00,
00187 0x00,
00188 0x00,
00189 0x00,
00190 0x00,
00191 0x00,
00192 0x00,
00193 0x00,
00194 0x00,
00195 0x00,
00196 0x00,
00197 0x00,
00198 0x00,
00199 0x00,
00200 0x00,
00201 0x00,
00202 0x00,
00203 0x00,
00204 /* U+0670 */ 0x0670,
00205 0x00,
00206 0x00,
00207 0x00,
```

```
00208 0x00,
00209 0x00,
00210 0x00,
00211 0x00,
00212 0x00,
00213 0x00,
00214 0x00,
00215 0x00,
00216 0x00,
00217 0x00,
00218 0x00,
00219 0x00,
00220 0x00,
00221 0x00,
00222 0x00,
00223 0x00,
00224 0x00,
00225 0x00,
00226 0x00,
00227 0x00,
00228 0x00,
00229 0x00,
00230 0x00,
00231 0x00,
00232 0x00,
00233 0x00,
00234 0x00,
00235 0x00,
00236 0x00,
00237 0x00,
00238 0x00,
00239 0x00,
00240 0x00,
00241 0x00,
00242 0x00,
00243 0x00,
00244 0x00,
00245 0x00,
00246 0x00,
00247 0x00,
00248 0x00,
00249 0x00,
00250 0x00,
00251 0x00,
00252 0x00,
00253 0x00,
00254 0x00,
00255 0x00,
00256 0x00,
00257 0x00,
00258 0x00,
00259 0x00,
00260 0x00,
00261 0x00,
00262 0x00,
00263 0x00,
00264 0x00,
00265 0x00,
00266 0x00,
00267 0x00,
00268 0x00,
00269 0x00,
00270 0x00,
00271 0x00,
00272 0x00,
00273 0x00,
00274 0x00,
00275 0x00,
00276 0x00,
00277 0x00,
00278 0x00,
00279 0x00,
00280 0x00,
00281 0x00,
00282 0x00,
00283 0x00,
00284 0x00,
00285 0x00,
00286 0x00,
00287 0x00,
00288 0x00,
00289 0x00,
00290 0x00,
00291 0x00,
00292 0x00,
00293 0x00,
00294 0x00,
```

```
00295 0x00,
00296 0x00,
00297 0x00,
00298 0x00,
00299 0x00,
00300 0x00,
00301 0x00,
00302 0x00,
00303 0x00,
00304 0x00,
00305 0x00,
00306 /* U+06D6 */ 0x06D6,
00307 /* U+06D7 */ 0x06D7,
00308 /* U+06D8 */ 0x06D8,
00309 /* U+06D9 */ 0x06D9,
00310 /* U+06DA */ 0x06DA,
00311 /* U+06DB */ 0x06DB,
00312 /* U+06DC */ 0x06DC,
00313 0x00,
00314 0x00,
00315 /* U+06DF */ 0x06DF,
00316 /* U+06E0 */ 0x06E0,
00317 /* U+06E1 */ 0x06E1,
00318 /* U+06E2 */ 0x06E2,
00319 /* U+06E3 */ 0x06E3,
00320 /* U+06E4 */ 0x06E4,
00321 0x00,
00322 0x00,
00323 /* U+06E7 */ 0x06E7,
00324 /* U+06E8 */ 0x06E8,
00325 0x00,
00326 /* U+06EA */ 0x06EA,
00327 /* U+06EB */ 0x06EB,
00328 /* U+06EC */ 0x06EC,
00329 /* U+06ED */ 0x06ED,
00330 };
00331
00332 static const unsigned short ucs_table_0901[] = {
00333 /* U+0901 */ 0x0901,
00334 /* U+0902 */ 0x0902,
00335 0x00,
00336 0x00,
00337 0x00,
00338 0x00,
00339 0x00,
00340 0x00,
00341 0x00,
00342 0x00,
00343 0x00,
00344 0x00,
00345 0x00,
00346 0x00,
00347 0x00,
00348 0x00,
00349 0x00,
00350 0x00,
00351 0x00,
00352 0x00,
00353 0x00,
00354 0x00,
00355 0x00,
00356 0x00,
00357 0x00,
00358 0x00,
00359 0x00,
00360 0x00,
00361 0x00,
00362 0x00,
00363 0x00,
00364 0x00,
00365 0x00,
00366 0x00,
00367 0x00,
00368 0x00,
00369 0x00,
00370 0x00,
00371 0x00,
00372 0x00,
00373 0x00,
00374 0x00,
00375 0x00,
00376 0x00,
00377 0x00,
00378 0x00,
00379 0x00,
00380 0x00,
00381 0x00,
```

```
00382 0x00,
00383 0x00,
00384 0x00,
00385 0x00,
00386 0x00,
00387 0x00,
00388 0x00,
00389 0x00,
00390 0x00,
00391 0x00,
00392 /* U+093C */ 0x093C,
00393 0x00,
00394 0x00,
00395 0x00,
00396 0x00,
00397 /* U+0941 */ 0x0941,
00398 /* U+0942 */ 0x0942,
00399 /* U+0943 */ 0x0943,
00400 /* U+0944 */ 0x0944,
00401 /* U+0945 */ 0x0945,
00402 /* U+0946 */ 0x0946,
00403 /* U+0947 */ 0x0947,
00404 /* U+0948 */ 0x0948,
00405 0x00,
00406 0x00,
00407 0x00,
00408 0x00,
00409 /* U+094D */ 0x094D,
00410 0x00,
00411 0x00,
00412 0x00,
00413 /* U+0951 */ 0x0951,
00414 /* U+0952 */ 0x0952,
00415 /* U+0953 */ 0x0953,
00416 /* U+0954 */ 0x0954,
00417 0x00,
00418 0x00,
00419 0x00,
00420 0x00,
00421 0x00,
00422 0x00,
00423 0x00,
00424 0x00,
00425 0x00,
00426 0x00,
00427 0x00,
00428 0x00,
00429 0x00,
00430 /* U+0962 */ 0x0962,
00431 /* U+0963 */ 0x0963,
00432 0x00,
00433 0x00,
00434 0x00,
00435 0x00,
00436 0x00,
00437 0x00,
00438 0x00,
00439 0x00,
00440 0x00,
00441 0x00,
00442 0x00,
00443 0x00,
00444 0x00,
00445 0x00,
00446 0x00,
00447 0x00,
00448 0x00,
00449 0x00,
00450 0x00,
00451 0x00,
00452 0x00,
00453 0x00,
00454 0x00,
00455 0x00,
00456 0x00,
00457 0x00,
00458 0x00,
00459 0x00,
00460 0x00,
00461 /* U+0981 */ 0x0981,
00462 0x00,
00463 0x00,
00464 0x00,
00465 0x00,
00466 0x00,
00467 0x00,
00468 0x00,
```

```
00469 0x00,
00470 0x00,
00471 0x00,
00472 0x00,
00473 0x00,
00474 0x00,
00475 0x00,
00476 0x00,
00477 0x00,
00478 0x00,
00479 0x00,
00480 0x00,
00481 0x00,
00482 0x00,
00483 0x00,
00484 0x00,
00485 0x00,
00486 0x00,
00487 0x00,
00488 0x00,
00489 0x00,
00490 0x00,
00491 0x00,
00492 0x00,
00493 0x00,
00494 0x00,
00495 0x00,
00496 0x00,
00497 0x00,
00498 0x00,
00499 0x00,
00500 0x00,
00501 0x00,
00502 0x00,
00503 0x00,
00504 0x00,
00505 0x00,
00506 0x00,
00507 0x00,
00508 0x00,
00509 0x00,
00510 0x00,
00511 0x00,
00512 0x00,
00513 0x00,
00514 0x00,
00515 0x00,
00516 0x00,
00517 0x00,
00518 0x00,
00519 0x00,
00520 /* U+09BC */ 0x09BC,
00521 0x00,
00522 0x00,
00523 0x00,
00524 0x00,
00525 /* U+09C1 */ 0x09C1,
00526 /* U+09C2 */ 0x09C2,
00527 /* U+09C3 */ 0x09C3,
00528 /* U+09C4 */ 0x09C4,
00529 0x00,
00530 0x00,
00531 0x00,
00532 0x00,
00533 0x00,
00534 0x00,
00535 0x00,
00536 0x00,
00537 /* U+09CD */ 0x09CD,
00538 0x00,
00539 0x00,
00540 0x00,
00541 0x00,
00542 0x00,
00543 0x00,
00544 0x00,
00545 0x00,
00546 0x00,
00547 0x00,
00548 0x00,
00549 0x00,
00550 0x00,
00551 0x00,
00552 0x00,
00553 0x00,
00554 0x00,
00555 0x00,
```

```
00556 0x00,
00557 0x00,
00558 /* U+09E2 */ 0x09E2,
00559 /* U+09E3 */ 0x09E3,
00560 0x00,
00561 0x00,
00562 0x00,
00563 0x00,
00564 0x00,
00565 0x00,
00566 0x00,
00567 0x00,
00568 0x00,
00569 0x00,
00570 0x00,
00571 0x00,
00572 0x00,
00573 0x00,
00574 0x00,
00575 0x00,
00576 0x00,
00577 0x00,
00578 0x00,
00579 0x00,
00580 0x00,
00581 0x00,
00582 0x00,
00583 0x00,
00584 0x00,
00585 0x00,
00586 0x00,
00587 0x00,
00588 0x00,
00589 0x00,
00590 /* U+0A02 */ 0x0A02,
00591 0x00,
00592 0x00,
00593 0x00,
00594 0x00,
00595 0x00,
00596 0x00,
00597 0x00,
00598 0x00,
00599 0x00,
00600 0x00,
00601 0x00,
00602 0x00,
00603 0x00,
00604 0x00,
00605 0x00,
00606 0x00,
00607 0x00,
00608 0x00,
00609 0x00,
00610 0x00,
00611 0x00,
00612 0x00,
00613 0x00,
00614 0x00,
00615 0x00,
00616 0x00,
00617 0x00,
00618 0x00,
00619 0x00,
00620 0x00,
00621 0x00,
00622 0x00,
00623 0x00,
00624 0x00,
00625 0x00,
00626 0x00,
00627 0x00,
00628 0x00,
00629 0x00,
00630 0x00,
00631 0x00,
00632 0x00,
00633 0x00,
00634 0x00,
00635 0x00,
00636 0x00,
00637 0x00,
00638 0x00,
00639 0x00,
00640 0x00,
00641 0x00,
00642 0x00,
```

```
00643 0x00,
00644 0x00,
00645 0x00,
00646 0x00,
00647 0x00,
00648 /* U+0A3C */ 0x0A3C,
00649 0x00,
00650 0x00,
00651 0x00,
00652 0x00,
00653 /* U+0A41 */ 0x0A41,
00654 /* U+0A42 */ 0x0A42,
00655 0x00,
00656 0x00,
00657 0x00,
00658 0x00,
00659 /* U+0A47 */ 0x0A47,
00660 /* U+0A48 */ 0x0A48,
00661 0x00,
00662 0x00,
00663 /* U+0A4B */ 0x0A4B,
00664 /* U+0A4C */ 0x0A4C,
00665 /* U+0A4D */ 0x0A4D,
00666 0x00,
00667 0x00,
00668 0x00,
00669 0x00,
00670 0x00,
00671 0x00,
00672 0x00,
00673 0x00,
00674 0x00,
00675 0x00,
00676 0x00,
00677 0x00,
00678 0x00,
00679 0x00,
00680 0x00,
00681 0x00,
00682 0x00,
00683 0x00,
00684 0x00,
00685 0x00,
00686 0x00,
00687 0x00,
00688 0x00,
00689 0x00,
00690 0x00,
00691 0x00,
00692 0x00,
00693 0x00,
00694 0x00,
00695 0x00,
00696 0x00,
00697 0x00,
00698 0x00,
00699 0x00,
00700 /* U+0A70 */ 0x0A70,
00701 /* U+0A71 */ 0x0A71,
00702 0x00,
00703 0x00,
00704 0x00,
00705 0x00,
00706 0x00,
00707 0x00,
00708 0x00,
00709 0x00,
00710 0x00,
00711 0x00,
00712 0x00,
00713 0x00,
00714 0x00,
00715 0x00,
00716 0x00,
00717 /* U+0A81 */ 0x0A81,
00718 /* U+0A82 */ 0x0A82,
00719 0x00,
00720 0x00,
00721 0x00,
00722 0x00,
00723 0x00,
00724 0x00,
00725 0x00,
00726 0x00,
00727 0x00,
00728 0x00,
00729 0x00,
```

```
00730 0x00,
00731 0x00,
00732 0x00,
00733 0x00,
00734 0x00,
00735 0x00,
00736 0x00,
00737 0x00,
00738 0x00,
00739 0x00,
00740 0x00,
00741 0x00,
00742 0x00,
00743 0x00,
00744 0x00,
00745 0x00,
00746 0x00,
00747 0x00,
00748 0x00,
00749 0x00,
00750 0x00,
00751 0x00,
00752 0x00,
00753 0x00,
00754 0x00,
00755 0x00,
00756 0x00,
00757 0x00,
00758 0x00,
00759 0x00,
00760 0x00,
00761 0x00,
00762 0x00,
00763 0x00,
00764 0x00,
00765 0x00,
00766 0x00,
00767 0x00,
00768 0x00,
00769 0x00,
00770 0x00,
00771 0x00,
00772 0x00,
00773 0x00,
00774 0x00,
00775 0x00,
00776 /* U+0ABC */ 0x0ABC,
00777 0x00,
00778 0x00,
00779 0x00,
00780 0x00,
00781 /* U+0AC1 */ 0x0AC1,
00782 /* U+0AC2 */ 0x0AC2,
00783 /* U+0AC3 */ 0x0AC3,
00784 /* U+0AC4 */ 0x0AC4,
00785 /* U+0AC5 */ 0x0AC5,
00786 0x00,
00787 /* U+0AC7 */ 0x0AC7,
00788 /* U+0AC8 */ 0x0AC8,
00789 0x00,
00790 0x00,
00791 0x00,
00792 0x00,
00793 /* U+0ACD */ 0x0ACD,
00794 0x00,
00795 0x00,
00796 0x00,
00797 0x00,
00798 0x00,
00799 0x00,
00800 0x00,
00801 0x00,
00802 0x00,
00803 0x00,
00804 0x00,
00805 0x00,
00806 0x00,
00807 0x00,
00808 0x00,
00809 0x00,
00810 0x00,
00811 0x00,
00812 0x00,
00813 0x00,
00814 0x00,
00815 0x00,
00816 0x00,
```



```
00817 0x00,
00818 0x00,
00819 0x00,
00820 0x00,
00821 0x00,
00822 0x00,
00823 0x00,
00824 0x00,
00825 0x00,
00826 0x00,
00827 0x00,
00828 0x00,
00829 0x00,
00830 0x00,
00831 0x00,
00832 0x00,
00833 0x00,
00834 0x00,
00835 0x00,
00836 0x00,
00837 0x00,
00838 0x00,
00839 0x00,
00840 0x00,
00841 0x00,
00842 0x00,
00843 0x00,
00844 0x00,
00845 /* U+0B01 */ 0x0B01,
00846 0x00,
00847 0x00,
00848 0x00,
00849 0x00,
00850 0x00,
00851 0x00,
00852 0x00,
00853 0x00,
00854 0x00,
00855 0x00,
00856 0x00,
00857 0x00,
00858 0x00,
00859 0x00,
00860 0x00,
00861 0x00,
00862 0x00,
00863 0x00,
00864 0x00,
00865 0x00,
00866 0x00,
00867 0x00,
00868 0x00,
00869 0x00,
00870 0x00,
00871 0x00,
00872 0x00,
00873 0x00,
00874 0x00,
00875 0x00,
00876 0x00,
00877 0x00,
00878 0x00,
00879 0x00,
00880 0x00,
00881 0x00,
00882 0x00,
00883 0x00,
00884 0x00,
00885 0x00,
00886 0x00,
00887 0x00,
00888 0x00,
00889 0x00,
00890 0x00,
00891 0x00,
00892 0x00,
00893 0x00,
00894 0x00,
00895 0x00,
00896 0x00,
00897 0x00,
00898 0x00,
00899 0x00,
00900 0x00,
00901 0x00,
00902 0x00,
00903 0x00,
```

```
00904 /* U+0B3C */ 0x0B3C,
00905 0x00,
00906 0x00,
00907 /* U+0B3F */ 0x0B3F,
00908 0x00,
00909 /* U+0B41 */ 0x0B41,
00910 /* U+0B42 */ 0x0B42,
00911 /* U+0B43 */ 0x0B43,
00912 0x00,
00913 0x00,
00914 0x00,
00915 0x00,
00916 0x00,
00917 0x00,
00918 0x00,
00919 0x00,
00920 0x00,
00921 /* U+0B4D */ 0x0B4D,
00922 0x00,
00923 0x00,
00924 0x00,
00925 0x00,
00926 0x00,
00927 0x00,
00928 0x00,
00929 0x00,
00930 /* U+0B56 */ 0x0B56,
00931 0x00,
00932 0x00,
00933 0x00,
00934 0x00,
00935 0x00,
00936 0x00,
00937 0x00,
00938 0x00,
00939 0x00,
00940 0x00,
00941 0x00,
00942 0x00,
00943 0x00,
00944 0x00,
00945 0x00,
00946 0x00,
00947 0x00,
00948 0x00,
00949 0x00,
00950 0x00,
00951 0x00,
00952 0x00,
00953 0x00,
00954 0x00,
00955 0x00,
00956 0x00,
00957 0x00,
00958 0x00,
00959 0x00,
00960 0x00,
00961 0x00,
00962 0x00,
00963 0x00,
00964 0x00,
00965 0x00,
00966 0x00,
00967 0x00,
00968 0x00,
00969 0x00,
00970 0x00,
00971 0x00,
00972 0x00,
00973 0x00,
00974 /* U+0B82 */ 0x0B82,
00975 0x00,
00976 0x00,
00977 0x00,
00978 0x00,
00979 0x00,
00980 0x00,
00981 0x00,
00982 0x00,
00983 0x00,
00984 0x00,
00985 0x00,
00986 0x00,
00987 0x00,
00988 0x00,
00989 0x00,
00990 0x00,
```

```
00991 0x00,
00992 0x00,
00993 0x00,
00994 0x00,
00995 0x00,
00996 0x00,
00997 0x00,
00998 0x00,
00999 0x00,
01000 0x00,
01001 0x00,
01002 0x00,
01003 0x00,
01004 0x00,
01005 0x00,
01006 0x00,
01007 0x00,
01008 0x00,
01009 0x00,
01010 0x00,
01011 0x00,
01012 0x00,
01013 0x00,
01014 0x00,
01015 0x00,
01016 0x00,
01017 0x00,
01018 0x00,
01019 0x00,
01020 0x00,
01021 0x00,
01022 0x00,
01023 0x00,
01024 0x00,
01025 0x00,
01026 0x00,
01027 0x00,
01028 0x00,
01029 0x00,
01030 0x00,
01031 0x00,
01032 0x00,
01033 0x00,
01034 0x00,
01035 0x00,
01036 /* U+0BC0 */ 0x0BC0,
01037 0x00,
01038 0x00,
01039 0x00,
01040 0x00,
01041 0x00,
01042 0x00,
01043 0x00,
01044 0x00,
01045 0x00,
01046 0x00,
01047 0x00,
01048 0x00,
01049 /* U+0BCD */ 0x0BCD,
01050 0x00,
01051 0x00,
01052 0x00,
01053 0x00,
01054 0x00,
01055 0x00,
01056 0x00,
01057 0x00,
01058 0x00,
01059 0x00,
01060 0x00,
01061 0x00,
01062 0x00,
01063 0x00,
01064 0x00,
01065 0x00,
01066 0x00,
01067 0x00,
01068 0x00,
01069 0x00,
01070 0x00,
01071 0x00,
01072 0x00,
01073 0x00,
01074 0x00,
01075 0x00,
01076 0x00,
01077 0x00,
```

```
01078 0x00,
01079 0x00,
01080 0x00,
01081 0x00,
01082 0x00,
01083 0x00,
01084 0x00,
01085 0x00,
01086 0x00,
01087 0x00,
01088 0x00,
01089 0x00,
01090 0x00,
01091 0x00,
01092 0x00,
01093 0x00,
01094 0x00,
01095 0x00,
01096 0x00,
01097 0x00,
01098 0x00,
01099 0x00,
01100 0x00,
01101 0x00,
01102 0x00,
01103 0x00,
01104 0x00,
01105 0x00,
01106 0x00,
01107 0x00,
01108 0x00,
01109 0x00,
01110 0x00,
01111 0x00,
01112 0x00,
01113 0x00,
01114 0x00,
01115 0x00,
01116 0x00,
01117 0x00,
01118 0x00,
01119 0x00,
01120 0x00,
01121 0x00,
01122 0x00,
01123 0x00,
01124 0x00,
01125 0x00,
01126 0x00,
01127 0x00,
01128 0x00,
01129 0x00,
01130 0x00,
01131 0x00,
01132 0x00,
01133 0x00,
01134 0x00,
01135 0x00,
01136 0x00,
01137 0x00,
01138 0x00,
01139 0x00,
01140 0x00,
01141 0x00,
01142 0x00,
01143 0x00,
01144 0x00,
01145 0x00,
01146 0x00,
01147 0x00,
01148 0x00,
01149 0x00,
01150 0x00,
01151 0x00,
01152 0x00,
01153 0x00,
01154 0x00,
01155 0x00,
01156 0x00,
01157 0x00,
01158 0x00,
01159 0x00,
01160 0x00,
01161 0x00,
01162 /* U+0C3E */ 0x0C3E,
01163 /* U+0C3F */ 0x0C3F,
01164 /* U+0C40 */ 0x0C40,
```

```
01165 0x00,
01166 0x00,
01167 0x00,
01168 0x00,
01169 0x00,
01170 /* U+0C46 */ 0x0C46,
01171 /* U+0C47 */ 0x0C47,
01172 /* U+0C48 */ 0x0C48,
01173 0x00,
01174 /* U+0C4A */ 0x0C4A,
01175 /* U+0C4B */ 0x0C4B,
01176 /* U+0C4C */ 0x0C4C,
01177 /* U+0C4D */ 0x0C4D,
01178 0x00,
01179 0x00,
01180 0x00,
01181 0x00,
01182 0x00,
01183 0x00,
01184 0x00,
01185 /* U+0C55 */ 0x0C55,
01186 /* U+0C56 */ 0x0C56,
01187 0x00,
01188 0x00,
01189 0x00,
01190 0x00,
01191 0x00,
01192 0x00,
01193 0x00,
01194 0x00,
01195 0x00,
01196 0x00,
01197 0x00,
01198 0x00,
01199 0x00,
01200 0x00,
01201 0x00,
01202 0x00,
01203 0x00,
01204 0x00,
01205 0x00,
01206 0x00,
01207 0x00,
01208 0x00,
01209 0x00,
01210 0x00,
01211 0x00,
01212 0x00,
01213 0x00,
01214 0x00,
01215 0x00,
01216 0x00,
01217 0x00,
01218 0x00,
01219 0x00,
01220 0x00,
01221 0x00,
01222 0x00,
01223 0x00,
01224 0x00,
01225 0x00,
01226 0x00,
01227 0x00,
01228 0x00,
01229 0x00,
01230 0x00,
01231 0x00,
01232 0x00,
01233 0x00,
01234 0x00,
01235 0x00,
01236 0x00,
01237 0x00,
01238 0x00,
01239 0x00,
01240 0x00,
01241 0x00,
01242 0x00,
01243 0x00,
01244 0x00,
01245 0x00,
01246 0x00,
01247 0x00,
01248 0x00,
01249 0x00,
01250 0x00,
01251 0x00,
```

```
01252 0x00,
01253 0x00,
01254 0x00,
01255 0x00,
01256 0x00,
01257 0x00,
01258 0x00,
01259 0x00,
01260 0x00,
01261 0x00,
01262 0x00,
01263 0x00,
01264 0x00,
01265 0x00,
01266 0x00,
01267 0x00,
01268 0x00,
01269 0x00,
01270 0x00,
01271 0x00,
01272 0x00,
01273 0x00,
01274 0x00,
01275 0x00,
01276 0x00,
01277 0x00,
01278 0x00,
01279 0x00,
01280 0x00,
01281 0x00,
01282 0x00,
01283 0x00,
01284 0x00,
01285 0x00,
01286 0x00,
01287 0x00,
01288 0x00,
01289 0x00,
01290 0x00,
01291 /* U+0CBF */ 0x0CBF,
01292 0x00,
01293 0x00,
01294 0x00,
01295 0x00,
01296 0x00,
01297 0x00,
01298 /* U+0CC6 */ 0x0CC6,
01299 0x00,
01300 0x00,
01301 0x00,
01302 0x00,
01303 0x00,
01304 /* U+0CCC */ 0x0CCC,
01305 /* U+0CCD */ 0x0CCD,
01306 0x00,
01307 0x00,
01308 0x00,
01309 0x00,
01310 0x00,
01311 0x00,
01312 0x00,
01313 0x00,
01314 0x00,
01315 0x00,
01316 0x00,
01317 0x00,
01318 0x00,
01319 0x00,
01320 0x00,
01321 0x00,
01322 0x00,
01323 0x00,
01324 0x00,
01325 0x00,
01326 0x00,
01327 0x00,
01328 0x00,
01329 0x00,
01330 0x00,
01331 0x00,
01332 0x00,
01333 0x00,
01334 0x00,
01335 0x00,
01336 0x00,
01337 0x00,
01338 0x00,
```

```
01339 0x00,
01340 0x00,
01341 0x00,
01342 0x00,
01343 0x00,
01344 0x00,
01345 0x00,
01346 0x00,
01347 0x00,
01348 0x00,
01349 0x00,
01350 0x00,
01351 0x00,
01352 0x00,
01353 0x00,
01354 0x00,
01355 0x00,
01356 0x00,
01357 0x00,
01358 0x00,
01359 0x00,
01360 0x00,
01361 0x00,
01362 0x00,
01363 0x00,
01364 0x00,
01365 0x00,
01366 0x00,
01367 0x00,
01368 0x00,
01369 0x00,
01370 0x00,
01371 0x00,
01372 0x00,
01373 0x00,
01374 0x00,
01375 0x00,
01376 0x00,
01377 0x00,
01378 0x00,
01379 0x00,
01380 0x00,
01381 0x00,
01382 0x00,
01383 0x00,
01384 0x00,
01385 0x00,
01386 0x00,
01387 0x00,
01388 0x00,
01389 0x00,
01390 0x00,
01391 0x00,
01392 0x00,
01393 0x00,
01394 0x00,
01395 0x00,
01396 0x00,
01397 0x00,
01398 0x00,
01399 0x00,
01400 0x00,
01401 0x00,
01402 0x00,
01403 0x00,
01404 0x00,
01405 0x00,
01406 0x00,
01407 0x00,
01408 0x00,
01409 0x00,
01410 0x00,
01411 0x00,
01412 0x00,
01413 0x00,
01414 0x00,
01415 0x00,
01416 0x00,
01417 0x00,
01418 0x00,
01419 0x00,
01420 0x00,
01421 /* U+0D41 */ 0x0D41,
01422 /* U+0D42 */ 0x0D42,
01423 /* U+0D43 */ 0x0D43,
01424 0x00,
01425 0x00,
```

```
01426 0x00,
01427 0x00,
01428 0x00,
01429 0x00,
01430 0x00,
01431 0x00,
01432 0x00,
01433 /* U+0D4D */ 0x0D4D,
01434 };
01435
01436 static const unsigned short ucs_table_0E31[] = {
01437 /* U+0E31 */ 0x0E31,
01438 0x00,
01439 0x00,
01440 /* U+0E34 */ 0x0E34,
01441 /* U+0E35 */ 0x0E35,
01442 /* U+0E36 */ 0x0E36,
01443 /* U+0E37 */ 0x0E37,
01444 /* U+0E38 */ 0x0E38,
01445 /* U+0E39 */ 0x0E39,
01446 /* U+0E3A */ 0x0E3A,
01447 0x00,
01448 0x00,
01449 0x00,
01450 0x00,
01451 0x00,
01452 0x00,
01453 0x00,
01454 0x00,
01455 0x00,
01456 0x00,
01457 0x00,
01458 0x00,
01459 /* U+0E47 */ 0x0E47,
01460 /* U+0E48 */ 0x0E48,
01461 /* U+0E49 */ 0x0E49,
01462 /* U+0E4A */ 0x0E4A,
01463 /* U+0E4B */ 0x0E4B,
01464 /* U+0E4C */ 0x0E4C,
01465 /* U+0E4D */ 0x0E4D,
01466 /* U+0E4E */ 0x0E4E,
01467 0x00,
01468 0x00,
01469 0x00,
01470 0x00,
01471 0x00,
01472 0x00,
01473 0x00,
01474 0x00,
01475 0x00,
01476 0x00,
01477 0x00,
01478 0x00,
01479 0x00,
01480 0x00,
01481 0x00,
01482 0x00,
01483 0x00,
01484 0x00,
01485 0x00,
01486 0x00,
01487 0x00,
01488 0x00,
01489 0x00,
01490 0x00,
01491 0x00,
01492 0x00,
01493 0x00,
01494 0x00,
01495 0x00,
01496 0x00,
01497 0x00,
01498 0x00,
01499 0x00,
01500 0x00,
01501 0x00,
01502 0x00,
01503 0x00,
01504 0x00,
01505 0x00,
01506 0x00,
01507 0x00,
01508 0x00,
01509 0x00,
01510 0x00,
01511 0x00,
01512 0x00,
```



```
01513 0x00,
01514 0x00,
01515 0x00,
01516 0x00,
01517 0x00,
01518 0x00,
01519 0x00,
01520 0x00,
01521 0x00,
01522 0x00,
01523 0x00,
01524 0x00,
01525 0x00,
01526 0x00,
01527 0x00,
01528 0x00,
01529 0x00,
01530 0x00,
01531 0x00,
01532 0x00,
01533 0x00,
01534 0x00,
01535 0x00,
01536 0x00,
01537 0x00,
01538 0x00,
01539 0x00,
01540 0x00,
01541 0x00,
01542 0x00,
01543 0x00,
01544 0x00,
01545 0x00,
01546 0x00,
01547 0x00,
01548 0x00,
01549 0x00,
01550 0x00,
01551 0x00,
01552 0x00,
01553 0x00,
01554 0x00,
01555 0x00,
01556 0x00,
01557 0x00,
01558 0x00,
01559 0x00,
01560 0x00,
01561 0x00,
01562 0x00,
01563 0x00,
01564 0x00,
01565 /* U+0EB1 */ 0x0EB1,
01566 0x00,
01567 0x00,
01568 /* U+0EB4 */ 0x0EB4,
01569 /* U+0EB5 */ 0x0EB5,
01570 /* U+0EB6 */ 0x0EB6,
01571 /* U+0EB7 */ 0x0EB7,
01572 /* U+0EB8 */ 0x0EB8,
01573 /* U+0EB9 */ 0x0EB9,
01574 0x00,
01575 /* U+0EBB */ 0x0EBB,
01576 /* U+0EBC */ 0x0EBC,
01577 0x00,
01578 0x00,
01579 0x00,
01580 0x00,
01581 0x00,
01582 0x00,
01583 0x00,
01584 0x00,
01585 0x00,
01586 0x00,
01587 0x00,
01588 /* U+0EC8 */ 0x0EC8,
01589 /* U+0EC9 */ 0x0EC9,
01590 /* U+0ECA */ 0x0ECA,
01591 /* U+0ECB */ 0x0ECB,
01592 /* U+0ECC */ 0x0ECC,
01593 /* U+0ECD */ 0x0ECD,
01594 0x00,
01595 0x00,
01596 0x00,
01597 0x00,
01598 0x00,
01599 0x00,
```

```
01600 0x00,
01601 0x00,
01602 0x00,
01603 0x00,
01604 0x00,
01605 0x00,
01606 0x00,
01607 0x00,
01608 0x00,
01609 0x00,
01610 0x00,
01611 0x00,
01612 0x00,
01613 0x00,
01614 0x00,
01615 0x00,
01616 0x00,
01617 0x00,
01618 0x00,
01619 0x00,
01620 0x00,
01621 0x00,
01622 0x00,
01623 0x00,
01624 0x00,
01625 0x00,
01626 0x00,
01627 0x00,
01628 0x00,
01629 0x00,
01630 0x00,
01631 0x00,
01632 0x00,
01633 0x00,
01634 0x00,
01635 0x00,
01636 0x00,
01637 0x00,
01638 0x00,
01639 0x00,
01640 0x00,
01641 0x00,
01642 0x00,
01643 0x00,
01644 0x00,
01645 0x00,
01646 0x00,
01647 0x00,
01648 0x00,
01649 0x00,
01650 0x00,
01651 0x00,
01652 0x00,
01653 0x00,
01654 0x00,
01655 0x00,
01656 0x00,
01657 0x00,
01658 0x00,
01659 0x00,
01660 0x00,
01661 0x00,
01662 0x00,
01663 0x00,
01664 0x00,
01665 0x00,
01666 0x00,
01667 0x00,
01668 /* U+0F18 */ 0x0F18,
01669 /* U+0F19 */ 0x0F19,
01670 0x00,
01671 0x00,
01672 0x00,
01673 0x00,
01674 0x00,
01675 0x00,
01676 0x00,
01677 0x00,
01678 0x00,
01679 0x00,
01680 0x00,
01681 0x00,
01682 0x00,
01683 0x00,
01684 0x00,
01685 0x00,
01686 0x00,
```

```
01687 0x00,
01688 0x00,
01689 0x00,
01690 0x00,
01691 0x00,
01692 0x00,
01693 0x00,
01694 0x00,
01695 0x00,
01696 0x00,
01697 /* U+0F35 */ 0x0F35,
01698 0x00,
01699 /* U+0F37 */ 0x0F37,
01700 0x00,
01701 /* U+0F39 */ 0x0F39,
01702 0x00,
01703 0x00,
01704 0x00,
01705 0x00,
01706 0x00,
01707 0x00,
01708 0x00,
01709 0x00,
01710 0x00,
01711 0x00,
01712 0x00,
01713 0x00,
01714 0x00,
01715 0x00,
01716 0x00,
01717 0x00,
01718 0x00,
01719 0x00,
01720 0x00,
01721 0x00,
01722 0x00,
01723 0x00,
01724 0x00,
01725 0x00,
01726 0x00,
01727 0x00,
01728 0x00,
01729 0x00,
01730 0x00,
01731 0x00,
01732 0x00,
01733 0x00,
01734 0x00,
01735 0x00,
01736 0x00,
01737 0x00,
01738 0x00,
01739 0x00,
01740 0x00,
01741 0x00,
01742 0x00,
01743 0x00,
01744 0x00,
01745 0x00,
01746 0x00,
01747 0x00,
01748 0x00,
01749 0x00,
01750 0x00,
01751 0x00,
01752 0x00,
01753 0x00,
01754 0x00,
01755 0x00,
01756 0x00,
01757 /* U+0F71 */ 0x0F71,
01758 /* U+0F72 */ 0x0F72,
01759 /* U+0F73 */ 0x0F73,
01760 /* U+0F74 */ 0x0F74,
01761 /* U+0F75 */ 0x0F75,
01762 /* U+0F76 */ 0x0F76,
01763 /* U+0F77 */ 0x0F77,
01764 /* U+0F78 */ 0x0F78,
01765 /* U+0F79 */ 0x0F79,
01766 /* U+0F7A */ 0x0F7A,
01767 /* U+0F7B */ 0x0F7B,
01768 /* U+0F7C */ 0x0F7C,
01769 /* U+0F7D */ 0x0F7D,
01770 /* U+0F7E */ 0x0F7E,
01771 0x00,
01772 /* U+0F80 */ 0x0F80,
01773 /* U+0F81 */ 0x0F81,
```

```
01774 /* U+0F82 */ 0x0F82,
01775 /* U+0F83 */ 0x0F83,
01776 /* U+0F84 */ 0x0F84,
01777 0x00,
01778 /* U+0F86 */ 0x0F86,
01779 /* U+0F87 */ 0x0F87,
01780 0x00,
01781 0x00,
01782 0x00,
01783 0x00,
01784 0x00,
01785 0x00,
01786 0x00,
01787 0x00,
01788 /* U+0F90 */ 0x0F90,
01789 /* U+0F91 */ 0x0F91,
01790 /* U+0F92 */ 0x0F92,
01791 /* U+0F93 */ 0x0F93,
01792 /* U+0F94 */ 0x0F94,
01793 /* U+0F95 */ 0x0F95,
01794 0x00,
01795 /* U+0F97 */ 0x0F97,
01796 0x00,
01797 /* U+0F99 */ 0x0F99,
01798 /* U+0F9A */ 0x0F9A,
01799 /* U+0F9B */ 0x0F9B,
01800 /* U+0F9C */ 0x0F9C,
01801 /* U+0F9D */ 0x0F9D,
01802 /* U+0F9E */ 0x0F9E,
01803 /* U+0F9F */ 0x0F9F,
01804 /* U+0FA0 */ 0x0FA0,
01805 /* U+0FA1 */ 0x0FA1,
01806 /* U+0FA2 */ 0x0FA2,
01807 /* U+0FA3 */ 0x0FA3,
01808 /* U+0FA4 */ 0x0FA4,
01809 /* U+0FA5 */ 0x0FA5,
01810 /* U+0FA6 */ 0x0FA6,
01811 /* U+0FA7 */ 0x0FA7,
01812 /* U+0FA8 */ 0x0FA8,
01813 /* U+0FA9 */ 0x0FA9,
01814 /* U+0FAA */ 0x0FAA,
01815 /* U+0FAB */ 0x0FAB,
01816 /* U+0FAC */ 0x0FAC,
01817 /* U+0FAD */ 0x0FAD,
01818 0x00,
01819 0x00,
01820 0x00,
01821 /* U+0FB1 */ 0x0FB1,
01822 /* U+0FB2 */ 0x0FB2,
01823 /* U+0FB3 */ 0x0FB3,
01824 /* U+0FB4 */ 0x0FB4,
01825 /* U+0FB5 */ 0x0FB5,
01826 /* U+0FB6 */ 0x0FB6,
01827 /* U+0FB7 */ 0x0FB7,
01828 0x00,
01829 /* U+0FB9 */ 0x0FB9,
01830 };
01831
01832 static const unsigned short ucs_table_20D0[] = {
01833 /* U+20D0 */ 0x20D0,
01834 /* U+20D1 */ 0x20D1,
01835 /* U+20D2 */ 0x20D2,
01836 /* U+20D3 */ 0x20D3,
01837 /* U+20D4 */ 0x20D4,
01838 /* U+20D5 */ 0x20D5,
01839 /* U+20D6 */ 0x20D6,
01840 /* U+20D7 */ 0x20D7,
01841 /* U+20D8 */ 0x20D8,
01842 /* U+20D9 */ 0x20D9,
01843 /* U+20DA */ 0x20DA,
01844 /* U+20DB */ 0x20DB,
01845 /* U+20DC */ 0x20DC,
01846 0x00,
01847 0x00,
01848 0x00,
01849 0x00,
01850 /* U+20E1 */ 0x20E1,
01851 };
01852
01853 static const unsigned short ucs_table_302A[] = {
01854 /* U+302A */ 0x302A,
01855 /* U+302B */ 0x302B,
01856 /* U+302C */ 0x302C,
01857 /* U+302D */ 0x302D,
01858 /* U+302E */ 0x302E,
01859 /* U+302F */ 0x302F,
01860 0x00,
```

```
01861 0x00,  
01862 0x00,  
01863 0x00,  
01864 0x00,  
01865 0x00,  
01866 0x00,  
01867 0x00,  
01868 0x00,  
01869 0x00,  
01870 0x00,  
01871 0x00,  
01872 0x00,  
01873 0x00,  
01874 0x00,  
01875 0x00,  
01876 0x00,  
01877 0x00,  
01878 0x00,  
01879 0x00,  
01880 0x00,  
01881 0x00,  
01882 0x00,  
01883 0x00,  
01884 0x00,  
01885 0x00,  
01886 0x00,  
01887 0x00,  
01888 0x00,  
01889 0x00,  
01890 0x00,  
01891 0x00,  
01892 0x00,  
01893 0x00,  
01894 0x00,  
01895 0x00,  
01896 0x00,  
01897 0x00,  
01898 0x00,  
01899 0x00,  
01900 0x00,  
01901 0x00,  
01902 0x00,  
01903 0x00,  
01904 0x00,  
01905 0x00,  
01906 0x00,  
01907 0x00,  
01908 0x00,  
01909 0x00,  
01910 0x00,  
01911 0x00,  
01912 0x00,  
01913 0x00,  
01914 0x00,  
01915 0x00,  
01916 0x00,  
01917 0x00,  
01918 0x00,  
01919 0x00,  
01920 0x00,  
01921 0x00,  
01922 0x00,  
01923 0x00,  
01924 0x00,  
01925 0x00,  
01926 0x00,  
01927 0x00,  
01928 0x00,  
01929 0x00,  
01930 0x00,  
01931 0x00,  
01932 0x00,  
01933 0x00,  
01934 0x00,  
01935 0x00,  
01936 0x00,  
01937 0x00,  
01938 0x00,  
01939 0x00,  
01940 0x00,  
01941 0x00,  
01942 0x00,  
01943 0x00,  
01944 0x00,  
01945 0x00,  
01946 0x00,  
01947 0x00,
```

```
01948 0x00,
01949 0x00,
01950 0x00,
01951 0x00,
01952 0x00,
01953 0x00,
01954 0x00,
01955 0x00,
01956 0x00,
01957 0x00,
01958 0x00,
01959 0x00,
01960 0x00,
01961 0x00,
01962 0x00,
01963 0x00,
01964 0x00,
01965 /* U+3099 */ 0x309B,
01966 /* U+309A */ 0x309C,
01967 };
01968
01969 static const unsigned short ucs_table_FB1E[] = {
01970 /* U+FB1E */ 0xFB1E,
01971 };
01972
01973 static const unsigned short ucs_table_FE20[] = {
01974 /* U+FE20 */ 0xFE20,
01975 /* U+FE21 */ 0xFE21,
01976 /* U+FE22 */ 0xFE22,
01977 /* U+FE23 */ 0xFE23,
01978 };
```

12.255 symbol_1b.h

```
00001 /* symbol */
00002
00003 static const char unicode_to_symbol_1b_0020[] = {
00004 /* U+0020 */ 0x20,
00005 /* U+0021 */ 0x21,
00006 0x00,
00007 /* U+0023 */ 0x23,
00008 0x00,
00009 /* U+0025 */ 0x25,
00010 /* U+0026 */ 0x26,
00011 0x00,
00012 /* U+0028 */ 0x28,
00013 /* U+0029 */ 0x29,
00014 0x00,
00015 /* U+002B */ 0x2B,
00016 /* U+002C */ 0x2C,
00017 0x00,
00018 /* U+002E */ 0x2E,
00019 /* U+002F */ 0x2F,
00020 /* U+0030 */ 0x30,
00021 /* U+0031 */ 0x31,
00022 /* U+0032 */ 0x32,
00023 /* U+0033 */ 0x33,
00024 /* U+0034 */ 0x34,
00025 /* U+0035 */ 0x35,
00026 /* U+0036 */ 0x36,
00027 /* U+0037 */ 0x37,
00028 /* U+0038 */ 0x38,
00029 /* U+0039 */ 0x39,
00030 /* U+003A */ 0x3A,
00031 /* U+003B */ 0x3B,
00032 /* U+003C */ 0x3C,
00033 /* U+003D */ 0x3D,
00034 /* U+003E */ 0x3E,
00035 /* U+003F */ 0x3F,
00036 0x00,
00037 0x00,
00038 0x00,
00039 0x00,
00040 0x00,
00041 0x00,
00042 0x00,
00043 0x00,
00044 0x00,
00045 0x00,
00046 0x00,
00047 0x00,
00048 0x00,
00049 0x00,
00050 0x00,
00051 0x00,
```

```
00052 0x00,
00053 0x00,
00054 0x00,
00055 0x00,
00056 0x00,
00057 0x00,
00058 0x00,
00059 0x00,
00060 0x00,
00061 0x00,
00062 0x00,
00063 /* U+005B */ 0x5B,
00064 0x00,
00065 /* U+005D */ 0x5D,
00066 0x00,
00067 /* U+005F */ 0x5F,
00068 0x00,
00069 0x00,
00070 0x00,
00071 0x00,
00072 0x00,
00073 0x00,
00074 0x00,
00075 0x00,
00076 0x00,
00077 0x00,
00078 0x00,
00079 0x00,
00080 0x00,
00081 0x00,
00082 0x00,
00083 0x00,
00084 0x00,
00085 0x00,
00086 0x00,
00087 0x00,
00088 0x00,
00089 0x00,
00090 0x00,
00091 0x00,
00092 0x00,
00093 0x00,
00094 0x00,
00095 /* U+007B */ 0x7B,
00096 /* U+007C */ 0x7C,
00097 /* U+007D */ 0x7D,
00098 0x00,
00099 0x00,
00100 0x00,
00101 0x00,
00102 0x00,
00103 0x00,
00104 0x00,
00105 0x00,
00106 0x00,
00107 0x00,
00108 0x00,
00109 0x00,
00110 0x00,
00111 0x00,
00112 0x00,
00113 0x00,
00114 0x00,
00115 0x00,
00116 0x00,
00117 0x00,
00118 0x00,
00119 0x00,
00120 0x00,
00121 0x00,
00122 0x00,
00123 0x00,
00124 0x00,
00125 0x00,
00126 0x00,
00127 0x00,
00128 0x00,
00129 0x00,
00130 0x00,
00131 0x00,
00132 /* U+00A0 */ 0x20,
00133 0x00,
00134 0x00,
00135 0x00,
00136 0x00,
00137 0x00,
00138 0x00,
```

```
00139 0x00,
00140 0x00,
00141 0x00,
00142 0x00,
00143 0x00,
00144 /* U+00AC */ (char)0xD8,
00145 0x00,
00146 0x00,
00147 0x00,
00148 /* U+00B0 */ (char)0xB0,
00149 /* U+00B1 */ (char)0xB1,
00150 0x00,
00151 0x00,
00152 0x00,
00153 /* U+00B5 */ 0x6D,
00154 0x00,
00155 0x00,
00156 0x00,
00157 0x00,
00158 0x00,
00159 0x00,
00160 0x00,
00161 0x00,
00162 0x00,
00163 0x00,
00164 0x00,
00165 0x00,
00166 0x00,
00167 0x00,
00168 0x00,
00169 0x00,
00170 0x00,
00171 0x00,
00172 0x00,
00173 0x00,
00174 0x00,
00175 0x00,
00176 0x00,
00177 0x00,
00178 0x00,
00179 0x00,
00180 0x00,
00181 0x00,
00182 0x00,
00183 0x00,
00184 0x00,
00185 0x00,
00186 0x00,
00187 /* U+00D7 */ (char)0xB4,
00188 0x00,
00189 0x00,
00190 0x00,
00191 0x00,
00192 0x00,
00193 0x00,
00194 0x00,
00195 0x00,
00196 0x00,
00197 0x00,
00198 0x00,
00199 0x00,
00200 0x00,
00201 0x00,
00202 0x00,
00203 0x00,
00204 0x00,
00205 0x00,
00206 0x00,
00207 0x00,
00208 0x00,
00209 0x00,
00210 0x00,
00211 0x00,
00212 0x00,
00213 0x00,
00214 0x00,
00215 0x00,
00216 0x00,
00217 0x00,
00218 0x00,
00219 /* U+00F7 */ (char)0xB8,
00220 };
00221
00222 static const char unicode_to_symbol_lb_0192[] = {
00223 /* U+0192 */ (char)0xA6,
00224 };
00225
```



```
00226 static const char unicode_to_symbol_1b_0391[] = {
00227 /* U+0391 */ 0x41,
00228 /* U+0392 */ 0x42,
00229 /* U+0393 */ 0x47,
00230 /* U+0394 */ 0x44,
00231 /* U+0395 */ 0x45,
00232 /* U+0396 */ 0x5A,
00233 /* U+0397 */ 0x48,
00234 /* U+0398 */ 0x51,
00235 /* U+0399 */ 0x49,
00236 /* U+039A */ 0x4B,
00237 /* U+039B */ 0x4C,
00238 /* U+039C */ 0x4D,
00239 /* U+039D */ 0x4E,
00240 /* U+039E */ 0x58,
00241 /* U+039F */ 0x4F,
00242 /* U+03A0 */ 0x50,
00243 /* U+03A1 */ 0x52,
00244 0x00,
00245 /* U+03A3 */ 0x53,
00246 /* U+03A4 */ 0x54,
00247 /* U+03A5 */ 0x55,
00248 /* U+03A6 */ 0x46,
00249 /* U+03A7 */ 0x43,
00250 /* U+03A8 */ 0x59,
00251 /* U+03A9 */ 0x57,
00252 0x00,
00253 0x00,
00254 0x00,
00255 0x00,
00256 0x00,
00257 0x00,
00258 0x00,
00259 /* U+03B1 */ 0x61,
00260 /* U+03B2 */ 0x62,
00261 /* U+03B3 */ 0x67,
00262 /* U+03B4 */ 0x64,
00263 /* U+03B5 */ 0x65,
00264 /* U+03B6 */ 0x7A,
00265 /* U+03B7 */ 0x68,
00266 /* U+03B8 */ 0x71,
00267 /* U+03B9 */ 0x69,
00268 /* U+03BA */ 0x6B,
00269 /* U+03BB */ 0x6C,
00270 /* U+03BC */ 0x6D,
00271 /* U+03BD */ 0x6E,
00272 /* U+03BE */ 0x78,
00273 /* U+03BF */ 0x6F,
00274 /* U+03C0 */ 0x70,
00275 /* U+03C1 */ 0x72,
00276 /* U+03C2 */ 0x56,
00277 /* U+03C3 */ 0x73,
00278 /* U+03C4 */ 0x74,
00279 /* U+03C5 */ 0x75,
00280 /* U+03C6 */ 0x66,
00281 /* U+03C7 */ 0x63,
00282 /* U+03C8 */ 0x79,
00283 /* U+03C9 */ 0x77,
00284 0x00,
00285 0x00,
00286 0x00,
00287 0x00,
00288 0x00,
00289 0x00,
00290 0x00,
00291 /* U+03D1 */ 0x4A,
00292 /* U+03D2 */ (char) 0xA1,
00293 0x00,
00294 0x00,
00295 /* U+03D5 */ 0x6A,
00296 /* U+03D6 */ 0x76,
00297 };
00298
00299 static const char unicode_to_symbol_1b_2022[] = {
00300 /* U+2022 */ (char) 0xB7,
00301 0x00,
00302 0x00,
00303 0x00,
00304 /* U+2026 */ (char) 0xBC,
00305 0x00,
00306 0x00,
00307 0x00,
00308 0x00,
00309 0x00,
00310 0x00,
00311 0x00,
00312 0x00,
```

```
00313 0x00,
00314 0x00,
00315 0x00,
00316 /* U+2032 */ (char) 0xA2,
00317 /* U+2033 */ (char) 0xB2,
00318 0x00,
00319 0x00,
00320 0x00,
00321 0x00,
00322 0x00,
00323 0x00,
00324 0x00,
00325 0x00,
00326 0x00,
00327 0x00,
00328 0x00,
00329 0x00,
00330 0x00,
00331 0x00,
00332 0x00,
00333 0x00,
00334 /* U+2044 */ (char) 0xA4,
00335 0x00,
00336 0x00,
00337 0x00,
00338 0x00,
00339 0x00,
00340 0x00,
00341 0x00,
00342 0x00,
00343 0x00,
00344 0x00,
00345 0x00,
00346 0x00,
00347 0x00,
00348 0x00,
00349 0x00,
00350 0x00,
00351 0x00,
00352 0x00,
00353 0x00,
00354 0x00,
00355 0x00,
00356 0x00,
00357 0x00,
00358 0x00,
00359 0x00,
00360 0x00,
00361 0x00,
00362 0x00,
00363 0x00,
00364 0x00,
00365 0x00,
00366 0x00,
00367 0x00,
00368 0x00,
00369 0x00,
00370 0x00,
00371 0x00,
00372 0x00,
00373 0x00,
00374 0x00,
00375 0x00,
00376 0x00,
00377 0x00,
00378 0x00,
00379 0x00,
00380 0x00,
00381 0x00,
00382 0x00,
00383 0x00,
00384 0x00,
00385 0x00,
00386 0x00,
00387 0x00,
00388 0x00,
00389 0x00,
00390 0x00,
00391 0x00,
00392 0x00,
00393 0x00,
00394 0x00,
00395 0x00,
00396 0x00,
00397 0x00,
00398 0x00,
00399 0x00,
```

```
00400 0x00,
00401 0x00,
00402 0x00,
00403 0x00,
00404 0x00,
00405 0x00,
00406 0x00,
00407 0x00,
00408 0x00,
00409 0x00,
00410 0x00,
00411 0x00,
00412 0x00,
00413 0x00,
00414 0x00,
00415 0x00,
00416 0x00,
00417 0x00,
00418 0x00,
00419 0x00,
00420 0x00,
00421 0x00,
00422 0x00,
00423 0x00,
00424 0x00,
00425 0x00,
00426 0x00,
00427 0x00,
00428 0x00,
00429 0x00,
00430 0x00,
00431 0x00,
00432 0x00,
00433 0x00,
00434 0x00,
00435 0x00,
00436 0x00,
00437 0x00,
00438 /* U+20AC */ (char) 0xA0,
00439 0x00,
00440 0x00,
00441 0x00,
00442 0x00,
00443 0x00,
00444 0x00,
00445 0x00,
00446 0x00,
00447 0x00,
00448 0x00,
00449 0x00,
00450 0x00,
00451 0x00,
00452 0x00,
00453 0x00,
00454 0x00,
00455 0x00,
00456 0x00,
00457 0x00,
00458 0x00,
00459 0x00,
00460 0x00,
00461 0x00,
00462 0x00,
00463 0x00,
00464 0x00,
00465 0x00,
00466 0x00,
00467 0x00,
00468 0x00,
00469 0x00,
00470 0x00,
00471 0x00,
00472 0x00,
00473 0x00,
00474 0x00,
00475 0x00,
00476 0x00,
00477 0x00,
00478 0x00,
00479 0x00,
00480 0x00,
00481 0x00,
00482 0x00,
00483 0x00,
00484 0x00,
00485 0x00,
00486 0x00,
```

```
00487 0x00,
00488 0x00,
00489 0x00,
00490 0x00,
00491 0x00,
00492 0x00,
00493 0x00,
00494 0x00,
00495 0x00,
00496 0x00,
00497 0x00,
00498 0x00,
00499 0x00,
00500 0x00,
00501 0x00,
00502 0x00,
00503 0x00,
00504 0x00,
00505 0x00,
00506 0x00,
00507 0x00,
00508 0x00,
00509 0x00,
00510 0x00,
00511 0x00,
00512 0x00,
00513 0x00,
00514 0x00,
00515 0x00,
00516 0x00,
00517 0x00,
00518 0x00,
00519 0x00,
00520 0x00,
00521 0x00,
00522 0x00,
00523 0x00,
00524 0x00,
00525 0x00,
00526 0x00,
00527 0x00,
00528 0x00,
00529 0x00,
00530 0x00,
00531 0x00,
00532 0x00,
00533 0x00,
00534 0x00,
00535 0x00,
00536 0x00,
00537 0x00,
00538 0x00,
00539 /* U+2111 */ (char) 0xC1,
00540 0x00,
00541 0x00,
00542 0x00,
00543 0x00,
00544 0x00,
00545 0x00,
00546 /* U+2118 */ (char) 0xC3,
00547 0x00,
00548 0x00,
00549 0x00,
00550 /* U+211C */ (char) 0xC2,
00551 0x00,
00552 0x00,
00553 0x00,
00554 0x00,
00555 0x00,
00556 0x00,
00557 0x00,
00558 0x00,
00559 0x00,
00560 /* U+2126 */ 0x57,
00561 0x00,
00562 0x00,
00563 0x00,
00564 0x00,
00565 0x00,
00566 0x00,
00567 0x00,
00568 0x00,
00569 0x00,
00570 0x00,
00571 0x00,
00572 0x00,
00573 0x00,
```

```
00574 0x00,  
00575 /* U+2135 */ (char) 0xC0,  
00576 0x00,  
00577 0x00,  
00578 0x00,  
00579 0x00,  
00580 0x00,  
00581 0x00,  
00582 0x00,  
00583 0x00,  
00584 0x00,  
00585 0x00,  
00586 0x00,  
00587 0x00,  
00588 0x00,  
00589 0x00,  
00590 0x00,  
00591 0x00,  
00592 0x00,  
00593 0x00,  
00594 0x00,  
00595 0x00,  
00596 0x00,  
00597 0x00,  
00598 0x00,  
00599 0x00,  
00600 0x00,  
00601 0x00,  
00602 0x00,  
00603 0x00,  
00604 0x00,  
00605 0x00,  
00606 0x00,  
00607 0x00,  
00608 0x00,  
00609 0x00,  
00610 0x00,  
00611 0x00,  
00612 0x00,  
00613 0x00,  
00614 0x00,  
00615 0x00,  
00616 0x00,  
00617 0x00,  
00618 0x00,  
00619 0x00,  
00620 0x00,  
00621 0x00,  
00622 0x00,  
00623 0x00,  
00624 0x00,  
00625 0x00,  
00626 0x00,  
00627 0x00,  
00628 0x00,  
00629 0x00,  
00630 0x00,  
00631 0x00,  
00632 0x00,  
00633 0x00,  
00634 0x00,  
00635 0x00,  
00636 0x00,  
00637 0x00,  
00638 0x00,  
00639 0x00,  
00640 0x00,  
00641 0x00,  
00642 0x00,  
00643 0x00,  
00644 0x00,  
00645 0x00,  
00646 0x00,  
00647 0x00,  
00648 0x00,  
00649 0x00,  
00650 0x00,  
00651 0x00,  
00652 0x00,  
00653 0x00,  
00654 0x00,  
00655 0x00,  
00656 0x00,  
00657 0x00,  
00658 0x00,  
00659 0x00,  
00660 0x00,
```

```
00661 0x00,
00662 0x00,
00663 0x00,
00664 0x00,
00665 0x00,
00666 /* U+2190 */ (char)0xAC,
00667 /* U+2191 */ (char)0xAD,
00668 /* U+2192 */ (char)0xAE,
00669 /* U+2193 */ (char)0xAF,
00670 /* U+2194 */ (char)0xAB,
00671 0x00,
00672 0x00,
00673 0x00,
00674 0x00,
00675 0x00,
00676 0x00,
00677 0x00,
00678 0x00,
00679 0x00,
00680 0x00,
00681 0x00,
00682 0x00,
00683 0x00,
00684 0x00,
00685 0x00,
00686 0x00,
00687 0x00,
00688 0x00,
00689 0x00,
00690 0x00,
00691 0x00,
00692 0x00,
00693 0x00,
00694 0x00,
00695 0x00,
00696 0x00,
00697 0x00,
00698 0x00,
00699 0x00,
00700 0x00,
00701 0x00,
00702 0x00,
00703 /* U+21B5 */ (char)0xBF,
00704 0x00,
00705 0x00,
00706 0x00,
00707 0x00,
00708 0x00,
00709 0x00,
00710 0x00,
00711 0x00,
00712 0x00,
00713 0x00,
00714 0x00,
00715 0x00,
00716 0x00,
00717 0x00,
00718 0x00,
00719 0x00,
00720 0x00,
00721 0x00,
00722 0x00,
00723 0x00,
00724 0x00,
00725 0x00,
00726 0x00,
00727 0x00,
00728 0x00,
00729 0x00,
00730 /* U+21D0 */ (char)0xDC,
00731 /* U+21D1 */ (char)0xDD,
00732 /* U+21D2 */ (char)0xDE,
00733 /* U+21D3 */ (char)0xDF,
00734 /* U+21D4 */ (char)0xDB,
00735 0x00,
00736 0x00,
00737 0x00,
00738 0x00,
00739 0x00,
00740 0x00,
00741 0x00,
00742 0x00,
00743 0x00,
00744 0x00,
00745 0x00,
00746 0x00,
00747 0x00,
```

```
00748 0x00,
00749 0x00,
00750 0x00,
00751 0x00,
00752 0x00,
00753 0x00,
00754 0x00,
00755 0x00,
00756 0x00,
00757 0x00,
00758 0x00,
00759 0x00,
00760 0x00,
00761 0x00,
00762 0x00,
00763 0x00,
00764 0x00,
00765 0x00,
00766 0x00,
00767 0x00,
00768 0x00,
00769 0x00,
00770 0x00,
00771 0x00,
00772 0x00,
00773 0x00,
00774 0x00,
00775 0x00,
00776 0x00,
00777 0x00,
00778 /* U+2200 */ 0x22,
00779 0x00,
00780 /* U+2202 */ (char) 0xB6,
00781 /* U+2203 */ 0x24,
00782 0x00,
00783 /* U+2205 */ (char) 0xC6,
00784 /* U+2206 */ 0x44,
00785 /* U+2207 */ (char) 0xD1,
00786 /* U+2208 */ (char) 0xCE,
00787 /* U+2209 */ (char) 0xCF,
00788 0x00,
00789 /* U+220B */ 0x27,
00790 0x00,
00791 0x00,
00792 0x00,
00793 /* U+220F */ (char) 0xD5,
00794 0x00,
00795 /* U+2211 */ (char) 0xE5,
00796 /* U+2212 */ 0x2D,
00797 0x00,
00798 0x00,
00799 /* U+2215 */ (char) 0xA4,
00800 0x00,
00801 /* U+2217 */ 0x2A,
00802 0x00,
00803 0x00,
00804 /* U+221A */ (char) 0xD6,
00805 0x00,
00806 0x00,
00807 /* U+221D */ (char) 0xB5,
00808 /* U+221E */ (char) 0xA5,
00809 0x00,
00810 /* U+2220 */ (char) 0xD0,
00811 0x00,
00812 0x00,
00813 0x00,
00814 0x00,
00815 0x00,
00816 0x00,
00817 /* U+2227 */ (char) 0xD9,
00818 /* U+2228 */ (char) 0xDA,
00819 /* U+2229 */ (char) 0xC7,
00820 /* U+222A */ (char) 0xC8,
00821 /* U+222B */ (char) 0xF2,
00822 0x00,
00823 0x00,
00824 0x00,
00825 0x00,
00826 0x00,
00827 0x00,
00828 0x00,
00829 0x00,
00830 /* U+2234 */ 0x5C,
00831 0x00,
00832 0x00,
00833 0x00,
00834 0x00,
```

```
00835 0x00,
00836 0x00,
00837 0x00,
00838 /* U+223C */ 0x7E,
00839 0x00,
00840 0x00,
00841 0x00,
00842 0x00,
00843 0x00,
00844 0x00,
00845 0x00,
00846 0x00,
00847 /* U+2245 */ 0x40,
00848 0x00,
00849 0x00,
00850 /* U+2248 */ (char) 0xBB,
00851 0x00,
00852 0x00,
00853 0x00,
00854 0x00,
00855 0x00,
00856 0x00,
00857 0x00,
00858 0x00,
00859 0x00,
00860 0x00,
00861 0x00,
00862 0x00,
00863 0x00,
00864 0x00,
00865 0x00,
00866 0x00,
00867 0x00,
00868 0x00,
00869 0x00,
00870 0x00,
00871 0x00,
00872 0x00,
00873 0x00,
00874 /* U+2260 */ (char) 0xB9,
00875 /* U+2261 */ (char) 0xBA,
00876 0x00,
00877 0x00,
00878 /* U+2264 */ (char) 0xA3,
00879 /* U+2265 */ (char) 0xB3,
00880 0x00,
00881 0x00,
00882 0x00,
00883 0x00,
00884 0x00,
00885 0x00,
00886 0x00,
00887 0x00,
00888 0x00,
00889 0x00,
00890 0x00,
00891 0x00,
00892 0x00,
00893 0x00,
00894 0x00,
00895 0x00,
00896 0x00,
00897 0x00,
00898 0x00,
00899 0x00,
00900 0x00,
00901 0x00,
00902 0x00,
00903 0x00,
00904 0x00,
00905 0x00,
00906 0x00,
00907 0x00,
00908 /* U+2282 */ (char) 0xCC,
00909 /* U+2283 */ (char) 0xC9,
00910 /* U+2284 */ (char) 0xCB,
00911 0x00,
00912 /* U+2286 */ (char) 0xCD,
00913 /* U+2287 */ (char) 0xCA,
00914 0x00,
00915 0x00,
00916 0x00,
00917 0x00,
00918 0x00,
00919 0x00,
00920 0x00,
00921 0x00,
```



```
00922 0x00,
00923 0x00,
00924 0x00,
00925 0x00,
00926 0x00,
00927 /* U+2295 */ (char) 0xC5,
00928 0x00,
00929 /* U+2297 */ (char) 0xC4,
00930 0x00,
00931 0x00,
00932 0x00,
00933 0x00,
00934 0x00,
00935 0x00,
00936 0x00,
00937 0x00,
00938 0x00,
00939 0x00,
00940 0x00,
00941 0x00,
00942 0x00,
00943 /* U+22A5 */ 0x5E,
00944 0x00,
00945 0x00,
00946 0x00,
00947 0x00,
00948 0x00,
00949 0x00,
00950 0x00,
00951 0x00,
00952 0x00,
00953 0x00,
00954 0x00,
00955 0x00,
00956 0x00,
00957 0x00,
00958 0x00,
00959 0x00,
00960 0x00,
00961 0x00,
00962 0x00,
00963 0x00,
00964 0x00,
00965 0x00,
00966 0x00,
00967 0x00,
00968 0x00,
00969 0x00,
00970 0x00,
00971 0x00,
00972 0x00,
00973 0x00,
00974 0x00,
00975 /* U+22C5 */ (char) 0xD7,
00976 0x00,
00977 0x00,
00978 0x00,
00979 0x00,
00980 0x00,
00981 0x00,
00982 0x00,
00983 0x00,
00984 0x00,
00985 0x00,
00986 0x00,
00987 0x00,
00988 0x00,
00989 0x00,
00990 0x00,
00991 0x00,
00992 0x00,
00993 0x00,
00994 0x00,
00995 0x00,
00996 0x00,
00997 0x00,
00998 0x00,
00999 0x00,
01000 0x00,
01001 0x00,
01002 0x00,
01003 0x00,
01004 0x00,
01005 0x00,
01006 0x00,
01007 0x00,
01008 0x00,
```

```
01009 0x00,
01010 0x00,
01011 0x00,
01012 0x00,
01013 0x00,
01014 0x00,
01015 0x00,
01016 0x00,
01017 0x00,
01018 0x00,
01019 0x00,
01020 0x00,
01021 0x00,
01022 0x00,
01023 0x00,
01024 0x00,
01025 0x00,
01026 0x00,
01027 0x00,
01028 0x00,
01029 0x00,
01030 0x00,
01031 0x00,
01032 0x00,
01033 0x00,
01034 0x00,
01035 0x00,
01036 0x00,
01037 0x00,
01038 0x00,
01039 0x00,
01040 0x00,
01041 0x00,
01042 0x00,
01043 0x00,
01044 0x00,
01045 0x00,
01046 0x00,
01047 0x00,
01048 0x00,
01049 0x00,
01050 0x00,
01051 0x00,
01052 0x00,
01053 0x00,
01054 0x00,
01055 0x00,
01056 0x00,
01057 0x00,
01058 0x00,
01059 0x00,
01060 0x00,
01061 0x00,
01062 0x00,
01063 0x00,
01064 0x00,
01065 0x00,
01066 /* U+2320 */ (char)0xF3,
01067 /* U+2321 */ (char)0xF5,
01068 0x00,
01069 0x00,
01070 0x00,
01071 0x00,
01072 0x00,
01073 0x00,
01074 0x00,
01075 /* U+2329 */ (char)0xE1,
01076 /* U+232A */ (char)0xF1,
01077 };
01078
01079 static const char unicode_to_symbol_1b_25CA[] = {
01080 /* U+25CA */ (char)0xE0,
01081 };
01082
01083 static const char unicode_to_symbol_1b_2660[] = {
01084 /* U+2660 */ (char)0xAA,
01085 0x00,
01086 0x00,
01087 /* U+2663 */ (char)0xA7,
01088 0x00,
01089 /* U+2665 */ (char)0xA9,
01090 /* U+2666 */ (char)0xA8,
01091 };
01092
01093 static const char unicode_to_symbol_1b_F6D9[] = {
01094 /* U+F6D9 */ (char)0xD3,
01095 /* U+F6DA */ (char)0xD2,
```

```

01096 /* U+F6DB */ (char)0xD4,
01097 };
01098
01099 static const char unicode_to_symbol_1b_F8E5[] = {
01100 /* U+F8E5 */ 0x60,
01101 /* U+F8E6 */ (char)0xBD,
01102 /* U+F8E7 */ (char)0xBE,
01103 /* U+F8E8 */ (char)0xE2,
01104 /* U+F8E9 */ (char)0xE3,
01105 /* U+F8EA */ (char)0xE4,
01106 /* U+F8EB */ (char)0xE6,
01107 /* U+F8EC */ (char)0xE7,
01108 /* U+F8ED */ (char)0xE8,
01109 /* U+F8EE */ (char)0xE9,
01110 /* U+F8EF */ (char)0xEA,
01111 /* U+F8F0 */ (char)0xEB,
01112 /* U+F8F1 */ (char)0xEC,
01113 /* U+F8F2 */ (char)0xED,
01114 /* U+F8F3 */ (char)0xEE,
01115 /* U+F8F4 */ (char)0xEF,
01116 /* U+F8F5 */ (char)0xF4,
01117 /* U+F8F6 */ (char)0xF6,
01118 /* U+F8F7 */ (char)0xF7,
01119 /* U+F8F8 */ (char)0xF8,
01120 /* U+F8F9 */ (char)0xF9,
01121 /* U+F8FA */ (char)0xFA,
01122 /* U+F8FB */ (char)0xFB,
01123 /* U+F8FC */ (char)0xFC,
01124 /* U+F8FD */ (char)0xFD,
01125 /* U+F8FE */ (char)0xFE,
01126 };

```

12.256 imKStoUCS.c

```

00001 /* $XFree86: xc/lib/X11/imKStoUCS.c,v 1.5 2003/11/17 22:20:11 dawes Exp $ */
00002
00003 #include "Xlibint.h"
00004 #include "Ximint.h"
00005
00006 static unsigned short const keysym_to_unicode_1a1_1ff[] = {
00007     0x0104, 0x02d8, 0x0141, 0x0000, 0x013d, 0x015a, 0x0000, /* 0x01a0-0x01a7 */
00008     0x0000, 0x0160, 0x015e, 0x0164, 0x0179, 0x0000, 0x017d, 0x017b, /* 0x01a8-0x01af */
00009     0x0000, 0x0105, 0x02db, 0x0142, 0x0000, 0x013e, 0x015b, 0x02c7, /* 0x01b0-0x01b7 */
00010     0x0000, 0x0161, 0x015f, 0x0165, 0x017a, 0x02dd, 0x017e, 0x017c, /* 0x01b8-0x01bf */
00011     0x0154, 0x0000, 0x0000, 0x0102, 0x0000, 0x0139, 0x0106, 0x0000, /* 0x01c0-0x01c7 */
00012     0x010c, 0x0000, 0x0118, 0x0000, 0x011a, 0x0000, 0x0000, 0x010e, /* 0x01c8-0x01cf */
00013     0x0110, 0x0143, 0x0147, 0x0000, 0x0000, 0x0150, 0x0000, 0x0000, /* 0x01d0-0x01d7 */
00014     0x0158, 0x016e, 0x0000, 0x0170, 0x0000, 0x0000, 0x0162, 0x0000, /* 0x01d8-0x01df */
00015     0x0155, 0x0000, 0x0000, 0x0103, 0x0000, 0x013a, 0x0107, 0x0000, /* 0x01e0-0x01e7 */
00016     0x010d, 0x0000, 0x0119, 0x0000, 0x011b, 0x0000, 0x0000, 0x010f, /* 0x01e8-0x01ef */
00017     0x0111, 0x0144, 0x0148, 0x0000, 0x0000, 0x0151, 0x0000, 0x0000, /* 0x01f0-0x01f7 */
00018     0x0159, 0x016f, 0x0000, 0x0171, 0x0000, 0x0000, 0x0163, 0x02d9, /* 0x01f8-0x01ff */
00019 };
00020
00021 static unsigned short const keysym_to_unicode_2a1_2fe[] = {
00022     0x0126, 0x0000, 0x0000, 0x0000, 0x0000, 0x0124, 0x0000, /* 0x02a0-0x02a7 */
00023     0x0000, 0x0130, 0x0000, 0x011e, 0x0134, 0x0000, 0x0000, 0x0000, /* 0x02a8-0x02af */
00024     0x0000, 0x0127, 0x0000, 0x0000, 0x0000, 0x0000, 0x0125, 0x0000, /* 0x02b0-0x02b7 */
00025     0x0000, 0x0131, 0x0000, 0x011f, 0x0135, 0x0000, 0x0000, 0x0000, /* 0x02b8-0x02bf */
00026     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x010a, 0x0108, 0x0000, /* 0x02c0-0x02c7 */
00027     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x02c8-0x02cf */
00028     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0120, 0x0000, 0x0000, /* 0x02d0-0x02d7 */
00029     0x011c, 0x0000, 0x0000, 0x0000, 0x0000, 0x016c, 0x015c, 0x0000, /* 0x02d8-0x02df */
00030     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x010b, 0x0109, 0x0000, /* 0x02e0-0x02e7 */
00031     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x02e8-0x02ef */
00032     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0121, 0x0000, 0x0000, /* 0x02f0-0x02f7 */
00033     0x011d, 0x0000, 0x0000, 0x0000, 0x0000, 0x016d, 0x015d, /* 0x02f8-0x02ff */
00034 };
00035
00036 static unsigned short const keysym_to_unicode_3a2_3fe[] = {
00037     0x0138, 0x0156, 0x0000, 0x0128, 0x013b, 0x0000, /* 0x03a0-0x03a7 */
00038     0x0000, 0x0000, 0x0112, 0x0122, 0x0166, 0x0000, 0x0000, 0x0000, /* 0x03a8-0x03af */
00039     0x0000, 0x0000, 0x0000, 0x0157, 0x0000, 0x0129, 0x013c, 0x0000, /* 0x03b0-0x03b7 */
00040     0x0000, 0x0000, 0x0000, 0x0123, 0x0167, 0x014a, 0x0000, 0x014b, /* 0x03b8-0x03bf */
00041     0x0100, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x012e, /* 0x03c0-0x03c7 */
00042     0x0000, 0x0000, 0x0000, 0x0000, 0x0116, 0x0000, 0x0000, 0x012a, /* 0x03c8-0x03cf */
00043     0x0000, 0x0145, 0x014c, 0x0136, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x03d0-0x03d7 */
00044     0x0000, 0x0172, 0x0000, 0x0000, 0x0000, 0x0168, 0x016a, 0x0000, /* 0x03d8-0x03df */
00045     0x0101, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x012f, /* 0x03e0-0x03e7 */
00046     0x0000, 0x0000, 0x0000, 0x0000, 0x0117, 0x0000, 0x0000, 0x012b, /* 0x03e8-0x03ef */
00047     0x0000, 0x0146, 0x014d, 0x0137, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x03f0-0x03f7 */
00048     0x0000, 0x0173, 0x0000, 0x0000, 0x0000, 0x0169, 0x016b, /* 0x03f8-0x03ff */
00049 };
00050
00051 static unsigned short const keysym_to_unicode_4a1_4df[] = {

```

```
00052         0x3002, 0x3008, 0x3009, 0x3001, 0x30fb, 0x30f2, 0x30a1, /* 0x04a0-0x04a7 */
00053         0x30a3, 0x30a5, 0x30a7, 0x30a9, 0x30e3, 0x30e5, 0x30e7, 0x30c3, /* 0x04a8-0x04af */
00054         0x30fc, 0x30a2, 0x30a4, 0x30a6, 0x30aa, 0x30ab, 0x30ad, /* 0x04b0-0x04b7 */
00055         0x30af, 0x30b1, 0x30b3, 0x30b5, 0x30b7, 0x30b9, 0x30bb, 0x30bd, /* 0x04b8-0x04bf */
00056         0x30bf, 0x30c1, 0x30c4, 0x30c6, 0x30c8, 0x30ca, 0x30cb, 0x30cc, /* 0x04c0-0x04c7 */
00057         0x30cd, 0x30ce, 0x30cf, 0x30d2, 0x30d5, 0x30db, 0x30de, /* 0x04c8-0x04cf */
00058         0x30df, 0x30e0, 0x30e1, 0x30e2, 0x30e4, 0x30e6, 0x30e8, 0x30e9, /* 0x04d0-0x04d7 */
00059         0x30ea, 0x30eb, 0x30ec, 0x30ed, 0x30ef, 0x30f3, 0x309b, 0x309c /* 0x04d8-0x04df */
00060     };
00061
00062     static unsigned short const keysym_to_unicode_590_5fe[] = {
00063         0x06f0, 0x06f1, 0x06f2, 0x06f3, 0x06f4, 0x06f5, 0x06f6, 0x06f7, /* 0x0590-0x0597 */
00064         0x06f8, 0x06f9, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x0598-0x059f */
00065         0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x066a, 0x0670, 0x0679, /* 0x05a0-0x05a7 */
00066
00067         0x067e, 0x0686, 0x0688, 0x0691, 0x060c, 0x0000, 0x06d4, 0x0000, /* 0x05ac-0x05af */
00068         0x0660, 0x0661, 0x0662, 0x0663, 0x0664, 0x0665, 0x0666, 0x0667, /* 0x05b0-0x05b7 */
00069         0x0668, 0x0669, 0x0000, 0x061b, 0x0000, 0x0000, 0x0000, 0x061f, /* 0x05b8-0x05bf */
00070         0x0000, 0x0621, 0x0622, 0x0623, 0x0624, 0x0625, 0x0626, 0x0627, /* 0x05c0-0x05c7 */
00071         0x0628, 0x0629, 0x062a, 0x062b, 0x062c, 0x062d, 0x062e, 0x062f, /* 0x05c8-0x05cf */
00072         0x0630, 0x0631, 0x0632, 0x0633, 0x0634, 0x0635, 0x0636, 0x0637, /* 0x05d0-0x05d7 */
00073         0x0638, 0x0639, 0x063a, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x05d8-0x05df */
00074         0x0640, 0x0641, 0x0642, 0x0643, 0x0644, 0x0645, 0x0646, 0x0647, /* 0x05e0-0x05e7 */
00075         0x0648, 0x0649, 0x064a, 0x064b, 0x064c, 0x064d, 0x064e, 0x064f, /* 0x05e8-0x05ef */
00076         0x0650, 0x0651, 0x0652, 0x0653, 0x0654, 0x0655, 0x0698, 0x06a4, /* 0x05f0-0x05f7 */
00077         0x06a9, 0x06af, 0x06ba, 0x06be, 0x06cc, 0x06d2, 0x06c1, /* 0x05f8-0x05fe */
00078     };
00079
00080     static unsigned short const keysym_to_unicode_680_6ff[] = {
00081         0x0492, 0x0496, 0x049a, 0x049c, 0x04a2, 0x04ae, 0x04b0, 0x04b2, /* 0x0680-0x0687 */
00082         0x04b6, 0x04b8, 0x04ba, 0x0000, 0x04d8, 0x04e2, 0x04e8, 0x04ee, /* 0x0688-0x068f */
00083         0x0493, 0x0497, 0x049b, 0x049d, 0x04a3, 0x04af, 0x04b1, 0x04b3, /* 0x0690-0x0697 */
00084         0x04b7, 0x04b9, 0x04bb, 0x0000, 0x04d9, 0x04e3, 0x04e9, 0x04ef, /* 0x0698-0x069f */
00085         0x0000, 0x0452, 0x0453, 0x0451, 0x0454, 0x0455, 0x0456, 0x0457, /* 0x06a0-0x06a7 */
00086         0x0458, 0x0459, 0x045a, 0x045b, 0x045c, 0x0491, 0x045e, 0x045f, /* 0x06a8-0x06af */
00087         0x2116, 0x0402, 0x0403, 0x0401, 0x0404, 0x0405, 0x0406, 0x0407, /* 0x06b0-0x06b7 */
00088         0x0408, 0x0409, 0x040a, 0x040b, 0x040c, 0x0490, 0x040e, 0x040f, /* 0x06b8-0x06bf */
00089         0x044e, 0x0430, 0x0431, 0x0446, 0x0434, 0x0435, 0x0444, 0x0433, /* 0x06c0-0x06c7 */
00090         0x0445, 0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e, /* 0x06c8-0x06cf */
00091         0x043f, 0x044f, 0x0440, 0x0441, 0x0442, 0x0443, 0x0436, 0x0432, /* 0x06d0-0x06d7 */
00092         0x044c, 0x044b, 0x0437, 0x0448, 0x044d, 0x0449, 0x0447, 0x044a, /* 0x06d8-0x06df */
00093         0x042e, 0x0410, 0x0411, 0x0426, 0x0414, 0x0415, 0x0424, 0x0413, /* 0x06e0-0x06ef */
00094         0x0425, 0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e, /* 0x06f0-0x06ff */
00095         0x041f, 0x042f, 0x0420, 0x0421, 0x0422, 0x0423, 0x0416, 0x0412, /* 0x06f0-0x06ff */
00096         0x042c, 0x042b, 0x0417, 0x0428, 0x042d, 0x0429, 0x0427, 0x042a /* 0x06f8-0x06ff */
00097     };
00098
00099     static unsigned short const keysym_to_unicode_7a1_7f9[] = {
00100         0x0386, 0x0388, 0x0389, 0x038a, 0x03aa, 0x0000, 0x038c, /* 0x07a0-0x07a7 */
00101         0x038e, 0x03ab, 0x0000, 0x038f, 0x0000, 0x0000, 0x0385, 0x2015, /* 0x07a8-0x07af */
00102         0x0000, 0x03ac, 0x03ad, 0x03ae, 0x03af, 0x03ca, 0x0390, 0x03cc, /* 0x07b0-0x07b7 */
00103         0x03cd, 0x03cb, 0x03b0, 0x03ce, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x07b8-0x07bf */
00104         0x0000, 0x0391, 0x0392, 0x0393, 0x0394, 0x0395, 0x0396, 0x0397, /* 0x07c0-0x07c7 */
00105         0x0398, 0x0399, 0x039a, 0x039b, 0x039c, 0x039d, 0x039e, 0x039f, /* 0x07c8-0x07cf */
00106         0x03a0, 0x03a1, 0x03a3, 0x0000, 0x03a4, 0x03a5, 0x03a6, 0x03a7, /* 0x07d0-0x07d7 */
00107         0x03a8, 0x03a9, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x07d8-0x07df */
00108         0x0000, 0x03b1, 0x03b2, 0x03b3, 0x03b4, 0x03b5, 0x03b6, 0x03b7, /* 0x07e0-0x07ef */
00109         0x03b8, 0x03b9, 0x03ba, 0x03bb, 0x03bc, 0x03bd, 0x03be, 0x03bf, /* 0x07f0-0x07ff */
00110         0x03c0, 0x03c1, 0x03c3, 0x03c2, 0x03c4, 0x03c5, 0x03c6, 0x03c7, /* 0x07f0-0x07ff */
00111         0x03c8, 0x03c9
00112     };
00113
00114     static unsigned short const keysym_to_unicode_8a4_8fe[] = {
00115         0x231d, 0x231e, 0x231f, 0x0000, 0x0000, 0x0000, 0x2320, 0x2321, /* 0x08a0-0x08a7 */
00116         0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x08a8-0x08af */
00117         0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x08b0-0x08b7 */
00118         0x0000, 0x0000, 0x0000, 0x0000, 0x2264, 0x2260, 0x2265, 0x222b, /* 0x08b8-0x08bf */
00119         0x2234, 0x0000, 0x221e, 0x0000, 0x0000, 0x2207, 0x0000, 0x0000, /* 0x08c0-0x08cf */
00120         0x2245, 0x2246, 0x0000, 0x0000, 0x0000, 0x0000, 0x22a2, 0x0000, /* 0x08d0-0x08df */
00121         0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x221a, 0x0000, /* 0x08e0-0x08ef */
00122         0x0000, 0x0000, 0x2282, 0x2283, 0x2229, 0x222a, 0x2227, 0x2228, /* 0x08f0-0x08ff */
00123         0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x08f0-0x08ff */
00124         0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x08f0-0x08ff */
00125         0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0192, 0x0000, /* 0x08f0-0x08ff */
00126         0x0000, 0x0000, 0x0000, 0x2190, 0x2191, 0x2192, 0x2193, /* 0x08f8-0x08ff */
00127     };
00128
00129     static unsigned short const keysym_to_unicode_9df_9f8[] = {
00130         0x2666, 0x25a6, 0x2409, 0x240c, 0x240d, 0x240a, 0x0000, 0x2422, /* 0x09d8-0x09df */
00131         0x240a, 0x240b, 0x2518, 0x2510, 0x250c, 0x2514, 0x253c, 0x2500, /* 0x09e0-0x09ef */
00132         0x0000, 0x0000, 0x0000, 0x0000, 0x251c, 0x2524, 0x2534, 0x252c, /* 0x09f0-0x09ff */
00133         0x2502
00134     };
00135
00136     static unsigned short const keysym_to_unicode_aal_afe[] = {
00137         0x2003, 0x2002, 0x2004, 0x2005, 0x2007, 0x2008, 0x2009, /* 0x0aa0-0x0aa7 */
00138     };
```

```
00139     0x200a, 0x2014, 0x2013, 0x0000, 0x0000, 0x0000, 0x2026, 0x2025, /* 0x0aa8-0x0aaf */
00140     0x2153, 0x2154, 0x2155, 0x2156, 0x2157, 0x2158, 0x2159, 0x215a, /* 0x0ab0-0x0ab7 */
00141     0x2105, 0x0000, 0x0000, 0x0000, 0x2039, 0x2024, 0x203a, 0x0000, /* 0x0ab8-0x0abf */
00142     0x0000, 0x0000, 0x0000, 0x215b, 0x215c, 0x215d, 0x215e, 0x0000, /* 0x0ac0-0x0ac7 */
00143     0x0000, 0x2122, 0x2120, 0x0000, 0x25c1, 0x25b7, 0x25cb, 0x25ad, /* 0x0ac8-0x0acf */
00144     0x2018, 0x2019, 0x201c, 0x201d, 0x211e, 0x0000, 0x2032, 0x2033, /* 0x0ad0-0x0ad7 */
00145     0x0000, 0x271d, 0x0000, 0x220e, 0x25c2, 0x2023, 0x25cf, 0x25ac, /* 0x0ad8-0x0adf */
00146     0x25e6, 0x25ab, 0x25ae, 0x25b5, 0x25bf, 0x2606, 0x2022, 0x25aa, /* 0x0ae0-0x0ae7 */
00147     0x25b4, 0x25be, 0x261a, 0x261b, 0x2663, 0x2666, 0x2665, 0x0000, /* 0x0ae8-0x0aef */
00148     0x2720, 0x2020, 0x2021, 0x2713, 0x2612, 0x266f, 0x266d, 0x2642, /* 0x0af0-0x0af7 */
00149     0x2640, 0x2121, 0x2315, 0x2117, 0x2038, 0x201a, 0x201e, /* 0x0af8-0x0aff */
00150 };
00151
00152 /* none of the APL keysyms match the Unicode characters */
00153
00154 static unsigned short const keysym_to_unicode_cdf_cfa[] = {
00155     0x2017, /* 0x0cd8-0x0cdf */
00156     0x05d0, 0x05d1, 0x05d2, 0x05d3, 0x05d4, 0x05d5, 0x05d6, 0x05d7, /* 0x0ce0-0x0ce7 */
00157     0x05d8, 0x05d9, 0x05da, 0x05db, 0x05dc, 0x05dd, 0x05de, 0x05df, /* 0x0ce8-0x0cef */
00158     0x05e0, 0x05e1, 0x05e2, 0x05e3, 0x05e4, 0x05e5, 0x05e6, 0x05e7, /* 0x0cf0-0x0cf7 */
00159     0x05e8, 0x05e9, 0x05ea, /* 0x0cf8-0x0cff */
00160 };
00161
00162 static unsigned short const keysym_to_unicode_dal_df9[] = {
00163     0x0e01, 0x0e02, 0x0e03, 0x0e04, 0x0e05, 0x0e06, 0x0e07, /* 0x0da0-0x0da7 */
00164     0x0e08, 0x0e09, 0x0e0a, 0x0e0b, 0x0e0c, 0x0e0d, 0x0e0e, 0x0e0f, /* 0x0da8-0x0daf */
00165     0x0e10, 0x0e11, 0x0e12, 0x0e13, 0x0e14, 0x0e15, 0x0e16, 0x0e17, /* 0x0db0-0x0db7 */
00166     0x0e18, 0x0e19, 0x0e1a, 0x0e1b, 0x0e1c, 0x0e1d, 0x0e1e, 0x0e1f, /* 0x0db8-0x0dbf */
00167     0x0e20, 0x0e21, 0x0e22, 0x0e23, 0x0e24, 0x0e25, 0x0e26, 0x0e27, /* 0x0dc0-0x0dc7 */
00168     0x0e28, 0x0e29, 0x0e2a, 0x0e2b, 0x0e2c, 0x0e2d, 0x0e2e, 0x0e2f, /* 0x0dc8-0x0dcf */
00169     0x0e30, 0x0e31, 0x0e32, 0x0e33, 0x0e34, 0x0e35, 0x0e36, 0x0e37, /* 0x0dd0-0x0dd7 */
00170     0x0e38, 0x0e39, 0x0e3a, 0x0000, 0x0000, 0x0000, 0x0e3e, 0x0e3f, /* 0x0dd8-0x0ddf */
00171     0x0e40, 0x0e41, 0x0e42, 0x0e43, 0x0e44, 0x0e45, 0x0e46, 0x0e47, /* 0x0de0-0x0de7 */
00172     0x0e48, 0x0e49, 0x0e4a, 0x0e4b, 0x0e4c, 0x0e4d, 0x0000, 0x0000, /* 0x0de8-0x0def */
00173     0x0e50, 0x0e51, 0x0e52, 0x0e53, 0x0e54, 0x0e55, 0x0e56, 0x0e57, /* 0x0df0-0x0df7 */
00174     0x0e58, 0x0e59, /* 0x0df8-0x0dff */
00175 };
00176
00177 static unsigned short const keysym_to_unicode_ea0_eff[] = {
00178     0x0000, 0x1101, 0x1101, 0x11aa, 0x1102, 0x11ac, 0x11ad, 0x1103, /* 0xea0-0xea7 */
00179     0x1104, 0x1105, 0x11b0, 0x11b1, 0x11b2, 0x11b3, 0x11b4, 0x11b5, /* 0xea8-0xeaf */
00180     0x11b6, 0x1106, 0x1107, 0x1108, 0x1109, 0x110a, 0x110b, /* 0xeb0-0xeb7 */
00181     0x110c, 0x110d, 0x110e, 0x110f, 0x1110, 0x1111, 0x1112, 0x1161, /* 0xeb8-0xebf */
00182     0x1162, 0x1163, 0x1164, 0x1165, 0x1166, 0x1167, 0x1168, 0x1169, /* 0xec0-0xec7 */
00183     0x116a, 0x116b, 0x116c, 0x116d, 0x116e, 0x116f, 0x1170, 0x1171, /* 0xec8-0xecf */
00184     0x1172, 0x1173, 0x1174, 0x1175, 0x11a8, 0x11a9, 0x11aa, 0x11ab, /* 0xed0-0xed7 */
00185     0x11ac, 0x11ad, 0x11ae, 0x11af, 0x11b0, 0x11b1, 0x11b2, 0x11b3, /* 0xed8-0xedf */
00186     0x11b4, 0x11b5, 0x11b6, 0x11b7, 0x11b8, 0x11b9, 0x11ba, 0x11bb, /* 0xee0-0xee7 */
00187     0x11bc, 0x11bd, 0x11be, 0x11bf, 0x11c0, 0x11c1, 0x11c2, 0x0000, /* 0xee8-0xeef */
00188     0x0000, 0x0000, 0x1140, 0x0000, 0x0000, 0x1159, 0x119e, 0x0000, /* 0xef0-0xef7 */
00189     0x11eb, 0x0000, 0x11f9, 0x0000, 0x0000, 0x0000, 0x0000, 0x20a9, /* 0xef8-0xeff */
00190 };
00191
00192 static unsigned short keysym_to_unicode_l2a1_l2fe[] = {
00193     0x1e02, 0x1e03, 0x0000, 0x0000, 0x0000, 0x1e0a, 0x0000, /* 0x12a0-0x12a7 */
00194     0x1e80, 0x0000, 0x1e82, 0x1e0b, 0x1ef2, 0x0000, 0x0000, 0x0000, /* 0x12a8-0x12af */
00195     0x1e1e, 0x1ef1, 0x0000, 0x0000, 0x1e40, 0x0000, 0x1e56, /* 0x12b0-0x12b7 */
00196     0x1e81, 0x1e57, 0x1e83, 0x1e60, 0x1ef3, 0x1e84, 0x1e85, 0x1e61, /* 0x12b8-0x12bf */
00197     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x12c0-0x12c7 */
00198     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x12c8-0x12cf */
00199     0x0174, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x1e6a, /* 0x12d0-0x12d7 */
00200     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0176, 0x0000, /* 0x12d8-0x12df */
00201     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x12e0-0x12ef */
00202     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x12f0-0x12ff */
00203     0x0175, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x1e6b, /* 0x12f0-0x12ff */
00204     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0177, /* 0x12f0-0x12ff */
00205 };
00206
00207 static unsigned short const keysym_to_unicode_l3bc_l3be[] = {
00208     0x0152, 0x0153, 0x0178, /* 0x13b8-0x13bf */
00209 };
00210
00211 static unsigned short keysym_to_unicode_l4a1_l4ff[] = {
00212     0x2741, 0x00a7, 0x0589, 0x0029, 0x0028, 0x00bb, 0x00ab, /* 0x14a0-0x14a7 */
00213     0x2014, 0x002e, 0x055d, 0x002c, 0x2013, 0x058a, 0x2026, 0x055c, /* 0x14a8-0x14af */
00214     0x055b, 0x055e, 0x0531, 0x0561, 0x0532, 0x0562, 0x0533, 0x0563, /* 0x14b0-0x14b7 */
00215     0x0534, 0x0564, 0x0535, 0x0565, 0x0536, 0x0566, 0x0537, 0x0567, /* 0x14b8-0x14bf */
00216     0x0538, 0x0568, 0x0539, 0x0569, 0x053a, 0x056a, 0x053b, 0x056b, /* 0x14c0-0x14c7 */
00217     0x053c, 0x056c, 0x053d, 0x056d, 0x053e, 0x056e, 0x053f, 0x056f, /* 0x14c8-0x14cf */
00218     0x0540, 0x0570, 0x0541, 0x0571, 0x0542, 0x0572, 0x0543, 0x0573, /* 0x14d0-0x14d7 */
00219     0x0544, 0x0574, 0x0545, 0x0575, 0x0546, 0x0576, 0x0547, 0x0577, /* 0x14d8-0x14df */
00220     0x0548, 0x0578, 0x0549, 0x0579, 0x054a, 0x057a, 0x054b, 0x057b, /* 0x14e0-0x14ef */
00221     0x054c, 0x057c, 0x054d, 0x057d, 0x054e, 0x057e, 0x054f, 0x057f, /* 0x14f0-0x14ff */
00222     0x0550, 0x0580, 0x0551, 0x0581, 0x0552, 0x0582, 0x0553, 0x0583, /* 0x14f0-0x14ff */
00223     0x0554, 0x0584, 0x0555, 0x0585, 0x0556, 0x0586, 0x2019, 0x0027, /* 0x14f8-0x14ff */
00224 };
00225
```

```
00226 static unsigned short keysym_to_unicode_15d0_15f6[] = {
00227     0x10d0, 0x10d1, 0x10d2, 0x10d3, 0x10d4, 0x10d5, 0x10d6, 0x10d7, /* 0x15d0-0x15d7 */
00228     0x10d8, 0x10d9, 0x10da, 0x10db, 0x10dc, 0x10dd, 0x10de, 0x10df, /* 0x15d8-0x15df */
00229     0x10e0, 0x10e1, 0x10e2, 0x10e3, 0x10e4, 0x10e5, 0x10e6, 0x10e7, /* 0x15e0-0x15e7 */
00230     0x10e8, 0x10e9, 0x10ea, 0x10eb, 0x10ec, 0x10ed, 0x10ee, 0x10ef, /* 0x15e8-0x15ef */
00231     0x10f0, 0x10f1, 0x10f2, 0x10f3, 0x10f4, 0x10f5, 0x10f6, /* 0x15f0-0x15f7 */
00232 };
00233
00234 static unsigned short keysym_to_unicode_16a0_16f6[] = {
00235     0x0000, 0x0000, 0xf0a2, 0x1e8a, 0x0000, 0xf0a5, 0x012c, 0xf0a7, /* 0x16a0-0x16a7 */
00236     0xf0a8, 0x01b5, 0x01e6, 0x0000, 0x0000, 0x0000, 0x0000, 0x019f, /* 0x16a8-0x16af */
00237     0x0000, 0x0000, 0xf0b2, 0x1e8b, 0x01d1, 0xf0b5, 0x012d, 0xf0b7, /* 0x16b0-0x16b7 */
00238     0xf0b8, 0x01b6, 0x01e7, 0x0000, 0x0000, 0x01d2, 0x0000, 0x0275, /* 0x16b8-0x16bf */
00239     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x018f, 0x0000, /* 0x16c0-0x16c7 */
00240     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x16c8-0x16cf */
00241     0x0000, 0x1e36, 0xf0d2, 0xf0d3, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x16d0-0x16d7 */
00242     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x16d8-0x16df */
00243     0x0000, 0x1e37, 0xf0e2, 0xf0e3, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x16e0-0x16ef */
00244     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /* 0x16f0-0x16ff */
00245     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0259, /* 0x16f0-0x16f6 */
00246 };
00247
00248 static unsigned short const keysym_to_unicode_1e9f_1eff[] = {
00249     0x0303,
00250     0x1ea0, 0x1ea1, 0x1ea2, 0x1ea3, 0x1ea4, 0x1ea5, 0x1ea6, 0x1ea7, /* 0x1ea0-0x1ea7 */
00251     0x1ea8, 0x1ea9, 0x1eaa, 0x1eab, 0x1eac, 0x1ead, 0x1eae, 0x1eaf, /* 0x1ea8-0x1eaf */
00252     0x1eb0, 0x1eb1, 0x1eb2, 0x1eb3, 0x1eb4, 0x1eb5, 0x1eb6, 0x1eb7, /* 0x1eb0-0x1eb7 */
00253     0x1eb8, 0x1eb9, 0x1eba, 0x1ebb, 0x1ebc, 0x1ebd, 0x1ebe, 0x1ebf, /* 0x1eb8-0x1ebf */
00254     0x1ec0, 0x1ec1, 0x1ec2, 0x1ec3, 0x1ec4, 0x1ec5, 0x1ec6, 0x1ec7, /* 0x1ec0-0x1ec7 */
00255     0x1ec8, 0x1ec9, 0x1eca, 0x1ecb, 0x1ecc, 0x1ecd, 0x1ece, 0x1ecf, /* 0x1ec8-0x1ecf */
00256     0x1ed0, 0x1ed1, 0x1ed2, 0x1ed3, 0x1ed4, 0x1ed5, 0x1ed6, 0x1ed7, /* 0x1ed0-0x1ed7 */
00257     0x1ed8, 0x1ed9, 0x1eda, 0x1edb, 0x1edc, 0x1edd, 0x1ede, 0x1edf, /* 0x1ed8-0x1edf */
00258     0x1ee0, 0x1ee1, 0x1ee2, 0x1ee3, 0x1ee4, 0x1ee5, 0x1ee6, 0x1ee7, /* 0x1ee0-0x1ee7 */
00259     0x1ee8, 0x1ee9, 0x1eea, 0x1eeb, 0x1eec, 0x1eed, 0x1eee, 0x1eef, /* 0x1ee8-0x1eef */
00260     0x1ef0, 0x1ef1, 0x0300, 0x0301, 0x1ef4, 0x1ef5, 0x1ef6, 0x1ef7, /* 0x1ef0-0x1ef7 */
00261     0x1ef8, 0x1ef9, 0x01a0, 0x01a1, 0x01af, 0x01b0, 0x0309, 0x0323, /* 0x1ef8-0x1eff */
00262 };
00263
00264 static unsigned short const keysym_to_unicode_20a0_20ac[] = {
00265     0x20a0, 0x20a1, 0x20a2, 0x20a3, 0x20a4, 0x20a5, 0x20a6, 0x20a7, /* 0x20a0-0x20a7 */
00266     0x20a8, 0x20a9, 0x20aa, 0x20ab, 0x20ac, /* 0x20a8-0x20af */
00267 };
00268
00269 static unsigned int
00270 KeySymToUcs4(KeySym keysym)
00271 {
00272     /* 'Unicode keysym' */
00273     if ((keysym & 0xff000000) == 0x01000000)
00274         return (keysym & 0x00ffffff);
00275
00276     if (keysym > 0 && keysym < 0x100)
00277         return keysym;
00278     else if (keysym > 0x1a0 && keysym < 0x200)
00279         return keysym_to_unicode_1a1_1ff[keysym - 0x1a1];
00280     else if (keysym > 0x2a0 && keysym < 0x2ff)
00281         return keysym_to_unicode_2a1_2fe[keysym - 0x2a1];
00282     else if (keysym > 0x3a1 && keysym < 0x3ff)
00283         return keysym_to_unicode_3a2_3fe[keysym - 0x3a2];
00284     else if (keysym > 0x4a0 && keysym < 0x4e0)
00285         return keysym_to_unicode_4a1_4df[keysym - 0x4a1];
00286     else if (keysym > 0x589 && keysym < 0x5ff)
00287         return keysym_to_unicode_590_5fe[keysym - 0x590];
00288     else if (keysym > 0x67f && keysym < 0x700)
00289         return keysym_to_unicode_680_6ff[keysym - 0x680];
00290     else if (keysym > 0x7a0 && keysym < 0x7fa)
00291         return keysym_to_unicode_7a1_7f9[keysym - 0x7a1];
00292     else if (keysym > 0x8a3 && keysym < 0x8ff)
00293         return keysym_to_unicode_8a4_8fe[keysym - 0x8a4];
00294     else if (keysym > 0x9de && keysym < 0x9f9)
00295         return keysym_to_unicode_9df_9f8[keysym - 0x9df];
00296     else if (keysym > 0xaa0 && keysym < 0xaff)
00297         return keysym_to_unicode_aal_afe[keysym - 0xaa1];
00298     else if (keysym > 0xcde && keysym < 0xcfb)
00299         return keysym_to_unicode_cdf_cfa[keysym - 0xcdf];
00300     else if (keysym > 0xda0 && keysym < 0xdfa)
00301         return keysym_to_unicode_dal_df9[keysym - 0xda1];
00302     else if (keysym > 0xe9f && keysym < 0xf00)
00303         return keysym_to_unicode_ea0_eff[keysym - 0xea0];
00304     else if (keysym > 0x12a0 && keysym < 0x12ff)
00305         return keysym_to_unicode_12a1_12fe[keysym - 0x12a1];
00306     else if (keysym > 0x13bb && keysym < 0x13bf)
00307         return keysym_to_unicode_13bc_13be[keysym - 0x13bc];
00308     else if (keysym > 0x14a0 && keysym < 0x1500)
00309         return keysym_to_unicode_14a1_14ff[keysym - 0x14a1];
00310     else if (keysym > 0x15cf && keysym < 0x15f7)
00311         return keysym_to_unicode_15d0_15f6[keysym - 0x15d0];
00312     else if (keysym > 0x169f && keysym < 0x16f7)
```

```

00313         return keySYM_to_unicode_16a0_16f6[keySYM - 0x16a0];
00314     else if (keySYM > 0x1e9e && keySYM < 0x1f00)
00315         return keySYM_to_unicode_1e9f_1eff[keySYM - 0x1e9f];
00316     else if (keySYM > 0x209f && keySYM < 0x20ad)
00317         return keySYM_to_unicode_20a0_20ac[keySYM - 0x20a0];
00318     else
00319         return 0;
00320 }

```

12.257 armSCII_8.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/armSCII_8.h,v 1.4 2003/07/15 17:33:45 pascal Exp $ */
00002
00003 /*
00004  * ARMSCII-8
00005  */
00006
00007 static const unsigned short armSCII_8_uni[96] = {
00008     /* 0xa0 */
00009     0x00a0, 0xffff, 0x0587, 0x0589, 0x0029, 0x0028, 0x00bb, 0x00ab,
00010     0x2014, 0x002e, 0x055d, 0x002c, 0x002d, 0x058a, 0x2026, 0x055c,
00011     /* 0xb0 */
00012     0x055b, 0x055e, 0x0531, 0x0561, 0x0532, 0x0562, 0x0533, 0x0563,
00013     0x0534, 0x0564, 0x0535, 0x0565, 0x0536, 0x0566, 0x0537, 0x0567,
00014     /* 0xc0 */
00015     0x0538, 0x0568, 0x0539, 0x0569, 0x053a, 0x056a, 0x053b, 0x056b,
00016     0x053c, 0x056c, 0x053d, 0x056d, 0x053e, 0x056e, 0x053f, 0x056f,
00017     /* 0xd0 */
00018     0x0540, 0x0570, 0x0541, 0x0571, 0x0542, 0x0572, 0x0543, 0x0573,
00019     0x0544, 0x0574, 0x0545, 0x0575, 0x0546, 0x0576, 0x0547, 0x0577,
00020     /* 0xe0 */
00021     0x0548, 0x0578, 0x0549, 0x0579, 0x054a, 0x057a, 0x054b, 0x057b,
00022     0x054c, 0x057c, 0x054d, 0x057d, 0x054e, 0x057e, 0x054f, 0x057f,
00023     /* 0xf0 */
00024     0x0550, 0x0580, 0x0551, 0x0581, 0x0552, 0x0582, 0x0553, 0x0583,
00025     0x0554, 0x0584, 0x0555, 0x0585, 0x0556, 0x0586, 0x055a, 0xffff,
00026 };
00027
00028 static int
00029 armSCII_8_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00030 {
00031     unsigned char c = *s;
00032     if (c < 0xa0) {
00033         *pwc = (ucs4_t) c;
00034         return 1;
00035     }
00036     else {
00037         unsigned short wc = armSCII_8_uni[c-0xa0];
00038         if (wc != 0xffff) {
00039             *pwc = (ucs4_t) wc;
00040             return 1;
00041         }
00042     }
00043     return RET_ILSEQ;
00044 }
00045
00046 static const unsigned char armSCII_8_page00[8] = {
00047     0xa5, 0xa4, 0x2a, 0x2b, 0xab, 0xac, 0xa9, 0x2f, /* 0x28-0x2f */
00048 };
00049 static const unsigned char armSCII_8_page00_1[32] = {
00050     0xa0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00051     0x00, 0x00, 0x00, 0xa7, 0x00, 0x00, 0x00, 0x00, /* 0xa8-0xaf */
00052     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb0-0xb7 */
00053     0x00, 0x00, 0x00, 0xa6, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00054 };
00055 static const unsigned char armSCII_8_page05[96] = {
00056     0x00, 0xb2, 0xb4, 0xb6, 0xb8, 0xba, 0xbc, 0xbe, /* 0x30-0x37 */
00057     0xc0, 0xc2, 0xc4, 0xc6, 0xc8, 0xca, 0xcc, 0xce, /* 0x38-0x3f */
00058     0xd0, 0xd2, 0xd4, 0xd6, 0xd8, 0xda, 0xdc, 0xde, /* 0x40-0x47 */
00059     0xe0, 0xe2, 0xe4, 0xe6, 0xe8, 0xea, 0xec, 0xee, /* 0x48-0x4f */
00060     0xf0, 0xf2, 0xf4, 0xf6, 0xfa, 0xfc, 0x00, 0x00, /* 0x50-0x57 */
00061     0x00, 0x00, 0xfe, 0xb0, 0xaf, 0xaa, 0xb1, 0x00, /* 0x58-0x5f */
00062     0x00, 0xb3, 0xb5, 0xb7, 0xb9, 0xbb, 0xbd, 0xbf, /* 0x60-0x67 */
00063     0xc1, 0xc3, 0xc5, 0xc7, 0xc9, 0xcb, 0xcd, 0xcf, /* 0x68-0x6f */
00064     0xd1, 0xd3, 0xd5, 0xd7, 0xd9, 0xdb, 0xdd, 0xdf, /* 0x70-0x77 */
00065     0xe1, 0xe3, 0xe5, 0xe7, 0xe9, 0xeb, 0xed, 0xef, /* 0x78-0x7f */
00066     0xf1, 0xf3, 0xf5, 0xf7, 0xfb, 0xfd, 0xa2, 0x00, /* 0x80-0x87 */
00067     0x00, 0xa3, 0xad, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00068 };
00069 static const unsigned char armSCII_8_page20[24] = {
00070     0x00, 0x00, 0x00, 0x00, 0xa8, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00071     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00072     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xae, 0x00, /* 0x20-0x27 */
00073 };
00074

```

```

00075 static int
00076 armSCII_8_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00077 {
00078     unsigned char c = 0;
00079     if (wc < 0x0028) {
00080         *r = wc;
00081         return 1;
00082     }
00083     else if (wc >= 0x0028 && wc < 0x0030)
00084         c = armSCII_8_page00[wc-0x0028];
00085     else if (wc >= 0x0030 && wc < 0x00a0)
00086         c = wc;
00087     else if (wc >= 0x00a0 && wc < 0x00c0)
00088         c = armSCII_8_page00_1[wc-0x00a0];
00089     else if (wc >= 0x0530 && wc < 0x0590)
00090         c = armSCII_8_page05[wc-0x0530];
00091     else if (wc >= 0x2010 && wc < 0x2028)
00092         c = armSCII_8_page20[wc-0x2010];
00093     if (c != 0) {
00094         *r = c;
00095         return 1;
00096     }
00097     return RET_ILSEQ;
00098 }

```

12.258 ascii.h

```

00001 /* $XFree86: xc/lib/X11/libcUniConv/ascii.h,v 1.3 2000/11/29 17:40:28 dawes Exp $ */
00002
00003 /*
00004  * ASCII
00005  */
00006
00007 static int
00008 ascii_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00009 {
00010     unsigned char c = *s;
00011     if (c < 0x80) {
00012         *pwc = (ucs4_t) c;
00013         return 1;
00014     }
00015     return RET_ILSEQ;
00016 }
00017
00018 static int
00019 ascii_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00020 {
00021     if (wc < 0x0080) {
00022         *r = wc;
00023         return 1;
00024     }
00025     return RET_ILSEQ;
00026 }

```

12.259 big5.h

```

00001 /* $XFree86: xc/lib/X11/libcUniConv/big5.h,v 1.2 2003/05/27 22:26:28 tsi Exp $ */
00002
00003 /*
00004  * BIG5
00005  */
00006 #ifdef NEED_TOWC
00007 static const unsigned short big5_2uni_pageal[6121] = {
00008     /* 0xal */
00009     0x3000, 0xff0c, 0x3001, 0x3002, 0xff0e, 0x2022, 0xff1b, 0xff1a,
00010     0xff1f, 0xff01, 0xfe30, 0x2026, 0x2025, 0xfe50, 0xff64, 0xfe52,
00011     0x00b7, 0xfe54, 0xfe55, 0xfe56, 0xfe57, 0xff5c, 0x2013, 0xfe31,
00012     0x2014, 0xfe33, 0xfffd, 0xfe34, 0xfe4f, 0xff08, 0xff09, 0xfe35,
00013     0xfe36, 0xff5b, 0xff5d, 0xfe37, 0xfe38, 0x3014, 0x3015, 0xfe39,
00014     0xfe3a, 0x3010, 0x3011, 0xfe3b, 0xfe3c, 0x300a, 0x300b, 0xfe3d,
00015     0xfe3e, 0x3008, 0x3009, 0xfe3f, 0xfe40, 0x300c, 0x300d, 0xfe41,
00016     0xfe42, 0x300e, 0x300f, 0xfe43, 0xfe44, 0xfe59, 0xfe5a, 0xfe5b,
00017     0xfe5c, 0xfe5d, 0xfe5e, 0x2018, 0x2019, 0x201c, 0x201d, 0x301d,
00018     0x301e, 0x2035, 0x2032, 0xff03, 0xff06, 0xff0a, 0x203b, 0x00a7,
00019     0x3003, 0x25cb, 0x25cf, 0x25b3, 0x25b2, 0x25ce, 0x2606, 0x2605,
00020     0x25c7, 0x25c6, 0x25a1, 0x25a0, 0x25bd, 0x25bc, 0x32a3, 0x2105,
00021     0x203e, 0xfffd, 0xff3f, 0xfffd, 0xfe49, 0xfe4a, 0xfe4d, 0xfe4e,
00022     0xfe4b, 0xfe4c, 0xfe5f, 0xfe60, 0xfe61, 0xff0b, 0xff0d, 0x00d7,
00023     0x00f7, 0x00b1, 0x221a, 0xff1c, 0xff1e, 0xff1d, 0x2266, 0x2267,
00024     0x2260, 0x221e, 0x2252, 0x2261, 0xfe62, 0xfe63, 0xfe64, 0xfe65,
00025     0xfe66, 0x223c, 0x2229, 0x222a, 0x22a5, 0x2220, 0x221f, 0x22bf,
00026     0x33d2, 0x33d1, 0x222b, 0x222e, 0x2235, 0x2234, 0x2640, 0x2642,

```



```
00027 0x2641, 0x2609, 0x2191, 0x2193, 0x2190, 0x2192, 0x2196, 0x2197,
00028 0x2199, 0x2198, 0x2225, 0x2223, 0xffffd,
00029 /* 0xa2 */
00030 0xffffd, 0xff0f, 0xff3c, 0xff04, 0x00a5, 0x3012, 0x00a2, 0x00a3,
00031 0xff05, 0xff20, 0x2103, 0x2109, 0xfe69, 0xfe6a, 0xfe6b, 0x33d5,
00032 0x339c, 0x339d, 0x339e, 0x33ce, 0x33a1, 0x338e, 0x338f, 0x33c4,
00033 0x00b0, 0x5159, 0x515b, 0x515e, 0x515d, 0x5161, 0x5163, 0x55e7,
00034 0x74e9, 0x7cce, 0x2581, 0x2582, 0x2583, 0x2584, 0x2585, 0x2586,
00035 0x2587, 0x2588, 0x258f, 0x258e, 0x258d, 0x258c, 0x258b, 0x258a,
00036 0x2589, 0x253c, 0x2534, 0x252c, 0x2524, 0x251c, 0x2594, 0x2500,
00037 0x2502, 0x2595, 0x250c, 0x2510, 0x2514, 0x2518, 0x256d, 0x256e,
00038 0x2570, 0x256f, 0x2550, 0x255e, 0x256a, 0x2561, 0x25e2, 0x25e3,
00039 0x25e5, 0x25e4, 0x2571, 0x2572, 0x2573, 0xff10, 0xff11, 0xff12,
00040 0xff13, 0xff14, 0xff15, 0xff16, 0xff17, 0xff18, 0xff19, 0x2160,
00041 0x2161, 0x2162, 0x2163, 0x2164, 0x2165, 0x2166, 0x2167, 0x2168,
00042 0x2169, 0x3021, 0x3022, 0x3023, 0x3024, 0x3025, 0x3026, 0x3027,
00043 0x3028, 0x3029, 0xffffd, 0x5344, 0xffffd, 0xff21, 0xff22, 0xff23,
00044 0xff24, 0xff25, 0xff26, 0xff27, 0xff28, 0xff29, 0xff2a, 0xff2b,
00045 0xff2c, 0xff2d, 0xff2e, 0xff2f, 0xff30, 0xff31, 0xff32, 0xff33,
00046 0xff34, 0xff35, 0xff36, 0xff37, 0xff38, 0xff39, 0xff3a, 0xff41,
00047 0xff42, 0xff43, 0xff44, 0xff45, 0xff46, 0xff47, 0xff48, 0xff49,
00048 0xff4a, 0xff4b, 0xff4c, 0xff4d, 0xff4e, 0xff4f, 0xff50, 0xff51,
00049 0xff52, 0xff53, 0xff54, 0xff55, 0xff56,
00050 /* 0xa3 */
00051 0xff57, 0xff58, 0xff59, 0xff5a, 0x0391, 0x0392, 0x0393, 0x0394,
00052 0x0395, 0x0396, 0x0397, 0x0398, 0x0399, 0x039a, 0x039b, 0x039c,
00053 0x039d, 0x039e, 0x039f, 0x03a0, 0x03a1, 0x03a3, 0x03a4, 0x03a5,
00054 0x03a6, 0x03a7, 0x03a8, 0x03a9, 0x03b1, 0x03b2, 0x03b3, 0x03b4,
00055 0x03b5, 0x03b6, 0x03b7, 0x03b8, 0x03b9, 0x03ba, 0x03bb, 0x03bc,
00056 0x03bd, 0x03be, 0x03bf, 0x03c0, 0x03c1, 0x03c3, 0x03c4, 0x03c5,
00057 0x03c6, 0x03c7, 0x03c8, 0x03c9, 0x3105, 0x3106, 0x3107, 0x3108,
00058 0x3109, 0x310a, 0x310b, 0x310c, 0x310d, 0x310e, 0x310f, 0x3110,
00059 0x3111, 0x3112, 0x3113, 0x3114, 0x3115, 0x3116, 0x3117, 0x3118,
00060 0x3119, 0x311a, 0x311b, 0x311c, 0x311d, 0x311e, 0x311f, 0x3120,
00061 0x3121, 0x3122, 0x3123, 0x3124, 0x3125, 0x3126, 0x3127, 0x3128,
00062 0x3129, 0x02d9, 0x02c9, 0x02ca, 0x02c7, 0x02cb, 0xffffd, 0xffffd,
00063 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00064 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00065 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00066 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00067 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00068 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00069 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00070 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00071 /* 0xa4 */
00072 0x4e00, 0x4e59, 0x4e01, 0x4e03, 0x4e43, 0x4e5d, 0x4e86, 0x4e8c,
00073 0x4eba, 0x513f, 0x5165, 0x516b, 0x51e0, 0x5200, 0x5201, 0x529b,
00074 0x5315, 0x5341, 0x535c, 0x53c8, 0x4e09, 0x4e0b, 0x4e08, 0x4e0a,
00075 0x4e2b, 0x4e38, 0x51e1, 0x4e45, 0x4e48, 0x4e5f, 0x4e5e, 0x4e8e,
00076 0x4ea1, 0x5140, 0x5203, 0x52fa, 0x5343, 0x53c9, 0x53e3, 0x571f,
00077 0x58eb, 0x5915, 0x5927, 0x5973, 0x5b50, 0x5b51, 0x5b53, 0x5b5f,
00078 0x5c0f, 0x5c22, 0x5c38, 0x5c71, 0x5ddd, 0x5de5, 0x5df1, 0x5df2,
00079 0x5df3, 0x5dfe, 0x5e72, 0x5efe, 0x5f0b, 0x5f13, 0x624d, 0x4e11,
00080 0x4e10, 0x4e0d, 0x4e2d, 0x4e30, 0x4e39, 0x4e4b, 0x5c39, 0x4e88,
00081 0x4e91, 0x4e95, 0x4e92, 0x4e94, 0x4ea2, 0x4ec1, 0x4ec0, 0x4ec3,
00082 0x4ec6, 0x4ec7, 0x4ecd, 0x4eca, 0x4ecb, 0x4ec4, 0x5143, 0x5141,
00083 0x5167, 0x516d, 0x516e, 0x516c, 0x5197, 0x51f6, 0x5206, 0x5207,
00084 0x5208, 0x52fb, 0x52fe, 0x52ff, 0x5316, 0x5339, 0x5348, 0x5347,
00085 0x5345, 0x535e, 0x5384, 0x53cb, 0x53ca, 0x53cd, 0x58ec, 0x5929,
00086 0x592b, 0x592a, 0x592d, 0x5b54, 0x5c11, 0x5c24, 0x5c3a, 0x5c6f,
00087 0x5df4, 0x5e7b, 0x5eff, 0x5f14, 0x5f15, 0x5fc3, 0x6208, 0x6236,
00088 0x624b, 0x624e, 0x652f, 0x6587, 0x6597, 0x65a4, 0x65b9, 0x65e5,
00089 0x66f0, 0x6708, 0x6728, 0x6b20, 0x6b62, 0x6b79, 0x6bcb, 0x6bd4,
00090 0x6bdb, 0x6c0f, 0x6c34, 0x706b, 0x722a, 0x7236, 0x723b, 0x7247,
00091 0x7259, 0x725b, 0x72ac, 0x738b, 0x4e19,
00092 /* 0xa5 */
00093 0x4e16, 0x4e18, 0x4e14, 0x4e18, 0x4e3b, 0x4e4d, 0x4e4f, 0x4e4e,
00094 0x4ee5, 0x4ed5, 0x4ed4, 0x4ed5, 0x4ed6, 0x4ed7, 0x4ee3, 0x4ee4,
00095 0x4ed9, 0x4ede, 0x5145, 0x5144, 0x5189, 0x518a, 0x51ac, 0x51f9,
00096 0x51fa, 0x51f8, 0x520a, 0x52a0, 0x529f, 0x5305, 0x5306, 0x5317,
00097 0x531d, 0x4edf, 0x534a, 0x5349, 0x5361, 0x5360, 0x536f, 0x536e,
00098 0x53bb, 0x53ef, 0x53e4, 0x53f3, 0x53ec, 0x53ee, 0x53e9, 0x53e8,
00099 0x53fc, 0x53f8, 0x53f5, 0x53eb, 0x53e6, 0x53ea, 0x53f2, 0x53f1,
00100 0x53f0, 0x53e5, 0x53ed, 0x53fb, 0x56db, 0x56da, 0x5916, 0x592e,
00101 0x5931, 0x5974, 0x5976, 0x5b55, 0x5b83, 0x5c3c, 0x5de8, 0x5de7,
00102 0x5de6, 0x5e02, 0x5e03, 0x5e73, 0x5e7c, 0x5f01, 0x5f18, 0x5f17,
00103 0x5fc5, 0x620a, 0x6253, 0x6254, 0x6252, 0x6251, 0x65a5, 0x65e6,
00104 0x672e, 0x672c, 0x672a, 0x672b, 0x672d, 0x6b63, 0x6bcd, 0x6c11,
00105 0x6c10, 0x6c38, 0x6c41, 0x6c40, 0x6c3e, 0x72af, 0x738a, 0x7389,
00106 0x74dc, 0x74e6, 0x7518, 0x751f, 0x7528, 0x7529, 0x7530, 0x7531,
00107 0x7532, 0x7533, 0x758b, 0x767d, 0x76ae, 0x76bf, 0x76ee, 0x77db,
00108 0x77e2, 0x77f3, 0x793a, 0x79be, 0x7a74, 0x7acb, 0x4e1e, 0x4e1f,
00109 0x4e52, 0x4e53, 0x4e69, 0x4e99, 0x4ea4, 0x4ea6, 0x4ea5, 0x4eff,
00110 0x4f09, 0x4f19, 0x4f0a, 0x4f15, 0x4f0d, 0x4f10, 0x4f11, 0x4f0f,
00111 0x4ef2, 0x4ef6, 0x4efb, 0x4ef0, 0x4ef3, 0x4efd, 0x4f01, 0x4f0b,
00112 0x5149, 0x5147, 0x5146, 0x5148, 0x5168,
00113 /* 0xa6 */
```

```

00114 0x5171, 0x518d, 0x51b0, 0x5217, 0x5211, 0x5212, 0x520e, 0x5216,
00115 0x52a3, 0x5308, 0x5321, 0x5320, 0x5370, 0x5371, 0x5409, 0x540f,
00116 0x540c, 0x540a, 0x5410, 0x5401, 0x540b, 0x5404, 0x5411, 0x540d,
00117 0x5408, 0x5403, 0x540e, 0x5406, 0x5412, 0x56e0, 0x56de, 0x56dd,
00118 0x5733, 0x5730, 0x5728, 0x572d, 0x572c, 0x572f, 0x5729, 0x5919,
00119 0x591a, 0x5937, 0x5938, 0x5984, 0x5978, 0x5983, 0x597d, 0x5979,
00120 0x5982, 0x5981, 0x5b57, 0x5b58, 0x5b87, 0x5b88, 0x5b85, 0x5b89,
00121 0x5bfa, 0x5c16, 0x5c79, 0x5dde, 0x5e06, 0x5e76, 0x5e74, 0x5f0f,
00122 0x5f1b, 0x5fd9, 0x5fd6, 0x620e, 0x620c, 0x620d, 0x6210, 0x6263,
00123 0x625b, 0x6258, 0x6536, 0x65e9, 0x65e8, 0x65ec, 0x65ed, 0x66f2,
00124 0x66f3, 0x6709, 0x673d, 0x6734, 0x6731, 0x6735, 0x6b21, 0x6b64,
00125 0x6b7b, 0x6c16, 0x6c5d, 0x6c57, 0x6c59, 0x6c5f, 0x6c60, 0x6c50,
00126 0x6c55, 0x6c61, 0x6c5b, 0x6c4d, 0x6c4e, 0x7070, 0x725f, 0x725d,
00127 0x767e, 0x7af9, 0x7c73, 0x7cf8, 0x7f36, 0x7f8a, 0x7fbd, 0x8001,
00128 0x8003, 0x800c, 0x8012, 0x8033, 0x807f, 0x8089, 0x808b, 0x808c,
00129 0x81e3, 0x81ea, 0x81f3, 0x81fc, 0x820c, 0x821b, 0x821f, 0x826e,
00130 0x8272, 0x827e, 0x866b, 0x8840, 0x884c, 0x8863, 0x897f, 0x9621,
00131 0x4e32, 0x4ea8, 0x4f4d, 0x4f4f, 0x4f47, 0x4f57, 0x4f5e, 0x4f34,
00132 0x4f5b, 0x4f55, 0x4f30, 0x4f50, 0x4f51, 0x4f3d, 0x4f3a, 0x4f38,
00133 0x4f43, 0x4f54, 0x4f3c, 0x4f46, 0x4f63,
00134 /* 0xa7 */
00135 0x4f5c, 0x4f60, 0x4f2f, 0x4f4e, 0x4f36, 0x4f59, 0x4f5d, 0x4f48,
00136 0x4f5a, 0x514c, 0x514b, 0x514d, 0x5175, 0x51b6, 0x51b7, 0x5225,
00137 0x522a, 0x5229, 0x522a, 0x5228, 0x52ab, 0x52a9, 0x52aa, 0x52ac,
00138 0x5323, 0x5373, 0x5375, 0x541d, 0x542d, 0x541e, 0x543e, 0x5426,
00139 0x544e, 0x5427, 0x5446, 0x5443, 0x5433, 0x5448, 0x5442, 0x541b,
00140 0x5429, 0x544a, 0x5439, 0x543b, 0x5438, 0x542e, 0x5435, 0x5436,
00141 0x5420, 0x543c, 0x5440, 0x5431, 0x542b, 0x541f, 0x542c, 0x56ea,
00142 0x56f0, 0x56e4, 0x56eb, 0x574a, 0x5751, 0x5740, 0x574d, 0x5747,
00143 0x574e, 0x573e, 0x5750, 0x574f, 0x573b, 0x58ef, 0x593e, 0x599d,
00144 0x5992, 0x59a8, 0x599e, 0x59a3, 0x5999, 0x5996, 0x598d, 0x59a4,
00145 0x5993, 0x598a, 0x59a5, 0x5b5d, 0x5b5c, 0x5b5a, 0x5b5b, 0x5b8c,
00146 0x5b8b, 0x5b8f, 0x5c2c, 0x5c40, 0x5c41, 0x5c3f, 0x5c3e, 0x5c90,
00147 0x5c91, 0x5c94, 0x5c8c, 0x5deb, 0x5e0c, 0x5e8f, 0x5e87, 0x5e8a,
00148 0x5ef7, 0x5f04, 0x5f1f, 0x5f64, 0x5f62, 0x5f77, 0x5f79, 0x5fd8,
00149 0x5fcc, 0x5fd7, 0x5fcd, 0x5ff1, 0x5feb, 0x5ff8, 0x5fea, 0x6212,
00150 0x6211, 0x6284, 0x6297, 0x6296, 0x6280, 0x6276, 0x6289, 0x626d,
00151 0x628a, 0x627c, 0x627e, 0x6279, 0x6273, 0x6292, 0x626f, 0x6298,
00152 0x626e, 0x6295, 0x6293, 0x6291, 0x6286, 0x6539, 0x653b, 0x6538,
00153 0x65f1, 0x66f4, 0x675f, 0x674e, 0x674f, 0x6750, 0x6751, 0x675c,
00154 0x6756, 0x675e, 0x6749, 0x6746, 0x6760,
00155 /* 0xa8 */
00156 0x6753, 0x6757, 0x6b65, 0x6bcf, 0x6c42, 0x6c5e, 0x6c99, 0x6c81,
00157 0x6c88, 0x6c89, 0x6c85, 0x6c9b, 0x6c6a, 0x6c7a, 0x6c90, 0x6c70,
00158 0x6c8c, 0x6c68, 0x6c96, 0x6c92, 0x6c7d, 0x6c83, 0x6c72, 0x6c7e,
00159 0x6c74, 0x6c86, 0x6c76, 0x6c8d, 0x6c94, 0x6c98, 0x6c82, 0x7076,
00160 0x707c, 0x707d, 0x7078, 0x7262, 0x7261, 0x7260, 0x72c4, 0x72c2,
00161 0x7396, 0x752c, 0x752b, 0x7537, 0x7538, 0x7682, 0x76ef, 0x773c,
00162 0x79c1, 0x79c0, 0x79bf, 0x7a76, 0x7cfb, 0x7f55, 0x8096, 0x8093,
00163 0x809d, 0x8098, 0x809b, 0x809a, 0x80b2, 0x826f, 0x8292, 0x828b,
00164 0x828d, 0x898b, 0x89d2, 0x8a00, 0x8c37, 0x8c46, 0x8c55, 0x8c9d,
00165 0x8d64, 0x8d70, 0x8db3, 0x8eab, 0x8eca, 0x8f9b, 0x8fb0, 0x8fc2,
00166 0x8fc6, 0x8fc5, 0x8fc4, 0x5de1, 0x9091, 0x90a2, 0x90aa, 0x90a6,
00167 0x90a3, 0x9149, 0x91c6, 0x91cc, 0x9632, 0x962e, 0x9631, 0x962a,
00168 0x962c, 0x4e26, 0x4e56, 0x4e73, 0x4e8b, 0x4e9b, 0x4e9e, 0x4eab,
00169 0x4eac, 0x4f6f, 0x4f9d, 0x4f8d, 0x4f73, 0x4f7f, 0x4f6c, 0x4f9b,
00170 0x4f8b, 0x4f86, 0x4f83, 0x4f70, 0x4f75, 0x4f88, 0x4f69, 0x4f7b,
00171 0x4f96, 0x4f7e, 0x4f8f, 0x4f91, 0x4f7a, 0x5154, 0x5152, 0x5155,
00172 0x5169, 0x5177, 0x5176, 0x5178, 0x51bd, 0x51fd, 0x523b, 0x5238,
00173 0x5237, 0x523a, 0x5230, 0x522e, 0x5236, 0x5241, 0x52be, 0x52bb,
00174 0x5352, 0x5354, 0x5353, 0x5351, 0x5366, 0x5377, 0x5378, 0x5379,
00175 0x53d6, 0x53d4, 0x53d7, 0x5473, 0x5475,
00176 /* 0xa9 */
00177 0x5496, 0x5478, 0x5495, 0x5480, 0x547b, 0x5477, 0x5484, 0x5492,
00178 0x5486, 0x547c, 0x5490, 0x5471, 0x5476, 0x548c, 0x549a, 0x5462,
00179 0x5468, 0x548b, 0x547d, 0x548e, 0x56fa, 0x5783, 0x5777, 0x576a,
00180 0x5769, 0x5761, 0x5766, 0x5764, 0x577c, 0x591c, 0x5949, 0x5947,
00181 0x5948, 0x5944, 0x5954, 0x59be, 0x59bb, 0x59d4, 0x59b9, 0x59ae,
00182 0x59d1, 0x59c6, 0x59d0, 0x59cd, 0x59cb, 0x59d3, 0x59ca, 0x59af,
00183 0x59b3, 0x59d2, 0x59c5, 0x5b5f, 0x5b64, 0x5b63, 0x5b97, 0x5b9a,
00184 0x5b98, 0x5b9c, 0x5b99, 0x5b9b, 0x5c1a, 0x5c48, 0x5c45, 0x5c46,
00185 0x5cb7, 0x5ca1, 0x5cb8, 0x5ca9, 0x5cab, 0x5cb1, 0x5cb3, 0x5e18,
00186 0x5e1a, 0x5e16, 0x5e15, 0x5e1b, 0x5e11, 0x5e78, 0x5e9a, 0x5e97,
00187 0x5e9c, 0x5e95, 0x5e96, 0x5ef6, 0x5f26, 0x5f27, 0x5f29, 0x5f80,
00188 0x5f81, 0x5f7f, 0x5f7c, 0x5fdd, 0x5fe0, 0x5ffd, 0x5fff, 0x5fff,
00189 0x600f, 0x6014, 0x602f, 0x6035, 0x6016, 0x602a, 0x6015, 0x6021,
00190 0x6027, 0x6029, 0x602b, 0x601b, 0x6216, 0x6215, 0x623f, 0x623e,
00191 0x6240, 0x627f, 0x62c9, 0x62cc, 0x62c4, 0x62bf, 0x62c2, 0x62b9,
00192 0x62d2, 0x62db, 0x62ab, 0x62d3, 0x62d4, 0x62cb, 0x62c8, 0x62a8,
00193 0x62bd, 0x62bc, 0x62d0, 0x62d9, 0x62c7, 0x62cd, 0x62b5, 0x62da,
00194 0x62b1, 0x62d8, 0x62d6, 0x62d7, 0x62c6, 0x62ac, 0x62ce, 0x653e,
00195 0x65a7, 0x65bc, 0x65fa, 0x6614, 0x6613, 0x660c, 0x6606, 0x6602,
00196 0x660e, 0x6600, 0x660f, 0x6615, 0x660a,
00197 /* 0xaa */
00198 0x6607, 0x670d, 0x670b, 0x676d, 0x678b, 0x6795, 0x6771, 0x679c,
00199 0x6773, 0x6777, 0x6787, 0x679d, 0x6797, 0x676f, 0x6770, 0x677f,
00200 0x6789, 0x677e, 0x6790, 0x6775, 0x679a, 0x6793, 0x677c, 0x676a,

```

```
00201 0x6772, 0x6b23, 0x6b66, 0x6b67, 0x6b7f, 0x6c13, 0x6c1b, 0x6ce3,
00202 0x6ce8, 0x6cf3, 0x6cb1, 0x6ccc, 0x6ce5, 0x6cb3, 0x6cbd, 0x6cbe,
00203 0x6cbc, 0x6ce2, 0x6cab, 0x6cd5, 0x6cd3, 0x6cb8, 0x6cc4, 0x6cb9,
00204 0x6cc1, 0x6cae, 0x6cd7, 0x6cc5, 0x6cf1, 0x6cbf, 0x6cbb, 0x6ce1,
00205 0x6cdb, 0x6cca, 0x6cac, 0x6cef, 0x6cdc, 0x6cd6, 0x6ce0, 0x7095,
00206 0x708e, 0x7092, 0x708a, 0x7099, 0x722c, 0x722d, 0x7238, 0x7248,
00207 0x7267, 0x7269, 0x72c0, 0x72ce, 0x72d9, 0x72d7, 0x72d0, 0x73a9,
00208 0x73a8, 0x739f, 0x73ab, 0x73a5, 0x753d, 0x759d, 0x7599, 0x759a,
00209 0x7684, 0x76c2, 0x76f2, 0x76f4, 0x77e5, 0x77fd, 0x793e, 0x7940,
00210 0x7941, 0x79c9, 0x79c8, 0x7a7a, 0x7a79, 0x7afa, 0x7cfe, 0x7f54,
00211 0x7f8c, 0x7f8b, 0x8005, 0x80ba, 0x80a5, 0x80a2, 0x80b1, 0x80a1,
00212 0x80ab, 0x80a9, 0x80b4, 0x80aa, 0x80af, 0x81e5, 0x81fe, 0x820d,
00213 0x82b3, 0x829d, 0x8299, 0x82ad, 0x82bd, 0x829f, 0x82b9, 0x82b1,
00214 0x82ac, 0x82a5, 0x82af, 0x82b8, 0x82a3, 0x82b0, 0x82be, 0x82b7,
00215 0x864e, 0x8671, 0x521d, 0x8868, 0x8ecb, 0x8fce, 0x8fd4, 0x8fd1,
00216 0x90b5, 0x90b8, 0x90b1, 0x90b6, 0x91c7, 0x91d1, 0x9577, 0x9580,
00217 0x961c, 0x9640, 0x963f, 0x963b, 0x9644,
00218 /* 0xab */
00219 0x9642, 0x96b9, 0x96e8, 0x9752, 0x975e, 0x4e9f, 0x4ead, 0x4eae,
00220 0x4fe1, 0x4fb5, 0x4faf, 0x4fbf, 0x4fe0, 0x4fd1, 0x4fcf, 0x4fdd,
00221 0x4fc3, 0x4fb6, 0x4fd8, 0x4fdf, 0x4fca, 0x4fd7, 0x4fae, 0x4fd0,
00222 0x4fc4, 0x4fc2, 0x4fda, 0x4fce, 0x4fde, 0x4fb7, 0x5157, 0x5192,
00223 0x5191, 0x51a0, 0x524e, 0x5243, 0x524a, 0x524d, 0x524c, 0x524b,
00224 0x5247, 0x52c7, 0x52c9, 0x52c3, 0x52c1, 0x530d, 0x5357, 0x537b,
00225 0x539a, 0x53db, 0x54ac, 0x54c0, 0x54a8, 0x54ce, 0x54c9, 0x54b8,
00226 0x54a6, 0x54b3, 0x54c7, 0x54c2, 0x54bd, 0x54aa, 0x54c1, 0x54c4,
00227 0x54c8, 0x54af, 0x54ab, 0x54bf, 0x54bb, 0x54a9, 0x54a7, 0x54bf,
00228 0x56ff, 0x5782, 0x578b, 0x57a0, 0x57a3, 0x57a2, 0x57ce, 0x57ae,
00229 0x5793, 0x5955, 0x5951, 0x594f, 0x594e, 0x5950, 0x59dc, 0x59d8,
00230 0x59ff, 0x59e3, 0x59e8, 0x5a03, 0x59e5, 0x59ea, 0x59da, 0x59e6,
00231 0x5a01, 0x59fb, 0x5b69, 0x5ba3, 0x5ba6, 0x5ba4, 0x5ba2, 0x5ba5,
00232 0x5c01, 0x5c4e, 0x5c4f, 0x5c4d, 0x5c4b, 0x5cd9, 0x5cd2, 0x5df7,
00233 0x5e1d, 0x5e25, 0x5e1f, 0x5e7d, 0x5ea0, 0x5ea6, 0x5efa, 0x5f08,
00234 0x5f2d, 0x5f65, 0x5f88, 0x5f85, 0x5f8a, 0x5f8b, 0x5f87, 0x5f8c,
00235 0x5f89, 0x6012, 0x601d, 0x6020, 0x6025, 0x600e, 0x6028, 0x604d,
00236 0x6070, 0x6068, 0x6062, 0x6046, 0x6043, 0x606c, 0x606b, 0x606a,
00237 0x6064, 0x6241, 0x62dc, 0x6316, 0x6309, 0x62fc, 0x62ed, 0x6301,
00238 0x62ee, 0x62fd, 0x6307, 0x62f1, 0x62f7,
00239 /* 0xac */
00240 0x62ef, 0x62ec, 0x62fe, 0x62f4, 0x6311, 0x6302, 0x653f, 0x6545,
00241 0x65ab, 0x65bd, 0x65e2, 0x6625, 0x662d, 0x6620, 0x6627, 0x662f,
00242 0x661f, 0x6628, 0x6631, 0x6624, 0x66f7, 0x67ff, 0x67d3, 0x67f1,
00243 0x67d4, 0x67d0, 0x67ec, 0x67b6, 0x67af, 0x67f5, 0x67e9, 0x67ef,
00244 0x67c4, 0x67d1, 0x67b4, 0x67da, 0x67e5, 0x67b8, 0x67cf, 0x67de,
00245 0x67f3, 0x67b0, 0x67d9, 0x67e2, 0x67dd, 0x67d2, 0x6b6a, 0x6b83,
00246 0x6b86, 0x6bb5, 0x6bd2, 0x6bd7, 0x6c1f, 0x6cc9, 0x6d0b, 0x6d32,
00247 0x6d2a, 0x6d41, 0x6d25, 0x6d0c, 0x6d31, 0x6d1e, 0x6d17, 0x6d3b,
00248 0x6d3d, 0x6d3e, 0x6d36, 0x6d1b, 0x6cf5, 0x6d39, 0x6d27, 0x6d38,
00249 0x6d29, 0x6d2e, 0x6d35, 0x6d0e, 0x6d2b, 0x70ab, 0x70ba, 0x70b3,
00250 0x70ac, 0x70af, 0x70ad, 0x70b8, 0x70ae, 0x70a4, 0x7230, 0x7272,
00251 0x726f, 0x7274, 0x72e9, 0x72e0, 0x72e1, 0x73b7, 0x73ca, 0x73bb,
00252 0x73b2, 0x73cd, 0x73c0, 0x73b3, 0x751a, 0x752d, 0x754f, 0x754c,
00253 0x754e, 0x754b, 0x75ab, 0x75a4, 0x75a5, 0x75a2, 0x75a3, 0x7678,
00254 0x7686, 0x7687, 0x7688, 0x76c8, 0x76c6, 0x76c3, 0x76c5, 0x7701,
00255 0x76f9, 0x76f8, 0x7709, 0x770b, 0x76fe, 0x76fc, 0x7707, 0x77dc,
00256 0x7802, 0x7814, 0x780c, 0x780d, 0x7946, 0x7949, 0x7948, 0x7947,
00257 0x79b9, 0x79ba, 0x79d1, 0x79d2, 0x79cb, 0x7a7f, 0x7a81, 0x7aff,
00258 0x7afd, 0x7c7d, 0x7d02, 0x7d05, 0x7d00, 0x7d09, 0x7d07, 0x7d04,
00259 0x7d06, 0x7f38, 0x7f8e, 0x7fbf, 0x8004,
00260 /* 0xad */
00261 0x8010, 0x800d, 0x8011, 0x8036, 0x80d6, 0x80e5, 0x80da, 0x80c3,
00262 0x80c4, 0x80cc, 0x80e1, 0x80db, 0x80ce, 0x80de, 0x80e4, 0x80dd,
00263 0x81f4, 0x8222, 0x82e7, 0x8303, 0x8305, 0x82e3, 0x82db, 0x82e6,
00264 0x8304, 0x82e5, 0x8302, 0x8309, 0x82d2, 0x82d7, 0x82f1, 0x8301,
00265 0x82dc, 0x82d4, 0x82d1, 0x82de, 0x82d3, 0x82df, 0x82ef, 0x8306,
00266 0x8650, 0x8679, 0x867b, 0x867a, 0x884d, 0x886b, 0x8981, 0x89d4,
00267 0x8a08, 0x8a02, 0x8a03, 0x8c9e, 0x8ca0, 0x8d74, 0x8d73, 0x8db4,
00268 0x8ecd, 0x8ecc, 0x8ff0, 0x8fe6, 0x8fe2, 0x8fea, 0x8fe5, 0x8fed,
00269 0x8feb, 0x8fe4, 0x8fe8, 0x90ca, 0x90ce, 0x90c1, 0x90c3, 0x914b,
00270 0x914a, 0x91cd, 0x9582, 0x9650, 0x964b, 0x964c, 0x964d, 0x9762,
00271 0x9769, 0x97cb, 0x97ed, 0x97f3, 0x9801, 0x98a8, 0x98db, 0x98df,
00272 0x9996, 0x9999, 0x4e58, 0x4eb3, 0x500c, 0x500d, 0x5023, 0x4fef,
00273 0x5026, 0x5025, 0x4ff8, 0x5029, 0x5016, 0x5006, 0x503c, 0x501f,
00274 0x501a, 0x5012, 0x5011, 0x4ffa, 0x5000, 0x5014, 0x5028, 0x4ff1,
00275 0x5021, 0x500b, 0x5019, 0x5018, 0x4ff3, 0x4fee, 0x502d, 0x502a,
00276 0x4ffe, 0x502b, 0x5009, 0x517c, 0x51a4, 0x51a5, 0x51a2, 0x51cd,
00277 0x51cc, 0x51c6, 0x51cb, 0x5256, 0x525c, 0x5254, 0x525b, 0x525d,
00278 0x532a, 0x537f, 0x539f, 0x539d, 0x53df, 0x54e8, 0x5510, 0x5501,
00279 0x5537, 0x54fc, 0x54e5, 0x54f2, 0x5506, 0x54fa, 0x5514, 0x54e9,
00280 0x54ed, 0x54e1, 0x5509, 0x54ee, 0x54ea,
00281 /* 0xae */
00282 0x54e6, 0x5527, 0x5507, 0x54fd, 0x550f, 0x5703, 0x5704, 0x57c2,
00283 0x57d4, 0x57cb, 0x57c3, 0x5809, 0x590f, 0x5957, 0x5958, 0x595a,
00284 0x5a11, 0x5a18, 0x5a1c, 0x5a1f, 0x5a1b, 0x5a13, 0x59ec, 0x5a20,
00285 0x5a23, 0x5a29, 0x5a25, 0x5a0c, 0x5a09, 0x5b6b, 0x5c58, 0x5bb0,
00286 0x5bb3, 0x5bb6, 0x5bb4, 0x5bae, 0x5bb5, 0x5bb9, 0x5bb8, 0x5c04,
00287 0x5c51, 0x5c55, 0x5c50, 0x5ced, 0x5cfd, 0x5cfb, 0x5cea, 0x5ce8,
```

```

00288 0x5cf0, 0x5cf6, 0x5d01, 0x5cf4, 0x5dee, 0x5e2d, 0x5e2b, 0x5eab,
00289 0x5ead, 0x5ea7, 0x5f31, 0x5f92, 0x5f91, 0x5f90, 0x6059, 0x6063,
00290 0x6065, 0x6050, 0x6055, 0x606d, 0x6069, 0x606f, 0x6084, 0x609f,
00291 0x609a, 0x608d, 0x6094, 0x608c, 0x6085, 0x6096, 0x6247, 0x62f3,
00292 0x6308, 0x62ff, 0x634e, 0x633e, 0x632f, 0x6355, 0x6342, 0x6346,
00293 0x634f, 0x6349, 0x633a, 0x6350, 0x633d, 0x632a, 0x632b, 0x6328,
00294 0x634d, 0x634c, 0x6548, 0x6549, 0x6599, 0x65c1, 0x65c5, 0x6642,
00295 0x6649, 0x664f, 0x6643, 0x6652, 0x664c, 0x6645, 0x6641, 0x66f8,
00296 0x6714, 0x6715, 0x6717, 0x6821, 0x6838, 0x6848, 0x6846, 0x6853,
00297 0x6839, 0x6842, 0x6854, 0x6829, 0x68b3, 0x6817, 0x684c, 0x6851,
00298 0x683d, 0x67f4, 0x6850, 0x6840, 0x683c, 0x6843, 0x682a, 0x6845,
00299 0x6813, 0x6818, 0x6841, 0x6b8a, 0x6b89, 0x6bb7, 0x6c23, 0x6c27,
00300 0x6c28, 0x6c26, 0x6c24, 0x6cf0, 0x6d6a, 0x6d95, 0x6d88, 0x6d87,
00301 0x6d66, 0x6d78, 0x6d77, 0x6d59, 0x6d93,
00302 /* 0xaf */
00303 0x6d6c, 0x6d89, 0x6d6e, 0x6d5a, 0x6d74, 0x6d69, 0x6d8c, 0x6d8a,
00304 0x6d79, 0x6d85, 0x6d65, 0x6d94, 0x70ca, 0x70d8, 0x70e4, 0x70d9,
00305 0x70c8, 0x70cf, 0x7239, 0x7279, 0x72f9, 0x72fd, 0x72fe, 0x72fd,
00306 0x72f7, 0x7386, 0x73ed, 0x7409, 0x73ee, 0x73e0, 0x73ea, 0x73de,
00307 0x7554, 0x755d, 0x755c, 0x755a, 0x7559, 0x75be, 0x75c5, 0x75c7,
00308 0x75b2, 0x75b3, 0x75bd, 0x75bc, 0x75b9, 0x75c2, 0x75b8, 0x768b,
00309 0x76b0, 0x76ca, 0x76cd, 0x76ce, 0x7729, 0x771f, 0x7720, 0x7728,
00310 0x77e9, 0x7830, 0x7827, 0x7838, 0x781d, 0x7834, 0x7837, 0x7825,
00311 0x782d, 0x7820, 0x781f, 0x7832, 0x7955, 0x7950, 0x7960, 0x795f,
00312 0x7956, 0x795e, 0x795d, 0x7957, 0x795a, 0x79e4, 0x79e3, 0x79e7,
00313 0x79df, 0x79e6, 0x79e9, 0x79d8, 0x7a84, 0x7a88, 0x7ad9, 0x7b06,
00314 0x7b11, 0x7c89, 0x7d21, 0x7d17, 0x7d0b, 0x7d0a, 0x7d20, 0x7d22,
00315 0x7d14, 0x7d10, 0x7d15, 0x7d1a, 0x7d1c, 0x7d0d, 0x7d19, 0x7d1b,
00316 0x7f3a, 0x7f5f, 0x7f94, 0x7fc5, 0x7fc1, 0x8006, 0x8018, 0x8015,
00317 0x8019, 0x8017, 0x803d, 0x803f, 0x80f1, 0x8102, 0x80f0, 0x8105,
00318 0x80ed, 0x80f4, 0x8106, 0x80f8, 0x80f3, 0x8108, 0x80fd, 0x810a,
00319 0x80fc, 0x80ef, 0x81ed, 0x81ec, 0x8200, 0x8210, 0x822a, 0x822b,
00320 0x8228, 0x822c, 0x82bb, 0x832b, 0x8352, 0x8354, 0x834a, 0x8338,
00321 0x8350, 0x8349, 0x8335, 0x8334, 0x834f, 0x8332, 0x8339, 0x8336,
00322 0x8317, 0x8340, 0x8331, 0x8328, 0x8343,
00323 /* 0xb0 */
00324 0x8654, 0x868a, 0x86aa, 0x8693, 0x86a4, 0x86a9, 0x868c, 0x86a3,
00325 0x869c, 0x8870, 0x8877, 0x8881, 0x8882, 0x887d, 0x8877, 0x8a18,
00326 0x8a10, 0x8a0e, 0x8a0c, 0x8a15, 0x8a0a, 0x8a17, 0x8a13, 0x8a16,
00327 0x8a0f, 0x8a11, 0x8c48, 0x8c7a, 0x8c79, 0x8ca1, 0x8ca2, 0x8d77,
00328 0x8eac, 0x8ed2, 0x8ed4, 0x8ecf, 0x8fb1, 0x9001, 0x9006, 0x8ff7,
00329 0x9000, 0x8ffa, 0x8ff4, 0x9003, 0x8ff8, 0x9005, 0x8ff8, 0x9095,
00330 0x90e1, 0x90dd, 0x90e2, 0x9152, 0x914d, 0x914c, 0x91d8, 0x91dd,
00331 0x91d7, 0x91dc, 0x91d9, 0x9583, 0x9662, 0x9663, 0x9661, 0x965b,
00332 0x965d, 0x9664, 0x9658, 0x965e, 0x96bb, 0x96e2, 0x99ac, 0x9aa8,
00333 0x9ad8, 0x9b25, 0x9b32, 0x9b3c, 0x4e7e, 0x507a, 0x507d, 0x505c,
00334 0x5047, 0x5043, 0x504c, 0x505a, 0x5049, 0x5065, 0x5076, 0x504e,
00335 0x5055, 0x5075, 0x5074, 0x5077, 0x504f, 0x500f, 0x506f, 0x506d,
00336 0x515c, 0x5195, 0x51f0, 0x526a, 0x526f, 0x52d2, 0x52d9, 0x52d8,
00337 0x52d5, 0x5310, 0x530f, 0x5319, 0x533f, 0x5340, 0x533e, 0x53c3,
00338 0x66fc, 0x556c, 0x5566, 0x556e, 0x5544, 0x555e, 0x5561, 0x5543,
00339 0x554a, 0x5531, 0x5556, 0x554f, 0x5555, 0x5552, 0x5564, 0x5538,
00340 0x552e, 0x555c, 0x552c, 0x5563, 0x5533, 0x5541, 0x5557, 0x5708,
00341 0x570b, 0x5709, 0x57df, 0x5805, 0x580a, 0x5806, 0x57e0, 0x57e4,
00342 0x57fa, 0x5802, 0x5835, 0x57f7, 0x57f9, 0x5920, 0x5962, 0x5a36,
00343 0x5a41, 0x5a49, 0x5a66, 0x5a6a, 0x5a40,
00344 /* 0xb1 */
00345 0x5a3c, 0x5a62, 0x5a5a, 0x5a46, 0x5a4a, 0x5b70, 0x5bc7, 0x5bc5,
00346 0x5bc4, 0x5bc2, 0x5bbf, 0x5bc6, 0x5c09, 0x5c08, 0x5c07, 0x5c60,
00347 0x5c5c, 0x5c5d, 0x5d07, 0x5d06, 0x5d0e, 0x5d1b, 0x5d16, 0x5d22,
00348 0x5d11, 0x5d29, 0x5d14, 0x5d19, 0x5d24, 0x5d27, 0x5d17, 0x5de2,
00349 0x5e38, 0x5e36, 0x5e33, 0x5e37, 0x5eb7, 0x5eb8, 0x5eb6, 0x5eb5,
00350 0x5ebe, 0x5f35, 0x5f37, 0x5f57, 0x5f6c, 0x5f69, 0x5f6b, 0x5f97,
00351 0x5f99, 0x5f9e, 0x5f98, 0x5fa1, 0x5fa0, 0x5f9c, 0x607f, 0x60a3,
00352 0x6089, 0x60a0, 0x60a8, 0x60cb, 0x60b4, 0x60e6, 0x60bd, 0x60c5,
00353 0x60bb, 0x60b5, 0x60dc, 0x60bc, 0x60d8, 0x60d5, 0x60c6, 0x60df,
00354 0x60b8, 0x60da, 0x60c7, 0x621a, 0x621b, 0x6248, 0x63a0, 0x63a7,
00355 0x6372, 0x6396, 0x63a2, 0x63a5, 0x6377, 0x6367, 0x6398, 0x63aa,
00356 0x6371, 0x63a9, 0x6389, 0x6383, 0x639b, 0x636b, 0x63a8, 0x6384,
00357 0x6388, 0x6399, 0x63a1, 0x63ac, 0x6392, 0x638f, 0x6380, 0x637b,
00358 0x6369, 0x6368, 0x637a, 0x655d, 0x6556, 0x6551, 0x6559, 0x6557,
00359 0x555f, 0x654f, 0x6558, 0x6555, 0x6554, 0x659c, 0x659b, 0x65ac,
00360 0x65cf, 0x65cb, 0x65cc, 0x65ce, 0x665d, 0x665a, 0x6664, 0x6668,
00361 0x6666, 0x666e, 0x66f9, 0x52d7, 0x671b, 0x6881, 0x68af, 0x68a2,
00362 0x6893, 0x68b5, 0x687f, 0x6876, 0x68b1, 0x68a7, 0x6897, 0x68b0,
00363 0x6883, 0x68c4, 0x68ad, 0x6886, 0x6885, 0x6894, 0x689d, 0x68a8,
00364 0x689f, 0x68a1, 0x6882, 0x6b32, 0x6bba,
00365 /* 0xb2 */
00366 0x6beb, 0x6bec, 0x6c2b, 0x6d8e, 0x6dbc, 0x6df3, 0x6dd9, 0x6db2,
00367 0x6de1, 0x6dcc, 0x6de4, 0x6dfb, 0x6dfa, 0x6e05, 0x6dc7, 0x6dc6,
00368 0x6daf, 0x6dd1, 0x6dae, 0x6dde, 0x6df9, 0x6db8, 0x6df7, 0x6df5,
00369 0x6dc5, 0x6dd2, 0x6e1a, 0x6db5, 0x6dda, 0x6deb, 0x6dd8, 0x6dea,
00370 0x6df1, 0x6dee, 0x6de8, 0x6dc6, 0x6dc4, 0x6daa, 0x6dec, 0x6dbf,
00371 0x6de6, 0x70f9, 0x7109, 0x710a, 0x70fd, 0x70ef, 0x723d, 0x727d,
00372 0x7281, 0x731c, 0x731b, 0x7316, 0x7313, 0x7319, 0x7387, 0x7405,
00373 0x740a, 0x7403, 0x7406, 0x73fe, 0x740d, 0x74e0, 0x74f6, 0x74f7,
00374 0x751c, 0x7522, 0x7565, 0x7566, 0x7562, 0x7570, 0x758f, 0x75d4,

```

```
00375 0x75d5, 0x75b5, 0x75ca, 0x75cd, 0x768e, 0x76d4, 0x76d2, 0x76db,
00376 0x7737, 0x773e, 0x773c, 0x7736, 0x7738, 0x773a, 0x786b, 0x7843,
00377 0x784e, 0x7965, 0x7968, 0x796d, 0x79fb, 0x7a92, 0x7a95, 0x7b20,
00378 0x7b28, 0x7b1b, 0x7b2c, 0x7b26, 0x7b19, 0x7b1e, 0x7b2e, 0x7c92,
00379 0x7c97, 0x7c95, 0x7d46, 0x7d43, 0x7d71, 0x7d2e, 0x7d39, 0x7d3c,
00380 0x7d40, 0x7d30, 0x7d33, 0x7d44, 0x7d2f, 0x7d42, 0x7d32, 0x7d31,
00381 0x7f3d, 0x7f9e, 0x7f9a, 0x7fcc, 0x7fce, 0x7fd2, 0x801c, 0x804a,
00382 0x8046, 0x812f, 0x8116, 0x8123, 0x812b, 0x8129, 0x8130, 0x8124,
00383 0x8202, 0x8235, 0x8237, 0x8236, 0x8239, 0x838e, 0x839e, 0x8398,
00384 0x8378, 0x83a2, 0x8396, 0x83bd, 0x83ab, 0x8392, 0x838a, 0x8393,
00385 0x8389, 0x83a0, 0x8377, 0x837b, 0x837c,
00386 /* 0xb3 */
00387 0x8386, 0x83a7, 0x8655, 0x5f6a, 0x86c7, 0x86c0, 0x86b6, 0x86c4,
00388 0x86b5, 0x86c6, 0x86cb, 0x86b1, 0x86af, 0x86c9, 0x8853, 0x889e,
00389 0x8888, 0x88ab, 0x8892, 0x8896, 0x888d, 0x888b, 0x8993, 0x898f,
00390 0x8a2a, 0x8a1d, 0x8a23, 0x8a25, 0x8a31, 0x8a2d, 0x8a1f, 0x8a1b,
00391 0x8a22, 0x8c49, 0x8c5a, 0x8ca9, 0x8cac, 0x8cab, 0x8ca8, 0x8caa,
00392 0x8ca7, 0x8d67, 0x8d66, 0x8dbe, 0x8dba, 0x8edb, 0x8edf, 0x9019,
00393 0x900d, 0x901a, 0x9017, 0x9023, 0x901f, 0x901d, 0x9010, 0x9015,
00394 0x901e, 0x9020, 0x900f, 0x9022, 0x9016, 0x901b, 0x9014, 0x90e8,
00395 0x90ed, 0x90fd, 0x9157, 0x9157, 0x91ce, 0x91f5, 0x91e6, 0x91e3, 0x91e7,
00396 0x91ed, 0x91e9, 0x9589, 0x966a, 0x9675, 0x9673, 0x9678, 0x9670,
00397 0x9674, 0x9676, 0x9677, 0x966c, 0x96c0, 0x96ea, 0x96e9, 0x7ae0,
00398 0x7adf, 0x9802, 0x9803, 0x9803, 0x9b5a, 0x9ce5, 0x9e75, 0x9e7f, 0x9ea5,
00399 0x9ebb, 0x50a2, 0x508d, 0x5085, 0x5099, 0x5091, 0x5080, 0x5096,
00400 0x5098, 0x509a, 0x6700, 0x51f1, 0x5272, 0x5274, 0x5275, 0x5269,
00401 0x52de, 0x52dd, 0x52db, 0x535a, 0x535a, 0x557b, 0x5580, 0x55a7,
00402 0x557c, 0x558a, 0x559d, 0x5598, 0x5582, 0x559c, 0x55aa, 0x5594,
00403 0x5587, 0x558b, 0x5583, 0x55b3, 0x55ae, 0x559f, 0x553e, 0x55b2,
00404 0x559a, 0x55bb, 0x55ac, 0x55b1, 0x557e, 0x5589, 0x55ab, 0x5599,
00405 0x570d, 0x582f, 0x582a, 0x5834, 0x5824, 0x5830, 0x5831, 0x5821,
00406 0x581d, 0x5820, 0x58f9, 0x58fa, 0x5960,
00407 /* 0xb4 */
00408 0x5a77, 0x5a9a, 0x5a7f, 0x5a92, 0x5a9b, 0x5aa7, 0x5b73, 0x5b71,
00409 0x5bd2, 0x5bcc, 0x5bd3, 0x5bd0, 0x5c0a, 0x5c0b, 0x5c31, 0x5d4c,
00410 0x5d50, 0x5d3d, 0x5d47, 0x5dfd, 0x5e45, 0x5e3d, 0x5e40, 0x5e43,
00411 0x5e7e, 0x5eca, 0x5ec1, 0x5ec2, 0x5ec4, 0x5f3c, 0x5f6d, 0x5fa9,
00412 0x5faa, 0x5fa8, 0x60d1, 0x60e1, 0x60b2, 0x60b6, 0x60e0, 0x611c,
00413 0x6123, 0x60fa, 0x6115, 0x60f0, 0x60fb, 0x60f4, 0x6168, 0x60f1,
00414 0x610e, 0x60f6, 0x6109, 0x6100, 0x6112, 0x621f, 0x6249, 0x63a3,
00415 0x638c, 0x63cf, 0x63c0, 0x63e9, 0x63c9, 0x63c6, 0x63cd, 0x63d2,
00416 0x63e3, 0x63d0, 0x63e1, 0x63d6, 0x63ed, 0x63ee, 0x6376, 0x63f4,
00417 0x63ea, 0x63db, 0x6452, 0x63da, 0x63f9, 0x655e, 0x6566, 0x6562,
00418 0x6563, 0x6591, 0x6590, 0x65af, 0x666e, 0x6670, 0x6674, 0x6676,
00419 0x666f, 0x669d, 0x667a, 0x667e, 0x6677, 0x66fe, 0x66ff, 0x671f,
00420 0x671d, 0x68fa, 0x68d5, 0x68e0, 0x68d8, 0x68d7, 0x6905, 0x68df,
00421 0x68f5, 0x68ee, 0x68e7, 0x68f9, 0x68d2, 0x68f2, 0x68e3, 0x68cb,
00422 0x68cd, 0x690d, 0x6912, 0x690e, 0x68c9, 0x68da, 0x696e, 0x68fb,
00423 0x6b3e, 0x6b3a, 0x6b3d, 0x6b98, 0x6b96, 0x6bbc, 0x6bef, 0x6c2e,
00424 0x6c2f, 0x6c2c, 0x6e2f, 0x6e38, 0x6e54, 0x6e21, 0x6e32, 0x6e67,
00425 0x6e4a, 0x6e20, 0x6e25, 0x6e23, 0x6e1b, 0x6e5b, 0x6e58, 0x6e24,
00426 0x6e56, 0x6e6e, 0x6e2d, 0x6e26, 0x6e6f, 0x6e34, 0x6e4d, 0x6e3a,
00427 0x6e2c, 0x6e43, 0x6e1d, 0x6e3e, 0x6ecb,
00428 /* 0xb5 */
00429 0x6e89, 0x6e19, 0x6e4e, 0x6e63, 0x6e44, 0x6e72, 0x6e69, 0x6e5f,
00430 0x7119, 0x711a, 0x7126, 0x7130, 0x7121, 0x7136, 0x716e, 0x711c,
00431 0x724c, 0x7284, 0x7280, 0x7336, 0x7325, 0x7334, 0x7329, 0x743a,
00432 0x742a, 0x7433, 0x7422, 0x7425, 0x7435, 0x7436, 0x7434, 0x742f,
00433 0x741b, 0x7426, 0x7428, 0x7525, 0x7526, 0x756b, 0x756a, 0x75e2,
00434 0x75db, 0x75e3, 0x75d8, 0x75de, 0x75e0, 0x767b, 0x767c, 0x767e,
00435 0x7696, 0x7693, 0x76b4, 0x76dc, 0x774f, 0x77ed, 0x785d, 0x786c,
00436 0x786f, 0x7a0d, 0x7a08, 0x7a0b, 0x7a05, 0x7a00, 0x7a98, 0x7a97,
00437 0x7a96, 0x7ae3, 0x7ae3, 0x7b49, 0x7b56, 0x7b46, 0x7b50, 0x7b52,
00438 0x7b54, 0x7b4d, 0x7b4b, 0x7b4f, 0x7b51, 0x7c9f, 0x7ca5, 0x7d5e,
00439 0x7d50, 0x7d68, 0x7d55, 0x7d2b, 0x7d6e, 0x7d72, 0x7d61, 0x7d66,
00440 0x7d62, 0x7d70, 0x7d73, 0x5584, 0x7fd4, 0x7fd5, 0x800b, 0x8052,
00441 0x8085, 0x8155, 0x8154, 0x814b, 0x8151, 0x814e, 0x8139, 0x8146,
00442 0x813e, 0x814c, 0x8153, 0x8174, 0x8212, 0x821c, 0x83e9, 0x8403,
00443 0x83f8, 0x8400, 0x83e0, 0x83c5, 0x840b, 0x83c1, 0x83ef, 0x83f1,
00444 0x83f4, 0x8457, 0x840a, 0x83f0, 0x840c, 0x83cc, 0x83fd, 0x83f2,
00445 0x83ca, 0x8438, 0x840e, 0x8404, 0x83dc, 0x8407, 0x83d4, 0x83df,
00446 0x865b, 0x86df, 0x86d9, 0x86ed, 0x86d4, 0x86db, 0x86e4, 0x86d0,
00447 0x86de, 0x8857, 0x88c1, 0x88c2, 0x88b1, 0x8983, 0x8996, 0x8a3b,
00448 0x8a60, 0x8a55, 0x8a5e, 0x8a3c, 0x8a41,
00449 /* 0xb6 */
00450 0x8a54, 0x8a5b, 0x8a50, 0x8a46, 0x8a34, 0x8a3a, 0x8a36, 0x8a56,
00451 0x8c61, 0x8c82, 0x8caf, 0x8cbc, 0x8cb3, 0x8cbd, 0x8cc1, 0x8cbb,
00452 0x8cc0, 0x8cb4, 0x8cb7, 0x8cb6, 0x8cbf, 0x8cb8, 0x8d8a, 0x8d85,
00453 0x8d81, 0x8dce, 0x8ddd, 0x8dc3, 0x8dda, 0x8dd1, 0x8dcd, 0x8ddb,
00454 0x8dc6, 0x8efb, 0x8ef8, 0x8efc, 0x8f9c, 0x902e, 0x9035, 0x9031,
00455 0x9038, 0x9032, 0x9036, 0x9102, 0x90f5, 0x9109, 0x90fe, 0x9163,
00456 0x9165, 0x91cf, 0x9214, 0x9215, 0x9223, 0x9209, 0x921e, 0x920d,
00457 0x9210, 0x9207, 0x9211, 0x9594, 0x958f, 0x958b, 0x9591, 0x9593,
00458 0x9592, 0x958e, 0x968a, 0x968e, 0x968b, 0x967d, 0x9685, 0x9686,
00459 0x968d, 0x9672, 0x9684, 0x96c1, 0x96c5, 0x96c4, 0x96c6, 0x96c7,
00460 0x96ef, 0x96f2, 0x97cc, 0x9805, 0x9806, 0x9808, 0x98e7, 0x98ea,
00461 0x98ef, 0x98e9, 0x98f2, 0x98ed, 0x99ae, 0x99ad, 0x9ec3, 0x9ecd,
```

```

00462 0x9ed1, 0x4e82, 0x50ad, 0x50b5, 0x50b2, 0x50b3, 0x50c5, 0x50be,
00463 0x50ac, 0x50b7, 0x50bb, 0x50af, 0x50c7, 0x527f, 0x5277, 0x527d,
00464 0x52df, 0x52e6, 0x52e4, 0x52e2, 0x52e3, 0x532f, 0x55df, 0x55e8,
00465 0x55d3, 0x55e6, 0x55ce, 0x55dc, 0x55c7, 0x55d1, 0x55e3, 0x55e4,
00466 0x55ef, 0x55da, 0x55e1, 0x55c5, 0x55c6, 0x55e5, 0x55c9, 0x5712,
00467 0x5713, 0x585e, 0x5851, 0x5858, 0x5857, 0x585a, 0x5854, 0x586b,
00468 0x584c, 0x586d, 0x584a, 0x5862, 0x5852, 0x584b, 0x5967, 0x5ac1,
00469 0x5ac9, 0x5acc, 0x5abe, 0x5abd, 0x5abc,
00470 /* 0xb7 */
00471 0x5ab3, 0x5ac2, 0x5ab2, 0x5d69, 0x5d6f, 0x5e4c, 0x5e79, 0x5ec9,
00472 0x5ec8, 0x5f12, 0x5f59, 0x5fac, 0x5fae, 0x611a, 0x610f, 0x6148,
00473 0x611f, 0x60f3, 0x611b, 0x60f9, 0x6101, 0x6108, 0x614e, 0x614c,
00474 0x6144, 0x614d, 0x613e, 0x6134, 0x6127, 0x610d, 0x6106, 0x6137,
00475 0x6221, 0x6222, 0x6413, 0x643e, 0x641e, 0x642a, 0x642d, 0x643d,
00476 0x642c, 0x640f, 0x641c, 0x6414, 0x640d, 0x6436, 0x6416, 0x6417,
00477 0x6406, 0x656c, 0x659f, 0x65b0, 0x6697, 0x6689, 0x6687, 0x6688,
00478 0x6696, 0x6684, 0x6698, 0x668d, 0x6703, 0x6994, 0x696d, 0x695a,
00479 0x6977, 0x6960, 0x6954, 0x6954, 0x6975, 0x6930, 0x6982, 0x694a, 0x6968,
00480 0x696b, 0x695e, 0x6953, 0x6979, 0x6986, 0x695d, 0x6963, 0x695b,
00481 0x6b47, 0x6b72, 0x6bc0, 0x6bbf, 0x6bd3, 0x6bfd, 0x6ea2, 0x6eaf,
00482 0x6ed3, 0x6ed3, 0x6ec2, 0x6e90, 0x6e9d, 0x6ec7, 0x6ec5, 0x6ea5,
00483 0x6e98, 0x6ebc, 0x6eba, 0x6eab, 0x6ed1, 0x6e96, 0x6e9c, 0x6ec4,
00484 0x6ed4, 0x6eaa, 0x6ea7, 0x6eb4, 0x714e, 0x7159, 0x7169, 0x7164,
00485 0x7149, 0x7167, 0x715c, 0x715c, 0x716c, 0x716c, 0x714c, 0x7165, 0x715e,
00486 0x7146, 0x7168, 0x7156, 0x723a, 0x7252, 0x7337, 0x7345, 0x733f,
00487 0x733e, 0x746f, 0x745a, 0x7455, 0x745f, 0x745e, 0x7441, 0x743f,
00488 0x7459, 0x745b, 0x745c, 0x7456, 0x7578, 0x7600, 0x75f0, 0x7601,
00489 0x75f2, 0x75f1, 0x75fa, 0x75ff, 0x75f4, 0x75f3, 0x76de, 0x76df,
00490 0x775b, 0x776b, 0x7766, 0x775e, 0x7763,
00491 /* 0xb8 */
00492 0x7779, 0x776a, 0x776c, 0x775c, 0x7765, 0x7768, 0x7762, 0x77ee,
00493 0x788e, 0x78b0, 0x7897, 0x7898, 0x788c, 0x7889, 0x787c, 0x7891,
00494 0x7893, 0x787f, 0x797a, 0x797f, 0x7981, 0x842c, 0x79bd, 0x7a1c,
00495 0x7a1a, 0x7a20, 0x7a14, 0x7a1f, 0x7a1e, 0x7a9f, 0x7aa0, 0x7b77,
00496 0x7bc0, 0x7b60, 0x7b6e, 0x7b67, 0x7cb1, 0x7cb3, 0x7cb5, 0x7d93,
00497 0x7d79, 0x7d91, 0x7d81, 0x7d8f, 0x7d5b, 0x7f6e, 0x7f69, 0x7f6a,
00498 0x7f72, 0x7fa9, 0x7fa8, 0x7fa4, 0x8056, 0x8058, 0x8086, 0x8084,
00499 0x8171, 0x8170, 0x8178, 0x8165, 0x816e, 0x8173, 0x816b, 0x8179,
00500 0x817a, 0x8166, 0x8205, 0x8247, 0x8482, 0x8477, 0x843d, 0x8431,
00501 0x8475, 0x8466, 0x846b, 0x8449, 0x846c, 0x845b, 0x843c, 0x8435,
00502 0x8461, 0x8463, 0x8469, 0x846d, 0x8446, 0x865e, 0x865c, 0x865f,
00503 0x86f9, 0x8713, 0x8708, 0x8707, 0x8700, 0x86fe, 0x86fb, 0x8702,
00504 0x8703, 0x8706, 0x870a, 0x8859, 0x88df, 0x88d4, 0x88d9, 0x88dc,
00505 0x88d8, 0x88dd, 0x88e1, 0x88ca, 0x88d5, 0x88d2, 0x899c, 0x89e3,
00506 0x8a6b, 0x8a72, 0x8a73, 0x8a66, 0x8a69, 0x8a70, 0x8a87, 0x8a7c,
00507 0x8a63, 0x8aa0, 0x8a71, 0x8a85, 0x8a6d, 0x8a62, 0x8a6e, 0x8a6c,
00508 0x8a79, 0x8a7b, 0x8a3e, 0x8a68, 0x8c62, 0x8c8a, 0x8c89, 0x8cca,
00509 0x8cc7, 0x8cc8, 0x8cc4, 0x8cb2, 0x8cc3, 0x8cc2, 0x8cc5, 0x8de1,
00510 0x8ddf, 0x8de8, 0x8def, 0x8df3, 0x8dfa, 0x8dea, 0x8de4, 0x8de6,
00511 0x8eb2, 0x8f03, 0x8f09, 0x8efe, 0x8f0a,
00512 /* 0xb9 */
00513 0x8f9f, 0x8fb2, 0x904b, 0x904a, 0x9053, 0x9042, 0x9054, 0x903c,
00514 0x9055, 0x9050, 0x9047, 0x904f, 0x904e, 0x904d, 0x9051, 0x903e,
00515 0x9041, 0x9112, 0x9117, 0x916c, 0x916a, 0x9169, 0x91c9, 0x9237,
00516 0x9257, 0x9238, 0x923d, 0x9240, 0x923e, 0x925b, 0x924b, 0x9264,
00517 0x9251, 0x9234, 0x9249, 0x924d, 0x9245, 0x9239, 0x923f, 0x925a,
00518 0x9598, 0x9698, 0x9694, 0x9695, 0x96cd, 0x96cb, 0x96c9, 0x96ca,
00519 0x96f7, 0x96fb, 0x96f9, 0x96f6, 0x9756, 0x9774, 0x9776, 0x9810,
00520 0x9811, 0x9813, 0x980a, 0x9812, 0x980c, 0x98fc, 0x98f4, 0x98fd,
00521 0x98fe, 0x99b3, 0x99b1, 0x99b4, 0x9ae1, 0x9ce9, 0x9e82, 0x9f0e,
00522 0x9f13, 0x9f20, 0x50e7, 0x50e5, 0x50e5, 0x50d6, 0x50ed, 0x50da,
00523 0x50d5, 0x50cf, 0x50d1, 0x50f1, 0x50ce, 0x50e9, 0x5162, 0x51f3,
00524 0x5283, 0x5282, 0x5331, 0x53ad, 0x55fe, 0x5600, 0x561b, 0x5617,
00525 0x55fd, 0x5614, 0x5606, 0x5609, 0x560d, 0x560e, 0x55f7, 0x5616,
00526 0x561f, 0x5608, 0x5610, 0x55f6, 0x5718, 0x5716, 0x5875, 0x587e,
00527 0x5883, 0x5893, 0x588a, 0x5879, 0x5885, 0x587d, 0x58fd, 0x5925,
00528 0x5922, 0x5924, 0x596a, 0x5969, 0x5ae1, 0x5ae6, 0x5ae9, 0x5ad7,
00529 0x5ad6, 0x5ad8, 0x5ae3, 0x5b75, 0x5bde, 0x5be7, 0x5be1, 0x5be5,
00530 0x5be6, 0x5be8, 0x5be2, 0x5be4, 0x5bdf, 0x5c0d, 0x5c62, 0x5d84,
00531 0x5d87, 0x5e5b, 0x5e63, 0x5e55, 0x5e57, 0x5e54, 0x5ed3, 0x5ed6,
00532 0x5f0a, 0x5f46, 0x5f70, 0x5fb9, 0x6147,
00533 /* 0xba */
00534 0x613f, 0x614b, 0x6177, 0x6162, 0x6163, 0x615f, 0x615a, 0x6158,
00535 0x6175, 0x622a, 0x6487, 0x6458, 0x6454, 0x64a4, 0x6478, 0x645f,
00536 0x647a, 0x6451, 0x6467, 0x6434, 0x646d, 0x647b, 0x6572, 0x65a1,
00537 0x65d7, 0x65d6, 0x66a2, 0x66a8, 0x669d, 0x669c, 0x69a8, 0x6995,
00538 0x69c1, 0x69ae, 0x69d3, 0x69cb, 0x699b, 0x69b7, 0x69bb, 0x69ab,
00539 0x69b4, 0x69d0, 0x69cd, 0x69ad, 0x69cc, 0x69a6, 0x69c3, 0x69a3,
00540 0x6b49, 0x6b4c, 0x6c33, 0x6f33, 0x6f14, 0x6efe, 0x6f13, 0x6ef4,
00541 0x6f29, 0x6f3e, 0x6f20, 0x6f2c, 0x6f0f, 0x6f02, 0x6f22, 0x6eff,
00542 0x6eef, 0x6f06, 0x6f31, 0x6f38, 0x6f32, 0x6f23, 0x6f15, 0x6f2b,
00543 0x6f2f, 0x6f88, 0x6f2a, 0x6eec, 0x6f01, 0x6ef2, 0x6ecc, 0x6ef7,
00544 0x7194, 0x7199, 0x717d, 0x718a, 0x7184, 0x7192, 0x723e, 0x7292,
00545 0x7296, 0x7344, 0x7350, 0x7464, 0x7463, 0x746a, 0x7470, 0x746d,
00546 0x7504, 0x7591, 0x7627, 0x760d, 0x760b, 0x7609, 0x7613, 0x76e1,
00547 0x76e3, 0x7784, 0x777d, 0x777f, 0x7761, 0x78c1, 0x789f, 0x78a7,
00548 0x78b3, 0x78a9, 0x78a3, 0x798e, 0x798f, 0x798d, 0x7a2e, 0x7a31,

```

```
00549 0x7aaa, 0x7aa9, 0x7aed, 0x7aef, 0x7ba1, 0x7b95, 0x7b8b, 0x7b75,
00550 0x7b97, 0x7b9d, 0x7b94, 0x7b8f, 0x7bb8, 0x7b87, 0x7b84, 0x7cb9,
00551 0x7cbd, 0x7cbe, 0x7dbb, 0x7db0, 0x7d9c, 0x7dbd, 0x7dbe, 0x7da0,
00552 0x7dca, 0x7db4, 0x7db2, 0x7db1, 0x7dba, 0x7da2, 0x7dbf, 0x7db5,
00553 0x7db8, 0x7dad, 0x7dd2, 0x7dc7, 0x7dac,
00554 /* 0xbb */
00555 0x7f70, 0x7fe0, 0x7fe1, 0x7fdf, 0x805e, 0x805a, 0x8087, 0x8150,
00556 0x8180, 0x818f, 0x8188, 0x818a, 0x817f, 0x8182, 0x81e7, 0x81fa,
00557 0x8207, 0x8214, 0x821e, 0x824b, 0x84c9, 0x84bf, 0x84c6, 0x84c4,
00558 0x8499, 0x849e, 0x84b2, 0x849c, 0x84cb, 0x84b8, 0x84c0, 0x84d3,
00559 0x8490, 0x84bc, 0x84d1, 0x84ca, 0x873f, 0x871c, 0x873b, 0x8722,
00560 0x8725, 0x8734, 0x8718, 0x8755, 0x8737, 0x8729, 0x88f3, 0x8902,
00561 0x88f4, 0x88f9, 0x88f8, 0x88fd, 0x88e8, 0x891a, 0x88ef, 0x8aa6,
00562 0x8a8c, 0x8a9e, 0x8aa3, 0x8a8d, 0x8aa1, 0x8a93, 0x8aa4, 0x8aaa,
00563 0x8aa5, 0x8aa8, 0x8a98, 0x8a91, 0x8a9a, 0x8aa7, 0x8c6a, 0x8c8d,
00564 0x8c8c, 0x8cd3, 0x8cd1, 0x8cd2, 0x8d6b, 0x8d99, 0x8d95, 0x8dfc,
00565 0x8f14, 0x8f12, 0x8f15, 0x8f13, 0x8fa3, 0x9060, 0x9058, 0x905c,
00566 0x9063, 0x9059, 0x905e, 0x9062, 0x905d, 0x905b, 0x9119, 0x9118,
00567 0x911e, 0x9175, 0x9178, 0x9177, 0x9174, 0x9278, 0x9280, 0x9285,
00568 0x9298, 0x9296, 0x927b, 0x9293, 0x929c, 0x92a8, 0x927c, 0x9291,
00569 0x95a1, 0x95a8, 0x95a9, 0x95a3, 0x95a5, 0x95a4, 0x9699, 0x969c,
00570 0x969b, 0x96cc, 0x96d2, 0x9700, 0x977c, 0x9785, 0x97f6, 0x9817,
00571 0x9818, 0x98af, 0x98b1, 0x9903, 0x9905, 0x990c, 0x9909, 0x99c1,
00572 0x9aaf, 0x9ab0, 0x9ae6, 0x9b41, 0x9b42, 0x9cf4, 0x9cf6, 0x9cf3,
00573 0x9ebc, 0x9f3b, 0x9f4a, 0x5104, 0x5100, 0x50fb, 0x50f5, 0x50f9,
00574 0x5102, 0x5108, 0x5109, 0x5105, 0x51dc,
00575 /* 0xbc */
00576 0x5287, 0x5288, 0x5289, 0x528d, 0x528a, 0x52f0, 0x53b2, 0x562e,
00577 0x563b, 0x5639, 0x5632, 0x563f, 0x5634, 0x5629, 0x5653, 0x564e,
00578 0x5657, 0x5674, 0x5636, 0x562f, 0x5630, 0x5680, 0x589f, 0x589e,
00579 0x58b3, 0x589c, 0x58ae, 0x58a9, 0x58a6, 0x596d, 0x5b09, 0x5afb,
00580 0x5b0b, 0x5af5, 0x5b0c, 0x5b08, 0x5bee, 0x5bec, 0x5be9, 0x5beb,
00581 0x5c64, 0x5c65, 0x5d9d, 0x5d94, 0x5e62, 0x5e5f, 0x5e61, 0x5ee2,
00582 0x5eda, 0x5edf, 0x5edd, 0x5ee3, 0x5ee0, 0x5f48, 0x5f71, 0x5fb7,
00583 0x5fb5, 0x6176, 0x6167, 0x616e, 0x615d, 0x6155, 0x6182, 0x617c,
00584 0x6170, 0x616b, 0x617e, 0x61a7, 0x6190, 0x61ab, 0x618e, 0x61ac,
00585 0x619a, 0x61a4, 0x6194, 0x61ae, 0x622e, 0x6469, 0x646f, 0x6479,
00586 0x649e, 0x64b2, 0x6488, 0x6490, 0x64b0, 0x64a5, 0x6493, 0x6495,
00587 0x64a9, 0x6492, 0x64ae, 0x64ad, 0x64ab, 0x649a, 0x64ac, 0x6499,
00588 0x64a2, 0x64b3, 0x6575, 0x6577, 0x6578, 0x66ae, 0x66ab, 0x66b4,
00589 0x66b1, 0x6a23, 0x6a1f, 0x69e8, 0x6a01, 0x6a1e, 0x6a19, 0x69fd,
00590 0x6a21, 0x6a13, 0x6a0a, 0x69f3, 0x6a02, 0x6a05, 0x69ed, 0x6a11,
00591 0x6b50, 0x6b4e, 0x6ba4, 0x6bc5, 0x6bc6, 0x6f3f, 0x6f7c, 0x6f84,
00592 0x6f51, 0x6f66, 0x6f54, 0x6f86, 0x6f6d, 0x6f5b, 0x6f78, 0x6f6e,
00593 0x6f8e, 0x6f7a, 0x6f70, 0x6f64, 0x6f97, 0x6f58, 0x6ed5, 0x6f6f,
00594 0x6f60, 0x6f5f, 0x719f, 0x71ac, 0x71b1, 0x71a8, 0x7256, 0x729b,
00595 0x734e, 0x7357, 0x7469, 0x748b, 0x7483,
00596 /* 0xbd */
00597 0x747e, 0x7480, 0x757f, 0x7620, 0x7629, 0x761f, 0x7624, 0x7626,
00598 0x7621, 0x7622, 0x769a, 0x76ba, 0x76e4, 0x778e, 0x7787, 0x778c,
00599 0x7791, 0x778b, 0x778c, 0x78cb, 0x78c5, 0x78ba, 0x78ca, 0x78be, 0x78d5,
00600 0x78bc, 0x78d0, 0x7a3f, 0x7a3c, 0x7a40, 0x7a3d, 0x7a37, 0x7a3b,
00601 0x7aaf, 0x7aae, 0x7bad, 0x7bb1, 0x7bc4, 0x7bb4, 0x7bc6, 0x7bc7,
00602 0x7bc1, 0x7ba0, 0x7bcc, 0x7cca, 0x7de0, 0x7df4, 0x7def, 0x7dfb,
00603 0x7dd8, 0x7dec, 0x7ddd, 0x7de8, 0x7de3, 0x7dda, 0x7dde, 0x7de9,
00604 0x7d9e, 0x7dd9, 0x7df2, 0x7df9, 0x7f75, 0x7f77, 0x7faf, 0x7fe9,
00605 0x8026, 0x819b, 0x819c, 0x819d, 0x81a0, 0x819a, 0x8198, 0x8517,
00606 0x853d, 0x851a, 0x84ee, 0x852c, 0x852d, 0x8513, 0x8511, 0x8523,
00607 0x8521, 0x8514, 0x84ec, 0x8525, 0x84ff, 0x8506, 0x8782, 0x8774,
00608 0x8776, 0x8760, 0x8778, 0x8778, 0x8768, 0x8759, 0x8757, 0x874c,
00609 0x8753, 0x885b, 0x885d, 0x8910, 0x8907, 0x8912, 0x8913, 0x8915,
00610 0x890a, 0x8abc, 0x8ad2, 0x8ac7, 0x8ac4, 0x8a95, 0x8acb, 0x8af8,
00611 0x8ab2, 0x8ac9, 0x8ac2, 0x8abf, 0x8ab0, 0x8ad6, 0x8acd, 0x8ab6,
00612 0x8ab9, 0x8adb, 0x8c4c, 0x8c4e, 0x8c6c, 0x8ce0, 0x8cde, 0x8ce6,
00613 0x8ce4, 0x8cec, 0x8ced, 0x8ce2, 0x8ce3, 0x8cdc, 0x8cea, 0x8cef,
00614 0x8d6d, 0x8d9f, 0x8da3, 0x8e2b, 0x8e10, 0x8e1d, 0x8e22, 0x8e0f,
00615 0x8e29, 0x8e1f, 0x8e21, 0x8e1e, 0x8eba, 0x8f1d, 0x8f1b, 0x8f1f,
00616 0x8f29, 0x8f26, 0x8f2a, 0x8f1c, 0x8f1e,
00617 /* 0xbe */
00618 0x8f25, 0x9069, 0x906e, 0x9068, 0x906d, 0x9077, 0x9130, 0x912d,
00619 0x9127, 0x9131, 0x9187, 0x9189, 0x918b, 0x9183, 0x92c5, 0x92bb,
00620 0x92b7, 0x92ea, 0x92ac, 0x92e4, 0x92c1, 0x92b3, 0x92bc, 0x92d2,
00621 0x92c7, 0x92f0, 0x92b2, 0x95ad, 0x95b1, 0x9704, 0x9706, 0x9707,
00622 0x9709, 0x9760, 0x978d, 0x978b, 0x978f, 0x9821, 0x982b, 0x981c,
00623 0x98b3, 0x990a, 0x9913, 0x9912, 0x9918, 0x99dd, 0x99d0, 0x99df,
00624 0x99db, 0x99d1, 0x99d5, 0x99d2, 0x99d9, 0x9ab7, 0x9aee, 0x9aef,
00625 0x9b27, 0x9b45, 0x9b44, 0x9b77, 0x9b6f, 0x9d06, 0x9d09, 0x9d03,
00626 0x9ea9, 0x9ebe, 0x9ece, 0x58a8, 0x9f52, 0x5112, 0x5118, 0x5114,
00627 0x5110, 0x5115, 0x5180, 0x51aa, 0x51dd, 0x5291, 0x5293, 0x52f3,
00628 0x5659, 0x566b, 0x5679, 0x5669, 0x5664, 0x5678, 0x566a, 0x5668,
00629 0x5665, 0x5671, 0x566f, 0x566c, 0x5662, 0x5676, 0x58c1, 0x58be,
00630 0x58c7, 0x58c5, 0x596e, 0x5b1d, 0x5b34, 0x5b78, 0x5bf0, 0x5c0e,
00631 0x5f4a, 0x61b2, 0x6191, 0x61a9, 0x618a, 0x61cd, 0x61b6, 0x61be,
00632 0x61ca, 0x61c8, 0x6230, 0x64c5, 0x64c1, 0x64cb, 0x64bb, 0x64bc,
00633 0x64da, 0x64ca, 0x64c7, 0x64c2, 0x64cd, 0x64bf, 0x64d2, 0x64d4,
00634 0x64be, 0x6574, 0x66c6, 0x66c9, 0x66b9, 0x66c4, 0x66c7, 0x66b8,
00635 0x6a3d, 0x6a38, 0x6a3a, 0x6a59, 0x6a6b, 0x6a58, 0x6a39, 0x6a44,
```



```

00636 0x6a62, 0x6a61, 0x6a4b, 0x6a47, 0x6a35, 0x6a5f, 0x6a48, 0x6b59,
00637 0x6b77, 0x6c05, 0x6fc2, 0x6fb1, 0x6fa1,
00638 /* 0xbf */
00639 0x6fc3, 0x6fa4, 0x6fc1, 0x6fa7, 0x6fb3, 0x6fc0, 0x6fb9, 0x6fb6,
00640 0x6fa6, 0x6fa0, 0x6fb4, 0x71be, 0x71c9, 0x71d0, 0x71d2, 0x71c8,
00641 0x71d5, 0x71b9, 0x71ce, 0x71d9, 0x71dc, 0x71c3, 0x71c4, 0x7368,
00642 0x749c, 0x74a3, 0x7498, 0x749f, 0x749e, 0x74e2, 0x750c, 0x750d,
00643 0x7634, 0x7638, 0x763a, 0x76e7, 0x76e5, 0x77a0, 0x779e, 0x779f,
00644 0x77a5, 0x78e8, 0x78da, 0x78ec, 0x78e7, 0x79a6, 0x7a4d, 0x7a4e,
00645 0x7a46, 0x7a4c, 0x7a4b, 0x7aba, 0x7bd9, 0x7c11, 0x7bc9, 0x7be4,
00646 0x7bdb, 0x7be1, 0x7be9, 0x7be6, 0x7cd5, 0x7cd6, 0x7e0a, 0x7e11,
00647 0x7e08, 0x7e1b, 0x7e23, 0x7e1e, 0x7e1d, 0x7e09, 0x7e10, 0x7f79,
00648 0x7fb2, 0x7ff0, 0x7ff1, 0x7fee, 0x8028, 0x81b3, 0x81a9, 0x81a8,
00649 0x81fb, 0x8208, 0x8258, 0x8259, 0x854a, 0x8559, 0x8548, 0x8568,
00650 0x8569, 0x8543, 0x8549, 0x856d, 0x856a, 0x855e, 0x8783, 0x879f,
00651 0x879e, 0x87a2, 0x878d, 0x8861, 0x892a, 0x8932, 0x8925, 0x892b,
00652 0x8921, 0x89aa, 0x89a6, 0x8ae6, 0x8afa, 0x8aeb, 0x8af1, 0x8b00,
00653 0x8adc, 0x8ae7, 0x8aee, 0x8afe, 0x8b01, 0x8b02, 0x8af7, 0x8aed,
00654 0x8af3, 0x8af6, 0x8afc, 0x8c6b, 0x8c6d, 0x8c93, 0x8cf4, 0x8e44,
00655 0x8e31, 0x8e34, 0x8e42, 0x8e39, 0x8e35, 0x8f3b, 0x8f2f, 0x8f38,
00656 0x8f33, 0x8fa8, 0x8fa6, 0x9075, 0x9074, 0x9078, 0x9072, 0x907c,
00657 0x907a, 0x9134, 0x9192, 0x9320, 0x9336, 0x92f8, 0x9333, 0x932f,
00658 0x9322, 0x92fc, 0x932b, 0x9304, 0x931a,
00659 /* 0xc0 */
00660 0x9310, 0x9326, 0x9321, 0x9315, 0x932e, 0x9319, 0x95bb, 0x96a7,
00661 0x96a8, 0x96aa, 0x96d5, 0x970e, 0x9711, 0x9716, 0x970d, 0x9713,
00662 0x970f, 0x975b, 0x975c, 0x9766, 0x9798, 0x9830, 0x9838, 0x983b,
00663 0x9837, 0x982d, 0x9839, 0x9824, 0x9910, 0x9928, 0x991e, 0x991b,
00664 0x9921, 0x991a, 0x99ed, 0x99e2, 0x99f1, 0x99ab, 0x99ab, 0x99af,
00665 0x9aed, 0x9b28, 0x9b91, 0x9d15, 0x9d23, 0x9d26, 0x9d2d, 0x9d12,
00666 0x9d1b, 0x9ed8, 0x9ed4, 0x9f8d, 0x9f9c, 0x512a, 0x511f, 0x5121,
00667 0x5132, 0x52f5, 0x568e, 0x5680, 0x5690, 0x5685, 0x5687, 0x568f,
00668 0x58d5, 0x58d3, 0x58d1, 0x58ce, 0x5b30, 0x5b2a, 0x5b24, 0x5b7a,
00669 0x5c37, 0x5c68, 0x5dbc, 0x5dba, 0x5dbd, 0x5db8, 0x5e6b, 0x5f4c,
00670 0x5fbd, 0x61c9, 0x61c2, 0x61c7, 0x61e6, 0x61cb, 0x6232, 0x6234,
00671 0x64ce, 0x64ca, 0x64d8, 0x64e0, 0x64f0, 0x64e6, 0x64ec, 0x64f1,
00672 0x64e2, 0x64ed, 0x6582, 0x6583, 0x66d9, 0x66d6, 0x6a80, 0x6a94,
00673 0x6a84, 0x6aa2, 0x6a9c, 0x6adb, 0x6aa3, 0x6a7e, 0x6a97, 0x6a90,
00674 0x6aa0, 0x6a5c, 0x6bae, 0x6bda, 0x6c08, 0x6fd8, 0x6ff1, 0x6fdf,
00675 0x6fe0, 0x6fdb, 0x6fe4, 0x6feb, 0x6fef, 0x6f80, 0x6fec, 0x6fe1,
00676 0x6fe9, 0x6fd5, 0x6fee, 0x6ff0, 0x71e7, 0x71df, 0x71ee, 0x71e6,
00677 0x71e5, 0x71ed, 0x71ec, 0x71f4, 0x71e0, 0x7235, 0x7246, 0x7370,
00678 0x7372, 0x74a9, 0x74b0, 0x74a6, 0x74a8, 0x7646, 0x7642, 0x764c,
00679 0x76ea, 0x77b3, 0x77aa, 0x77b0, 0x77ac,
00680 /* 0xc1 */
00681 0x77a7, 0x77ad, 0x77ef, 0x78f7, 0x78fa, 0x78f4, 0x78ef, 0x7901,
00682 0x79a7, 0x79aa, 0x7a57, 0x7abf, 0x7c07, 0x7c0d, 0x7bfe, 0x7bf7,
00683 0x7c0c, 0x7be0, 0x7ce0, 0x7cdc, 0x7cde, 0x7ce2, 0x7cd7, 0x7cd9,
00684 0x7cdd, 0x7e2e, 0x7e3e, 0x7e46, 0x7e37, 0x7e32, 0x7e43, 0x7e2b,
00685 0x7e3d, 0x7e31, 0x7e45, 0x7e41, 0x7e34, 0x7e39, 0x7e48, 0x7e35,
00686 0x7e3f, 0x7e2f, 0x7f44, 0x7ff3, 0x7ffc, 0x8071, 0x8072, 0x8070,
00687 0x806f, 0x8073, 0x81c6, 0x81c3, 0x81ba, 0x81c2, 0x81c0, 0x81bf,
00688 0x81bd, 0x81c9, 0x81be, 0x81e8, 0x8209, 0x8271, 0x85aa, 0x8584,
00689 0x857e, 0x859c, 0x8591, 0x8594, 0x85af, 0x859b, 0x8587, 0x858a,
00690 0x858a, 0x8667, 0x87c0, 0x87d1, 0x87b3, 0x87d2, 0x87c6, 0x87ab,
00691 0x87bb, 0x87ba, 0x87c8, 0x87cb, 0x893b, 0x8936, 0x8944, 0x8938,
00692 0x893d, 0x89ac, 0x8b0e, 0x8b17, 0x8b1b, 0x8b1b, 0x8b0a, 0x8b20,
00693 0x8b1d, 0x8b04, 0x8b10, 0x8c41, 0x8c3f, 0x8c73, 0x8cfa, 0x8cfd,
00694 0x8cfc, 0x8cf8, 0x8cfb, 0x8da8, 0x8e49, 0x8e4b, 0x8e48, 0x8e4a,
00695 0x8f44, 0x8f3e, 0x8f42, 0x8f45, 0x8f3f, 0x907f, 0x907d, 0x9084,
00696 0x9081, 0x9082, 0x9080, 0x9139, 0x91a3, 0x919e, 0x919c, 0x934d,
00697 0x9382, 0x9328, 0x9375, 0x934a, 0x9365, 0x934b, 0x9318, 0x937e,
00698 0x936c, 0x935b, 0x9370, 0x935a, 0x9354, 0x95ca, 0x95cb, 0x95cc,
00699 0x95c8, 0x95c6, 0x96b1, 0x96b8, 0x96d6, 0x971c, 0x971e, 0x97a0,
00700 0x97d3, 0x9846, 0x98b6, 0x9935, 0x9a01,
00701 /* 0xc2 */
00702 0x99ff, 0x9bae, 0x9bab, 0x9baa, 0x9bad, 0x9d3b, 0x9d3f, 0x9e8b,
00703 0x9ecf, 0x9ede, 0x9edc, 0x9edd, 0x9edb, 0x9f3e, 0x9f4b, 0x53e2,
00704 0x5695, 0x56ae, 0x58d9, 0x58d8, 0x5b38, 0x5f5d, 0x61e3, 0x6233,
00705 0x64f4, 0x64f2, 0x64fe, 0x6506, 0x64fa, 0x64fb, 0x64f7, 0x65b7,
00706 0x66dc, 0x6726, 0x6ab3, 0x6aac, 0x6ac3, 0x6abb, 0x6ab8, 0x6ac2,
00707 0x6aae, 0x6aaf, 0x6b5f, 0x6b78, 0x6baf, 0x7009, 0x700b, 0x6ffe,
00708 0x7006, 0x6ffa, 0x7011, 0x700f, 0x71fb, 0x71fc, 0x71fe, 0x71f8,
00709 0x7377, 0x7375, 0x74a7, 0x74bf, 0x7515, 0x7656, 0x7658, 0x7652,
00710 0x77bd, 0x77bf, 0x77bb, 0x77bc, 0x790e, 0x79ae, 0x7a61, 0x7a62,
00711 0x7a60, 0x7ac4, 0x7ac5, 0x7c2b, 0x7c27, 0x7c2a, 0x7c1e, 0x7c23,
00712 0x7c21, 0x7ce7, 0x7e54, 0x7e55, 0x7e5e, 0x7e5a, 0x7e61, 0x7e52,
00713 0x7e59, 0x7f48, 0x7ff9, 0x7ffb, 0x8077, 0x8076, 0x81cd, 0x81cf,
00714 0x820a, 0x85cf, 0x85a9, 0x85cd, 0x85d0, 0x85c9, 0x85b0, 0x85ba,
00715 0x85b9, 0x85a6, 0x87ef, 0x87ec, 0x87f2, 0x87e0, 0x8986, 0x89b2,
00716 0x89f4, 0x8b28, 0x8b39, 0x8b2c, 0x8b2b, 0x8c50, 0x8d05, 0x8e59,
00717 0x8e63, 0x8e66, 0x8e64, 0x8e5f, 0x8e55, 0x8ec0, 0x8f49, 0x8f4d,
00718 0x9087, 0x9083, 0x9088, 0x91ab, 0x91ac, 0x91d0, 0x9394, 0x938a,
00719 0x9396, 0x93a2, 0x93b3, 0x93ae, 0x93ac, 0x93b0, 0x9398, 0x939a,
00720 0x9397, 0x95d4, 0x95d6, 0x95d0, 0x95d5, 0x96e2, 0x96dc, 0x96d9,
00721 0x96db, 0x96de, 0x9724, 0x97a3, 0x97a6,
00722 /* 0xc3 */

```



```
00723 0x97ad, 0x97f9, 0x984d, 0x984f, 0x984c, 0x984e, 0x9853, 0x98ba,
00724 0x993e, 0x993f, 0x993d, 0x992e, 0x99a5, 0x9a0e, 0x9ac1, 0x9b03,
00725 0x9b06, 0x9b4f, 0x9b4e, 0x9b4d, 0x9bca, 0x9bc9, 0x9bfd, 0x9bc8,
00726 0x9bc0, 0x9d51, 0x9d5d, 0x9d60, 0x9ee0, 0x9f15, 0x9f2c, 0x5133,
00727 0x56a5, 0x58de, 0x58df, 0x58e2, 0x5bf5, 0x9f90, 0x5eec, 0x61f2,
00728 0x61f7, 0x61f6, 0x61f5, 0x6500, 0x650f, 0x66e0, 0x66dd, 0x6ae5,
00729 0x6add, 0x6ada, 0x6ad3, 0x701b, 0x701f, 0x7028, 0x701a, 0x701d,
00730 0x7015, 0x7018, 0x7206, 0x720d, 0x7258, 0x72a2, 0x7378, 0x737a,
00731 0x74bd, 0x74ca, 0x74e3, 0x7587, 0x7586, 0x765f, 0x7661, 0x77c7,
00732 0x7919, 0x79b1, 0x7a6b, 0x7a69, 0x7c3e, 0x7c3f, 0x7c38, 0x7c3d,
00733 0x7c37, 0x7c40, 0x7e6b, 0x7e6d, 0x7e79, 0x7e69, 0x7e6a, 0x7f85,
00734 0x7e73, 0x7fb6, 0x7fb9, 0x7fb8, 0x81d8, 0x85e9, 0x85dd, 0x85ea,
00735 0x85d5, 0x85e4, 0x85e5, 0x85f7, 0x87fb, 0x8805, 0x880d, 0x87f9,
00736 0x87fe, 0x8960, 0x895f, 0x8956, 0x895e, 0x8b41, 0x8b58, 0x8b58,
00737 0x8b49, 0x8b5a, 0x8b4e, 0x8b4f, 0x8b46, 0x8b59, 0x8d08, 0x8d0a,
00738 0x8e7c, 0x8e72, 0x8e87, 0x8e76, 0x8e6c, 0x8e7a, 0x8e74, 0x8f54,
00739 0x8f4e, 0x8fad, 0x908a, 0x908b, 0x91b1, 0x91ae, 0x93e1, 0x93d1,
00740 0x93df, 0x93c8, 0x93c3, 0x93dc, 0x93dd, 0x93d6, 0x93e2, 0x93dc,
00741 0x93d8, 0x93e4, 0x93d7, 0x93e8, 0x95dc, 0x96b4, 0x96e3, 0x972a,
00742 0x9727, 0x9761, 0x97dc, 0x97fb, 0x985e,
00743 /* 0xc4 */
00744 0x9858, 0x985b, 0x98bc, 0x9945, 0x9949, 0x9a16, 0x9a19, 0x9b0d,
00745 0x9be8, 0x9be7, 0x9bd6, 0x9bdb, 0x9d89, 0x9d61, 0x9d72, 0x9d6a,
00746 0x9d6c, 0x9e92, 0x9e97, 0x9eb4, 0x52f8, 0x56a8, 0x56b7,
00747 0x56b6, 0x56b4, 0x56bc, 0x58e4, 0x5b40, 0x5b43, 0x5b7d, 0x5bf6,
00748 0x5dc9, 0x61f8, 0x61fa, 0x6518, 0x6514, 0x6519, 0x66e6, 0x6727,
00749 0x6aec, 0x703e, 0x703c, 0x7030, 0x7032, 0x7210, 0x737b, 0x74cf, 0x7662,
00750 0x7665, 0x7926, 0x792a, 0x792c, 0x792b, 0x7ac7, 0x7af6, 0x7c4c,
00751 0x7c43, 0x7c4d, 0x7cef, 0x7cf0, 0x8fae, 0x7e7d, 0x7e7c, 0x7e82,
00752 0x7f4c, 0x8000, 0x81da, 0x8266, 0x85fb, 0x85f9, 0x8611, 0x85fa,
00753 0x8606, 0x860b, 0x8607, 0x860a, 0x8814, 0x8815, 0x8964, 0x89ba,
00754 0x89f8, 0x8b70, 0x8b6c, 0x8b66, 0x8b6f, 0x8b5f, 0x8b6b, 0x8d0f,
00755 0x8d0d, 0x8e89, 0x8e81, 0x8e85, 0x8e82, 0x91b4, 0x91cb, 0x9418,
00756 0x9403, 0x93fd, 0x95e1, 0x9730, 0x98c4, 0x9952, 0x9951, 0x99a8,
00757 0x9a2b, 0x9a30, 0x9a37, 0x9a35, 0x9c13, 0x9c0d, 0x9e79, 0x9eb5,
00758 0x9ee8, 0x9f2f, 0x9f5f, 0x9f63, 0x9f61, 0x5137, 0x5138, 0x56c1,
00759 0x56c0, 0x56c2, 0x5914, 0x5c6c, 0x5dcd, 0x61fc, 0x61fe, 0x651d,
00760 0x651c, 0x6595, 0x66e9, 0x6afb, 0x6b04, 0x6afa, 0x6bb2, 0x704c,
00761 0x721b, 0x72a7, 0x74d6, 0x74d4, 0x7669, 0x77d3, 0x7c50, 0x7e8f,
00762 0x7e8c, 0x7fbc, 0x8617, 0x862d, 0x861a, 0x8823, 0x8822, 0x8821,
00763 0x881f, 0x896a, 0x896c, 0x89bd, 0x8b74,
00764 /* 0xc5 */
00765 0x8b77, 0x8b7d, 0x8d13, 0x8e8a, 0x8e8d, 0x8e8b, 0x8f5f, 0x8faf,
00766 0x91ba, 0x942e, 0x9433, 0x9435, 0x943a, 0x9438, 0x9432, 0x942b,
00767 0x95e2, 0x9738, 0x9739, 0x9732, 0x97ff, 0x9867, 0x9865, 0x9957,
00768 0x9a45, 0x9a43, 0x9a40, 0x9a3e, 0x9acf, 0x9b54, 0x9b51, 0x9c2d,
00769 0x9c25, 0x9daf, 0x9db4, 0x9dc2, 0x9db8, 0x9e9d, 0x9eef, 0x9f19,
00770 0x9f5c, 0x9f66, 0x9f67, 0x513c, 0x513b, 0x56c8, 0x56ca, 0x56c9,
00771 0x5b7f, 0x5dd4, 0x5dd2, 0x5f4e, 0x61ff, 0x6524, 0x6b0a, 0x6b61,
00772 0x7051, 0x7058, 0x7380, 0x74e4, 0x758a, 0x766e, 0x766c, 0x79b3,
00773 0x7c60, 0x7c5f, 0x807e, 0x807d, 0x81df, 0x8972, 0x896f, 0x89c9,
00774 0x8b80, 0x8d16, 0x8d17, 0x8e91, 0x8e93, 0x8f61, 0x9148, 0x9444,
00775 0x9451, 0x9452, 0x973d, 0x973e, 0x97c3, 0x97c1, 0x986b, 0x9955,
00776 0x9a55, 0x9a4d, 0x9ad2, 0x9b1a, 0x9c49, 0x9c31, 0x9c3e, 0x9c3b,
00777 0x9dd3, 0x9dd7, 0x9f34, 0x9f6c, 0x9f6a, 0x9f94, 0x56cc, 0x5dd6,
00778 0x6200, 0x6523, 0x652b, 0x652a, 0x66ec, 0x6b10, 0x74da, 0x7aca,
00779 0x7c64, 0x7c63, 0x7c65, 0x7e93, 0x7e96, 0x7e94, 0x81e2, 0x8638,
00780 0x863f, 0x8831, 0x8b8a, 0x9090, 0x908f, 0x9463, 0x9460, 0x9464,
00781 0x9768, 0x986f, 0x995c, 0x9a5a, 0x9a5b, 0x9a57, 0x9ad3, 0x9ad4,
00782 0x9ad1, 0x9c54, 0x9c57, 0x9c56, 0x9de5, 0x9e9f, 0x9ef4, 0x56d1,
00783 0x58e9, 0x652c, 0x705e, 0x7671, 0x7672, 0x77d7, 0x7f50, 0x7f88,
00784 0x8836, 0x8839, 0x8862, 0x8b93, 0x8b92,
00785 /* 0xc6 */
00786 0x8b96, 0x8277, 0x8d1b, 0x91c0, 0x946a, 0x9742, 0x9748, 0x9744,
00787 0x97c6, 0x9870, 0x9a5f, 0x9b22, 0x9b58, 0x9c5f, 0x9df9, 0x9dfa,
00788 0x9e7c, 0x9e7d, 0x9f07, 0x9f77, 0x9f72, 0x5ef3, 0x6b16, 0x7063,
00789 0x7c6c, 0x7c6e, 0x883b, 0x89c0, 0x8ea1, 0x91c1, 0x9472, 0x9470,
00790 0x9871, 0x995e, 0x9ad6, 0x9b23, 0x9ecc, 0x7064, 0x77da, 0x8b9a,
00791 0x9477, 0x97c9, 0x9a62, 0x9a65, 0x7e9c, 0x8b9c, 0x8eaa, 0x91c5,
00792 0x947d, 0x947e, 0x947c, 0x9c77, 0x9c78, 0x9ef7, 0x8c54, 0x947f,
00793 0x9e1a, 0x7228, 0x9a6a, 0x9b31, 0x9e1b, 0x9e1e, 0x7c72, 0x30fe,
00794 0x309d, 0x309e, 0x3005, 0x3041, 0x3042, 0x3043, 0x3044, 0x3045,
00795 0x3046, 0x3047, 0x3048, 0x3049, 0x304a, 0x304b, 0x304c, 0x304d,
00796 0x304e, 0x304f, 0x3050, 0x3051, 0x3052, 0x3053, 0x3054, 0x3055,
00797 0x3056, 0x3057, 0x3058, 0x3059, 0x305a, 0x305b, 0x305c, 0x305d,
00798 0x305e, 0x305f, 0x3060, 0x3061, 0x3062, 0x3063, 0x3064, 0x3065,
00799 0x3066, 0x3067, 0x3068, 0x3069, 0x306a, 0x306b, 0x306c, 0x306d,
00800 0x306e, 0x306f, 0x3070, 0x3071, 0x3072, 0x3073, 0x3074, 0x3075,
00801 0x3076, 0x3077, 0x3078, 0x3079, 0x307a, 0x307b, 0x307c, 0x307d,
00802 0x307e, 0x307f, 0x3080, 0x3081, 0x3082, 0x3083, 0x3084, 0x3085,
00803 0x3086, 0x3087, 0x3088, 0x3089, 0x308a, 0x308b, 0x308c, 0x308d,
00804 0x308e, 0x308f, 0x3090, 0x3091, 0x3092, 0x3093, 0x3094, 0x30a2,
00805 0x30a3, 0x30a4, 0x30a5, 0x30a6, 0x30a7,
00806 /* 0xc7 */
00807 0x30a8, 0x30a9, 0x30aa, 0x30ab, 0x30ac, 0x30ad, 0x30ae, 0x30af,
00808 0x30b0, 0x30b1, 0x30b2, 0x30b3, 0x30b4, 0x30b5, 0x30b6, 0x30b7,
00809 0x30b8, 0x30b9, 0x30ba, 0x30bb, 0x30bc, 0x30bd, 0x30be, 0x30bf,
```

```
00810 0x30c0, 0x30c1, 0x30c2, 0x30c3, 0x30c4, 0x30c5, 0x30c6, 0x30c7,
00811 0x30c8, 0x30c9, 0x30ca, 0x30cb, 0x30cc, 0x30cd, 0x30ce, 0x30cf,
00812 0x30d0, 0x30d1, 0x30d2, 0x30d3, 0x30d4, 0x30d5, 0x30d6, 0x30d7,
00813 0x30d8, 0x30d9, 0x30da, 0x30db, 0x30dc, 0x30dd, 0x30de, 0x30df,
00814 0x30e0, 0x30e1, 0x30e2, 0x30e3, 0x30e4, 0x30e5, 0x30e6, 0x30e7,
00815 0x30e8, 0x30e9, 0x30ea, 0x30eb, 0x30ec, 0x30ed, 0x30ee, 0x30ef,
00816 0x30f0, 0x30f1, 0x30f2, 0x30f3, 0x30f4, 0x30f5, 0x30f6, 0x0414,
00817 0x0415, 0x0401, 0x0416, 0x0417, 0x0418, 0x0419, 0x041a, 0x041b,
00818 0x041c, 0x0423, 0x0424, 0x0425, 0x0426, 0x0427, 0x0428, 0x0429,
00819 0x042a, 0x042b, 0x042c, 0x042d, 0x042e, 0x042f, 0x0430, 0x0431,
00820 0x0432, 0x0433, 0x0434, 0x0435, 0x0451, 0x0436, 0x0437, 0x0438,
00821 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e, 0x043f, 0x0440,
00822 0x0441, 0x0442, 0x0443, 0x0444, 0x0445, 0x0446, 0x0447, 0x0448,
00823 0x0449, 0x044a, 0x044b, 0x044c, 0x044d, 0x044e, 0x044f, 0x2460,
00824 0x2461, 0x2462, 0x2463, 0x2464, 0x2465, 0x2466, 0x2467, 0x2468,
00825 0x2469, 0x2474, 0x2475, 0x2476, 0x2477, 0x2478, 0x2479, 0x247a,
00826 0x247b, 0x247c, 0x247d,
00827 };
00828 static const unsigned short big5_2uni_pagec9[7652] = {
00829 /* 0xc9 */
00830 0x4e42, 0x4e5c, 0x51f5, 0x531a, 0x5382, 0x4e07, 0x4e0c, 0x4e47,
00831 0x4e8d, 0x56d7, 0xfa0c, 0x5c6e, 0x5f73, 0x4e0f, 0x5187, 0x4e0e,
00832 0x4e2e, 0x4e93, 0x4ec2, 0x4ec9, 0x4ec8, 0x5198, 0x52fc, 0x536c,
00833 0x53b9, 0x5720, 0x5903, 0x592c, 0x5c10, 0x5dff, 0x65e1, 0x6bb3,
00834 0x6bcc, 0x6c14, 0x723f, 0x4e31, 0x4e3c, 0x4ee8, 0x4edc, 0x4ee9,
00835 0x4ee1, 0x4edd, 0x4eda, 0x520c, 0x531c, 0x534c, 0x5722, 0x5723,
00836 0x5917, 0x592f, 0x5b81, 0x5b84, 0x5c12, 0x5c3b, 0x5c74, 0x5c73,
00837 0x5e04, 0x5e80, 0x5e82, 0x5fc9, 0x6209, 0x6250, 0x6c15, 0x6c36,
00838 0x6c43, 0x6c3f, 0x6c3b, 0x72ae, 0x72b0, 0x738a, 0x79b8, 0x808a,
00839 0x961e, 0x4f0e, 0x4f18, 0x4f10, 0x4f2c, 0x4ef5, 0x4f14, 0x4ef1, 0x4f00,
00840 0x4ef7, 0x4f08, 0x4f1d, 0x4f02, 0x4f05, 0x4f22, 0x4f13, 0x4f04,
00841 0x4ef4, 0x4f12, 0x51b1, 0x5213, 0x5209, 0x5210, 0x52a6, 0x5322,
00842 0x531f, 0x534d, 0x538a, 0x5407, 0x56e1, 0x56df, 0x572e, 0x572a,
00843 0x5734, 0x593c, 0x5980, 0x597c, 0x5985, 0x597b, 0x597e, 0x5977,
00844 0x597f, 0x5b56, 0x5c15, 0x5c25, 0x5c7c, 0x5c7a, 0x5c7b, 0x5c7e,
00845 0x5ddf, 0x5e75, 0x5e84, 0x5f02, 0x5f1a, 0x5f74, 0x5fd5, 0x5fd4,
00846 0x5fcf, 0x625c, 0x625e, 0x6264, 0x6261, 0x6266, 0x6262, 0x6259,
00847 0x6260, 0x625a, 0x6265, 0x65ef, 0x65ee, 0x673e, 0x6739, 0x6738,
00848 0x673b, 0x673a, 0x673f, 0x673c, 0x6733, 0x6c18, 0x6c46, 0x6c52,
00849 0x6c5c, 0x6c4f, 0x6c4a, 0x6c54, 0x6c4b,
00850 /* 0xca */
00851 0x6c4c, 0x7071, 0x725e, 0x72b4, 0x72b5, 0x738e, 0x752a, 0x767f,
00852 0x7a75, 0x7f51, 0x8278, 0x827c, 0x8280, 0x827d, 0x827f, 0x864d,
00853 0x897e, 0x9099, 0x9097, 0x9098, 0x909b, 0x9094, 0x9622, 0x9624,
00854 0x9620, 0x9623, 0x4f56, 0x4f3b, 0x4f62, 0x4f49, 0x4f53, 0x4f64,
00855 0x4f3e, 0x4f67, 0x4f52, 0x4f5f, 0x4f41, 0x4f58, 0x4f2d, 0x4f33,
00856 0x4f3f, 0x4f61, 0x518f, 0x51b9, 0x521c, 0x521e, 0x5221, 0x52ad,
00857 0x52ae, 0x5309, 0x5363, 0x5372, 0x538e, 0x538f, 0x5430, 0x5437,
00858 0x542a, 0x5454, 0x5445, 0x5419, 0x541c, 0x5425, 0x5418, 0x543d,
00859 0x544f, 0x5441, 0x5428, 0x5424, 0x5447, 0x56ee, 0x56e7, 0x56e5,
00860 0x5741, 0x5745, 0x574c, 0x5749, 0x574b, 0x5752, 0x5906, 0x5940,
00861 0x59a6, 0x5998, 0x59a0, 0x5997, 0x598e, 0x59a2, 0x5990, 0x598f,
00862 0x59a7, 0x59a1, 0x5b8e, 0x5b92, 0x5c28, 0x5c2a, 0x5c8d, 0x5c8f,
00863 0x5c88, 0x5c8b, 0x5c89, 0x5c92, 0x5c8a, 0x5c86, 0x5c93, 0x5c95,
00864 0x5de0, 0x5e0a, 0x5e0e, 0x5e8b, 0x5e89, 0x5e8c, 0x5e88, 0x5e8d,
00865 0x5f05, 0x5f1d, 0x5f78, 0x5f76, 0x5fd2, 0x5fd1, 0x5fd0, 0x5fed,
00866 0x5fe8, 0x5fee, 0x5ff3, 0x5ff1, 0x5ff4, 0x5fe3, 0x5ffa, 0x5fef,
00867 0x5ff7, 0x5ffb, 0x6000, 0x5ff4, 0x623a, 0x6283, 0x628c, 0x628e,
00868 0x628f, 0x6294, 0x6287, 0x6271, 0x627b, 0x627a, 0x6270, 0x6281,
00869 0x6288, 0x6277, 0x627d, 0x6272, 0x6274, 0x6537, 0x65f0, 0x65f4,
00870 0x65f3, 0x65f2, 0x65f5, 0x6745, 0x6747,
00871 /* 0xcb */
00872 0x6759, 0x6755, 0x674c, 0x6748, 0x675d, 0x674d, 0x675a, 0x674b,
00873 0x6bd0, 0x6c19, 0x6c1a, 0x6c78, 0x6c67, 0x6c6b, 0x6c84, 0x6c8b,
00874 0x6c8f, 0x6c71, 0x6c6f, 0x6c69, 0x6c9a, 0x6c6d, 0x6c87, 0x6c95,
00875 0x6c9c, 0x6c66, 0x6c73, 0x6c65, 0x6c7b, 0x6c8e, 0x7074, 0x707a,
00876 0x7263, 0x72b6, 0x72bd, 0x72c3, 0x72c6, 0x72c1, 0x72ba, 0x72c5,
00877 0x7395, 0x7397, 0x7393, 0x7394, 0x7392, 0x753a, 0x7539, 0x7594,
00878 0x7595, 0x7681, 0x793d, 0x8034, 0x8095, 0x8099, 0x8090, 0x8092,
00879 0x809c, 0x8290, 0x828f, 0x8285, 0x828e, 0x8291, 0x8293, 0x828a,
00880 0x8283, 0x8284, 0x8c78, 0x8fc9, 0x8fbf, 0x909f, 0x90a1, 0x90a5,
00881 0x909e, 0x90a7, 0x90a0, 0x9630, 0x9628, 0x962f, 0x962d, 0x4e33,
00882 0x4f98, 0x4f7c, 0x4f85, 0x4f7d, 0x4f80, 0x4f87, 0x4f76, 0x4f74,
00883 0x4f89, 0x4f84, 0x4f77, 0x4f4c, 0x4f97, 0x4f6a, 0x4f9a, 0x4f79,
00884 0x4f81, 0x4f78, 0x4f90, 0x4f9c, 0x4f94, 0x4f9e, 0x4f92, 0x4f82,
00885 0x4f95, 0x4f6b, 0x4f6e, 0x519e, 0x51bc, 0x51be, 0x5233, 0x5232,
00886 0x5233, 0x5246, 0x5231, 0x52bc, 0x530a, 0x530b, 0x533c, 0x5392,
00887 0x5394, 0x5487, 0x547f, 0x5481, 0x5491, 0x5482, 0x5488, 0x546b,
00888 0x547a, 0x547e, 0x5465, 0x546c, 0x5474, 0x5466, 0x548d, 0x546f,
00889 0x5461, 0x5460, 0x5498, 0x5463, 0x5467, 0x5464, 0x56f7, 0x56f9,
00890 0x576f, 0x5772, 0x576d, 0x576b, 0x5771, 0x5770, 0x5776, 0x5780,
00891 0x5775, 0x577b, 0x5773, 0x5774, 0x5762,
00892 /* 0xcc */
00893 0x5768, 0x577d, 0x590c, 0x5945, 0x59b5, 0x59ba, 0x59cf, 0x59ce,
00894 0x59b2, 0x59cc, 0x59c1, 0x59b6, 0x59bc, 0x59c3, 0x59d6, 0x59b1,
00895 0x59bd, 0x59c0, 0x59c8, 0x59b4, 0x59c7, 0x5b62, 0x5b65, 0x5b93,
00896 0x5b95, 0x5c44, 0x5c47, 0x5cae, 0x5ca4, 0x5ca0, 0x5cb5, 0x5caf,
```

```
00897 0x5ca8, 0x5cac, 0x5c9f, 0x5ca3, 0x5cad, 0x5ca2, 0x5caa, 0x5ca7,
00898 0x5c9d, 0x5ca5, 0x5cb6, 0x5cb0, 0x5ca6, 0x5e17, 0x5e14, 0x5e19,
00899 0x5f28, 0x5f22, 0x5f23, 0x5f24, 0x5f54, 0x5f82, 0x5f7e, 0x5f7d,
00900 0x5fde, 0x5fe5, 0x602d, 0x6026, 0x6019, 0x6032, 0x600b, 0x6034,
00901 0x600a, 0x6017, 0x6033, 0x601a, 0x601e, 0x602c, 0x6022, 0x600d,
00902 0x6010, 0x602e, 0x6013, 0x6011, 0x600c, 0x6009, 0x601c, 0x6214,
00903 0x623d, 0x62ad, 0x62b4, 0x62d1, 0x62be, 0x62aa, 0x62b6, 0x62ca,
00904 0x62ae, 0x62b3, 0x62af, 0x62bb, 0x62a9, 0x62b0, 0x62b8, 0x653d,
00905 0x65a8, 0x65bb, 0x6609, 0x65fc, 0x6604, 0x6612, 0x6608, 0x65fb,
00906 0x6603, 0x660b, 0x660d, 0x6605, 0x65fd, 0x6611, 0x6610, 0x66f6,
00907 0x670a, 0x6785, 0x676c, 0x678e, 0x6792, 0x6776, 0x677b, 0x6798,
00908 0x6786, 0x6784, 0x6774, 0x678d, 0x678c, 0x677a, 0x679f, 0x6791,
00909 0x6799, 0x6783, 0x677d, 0x6781, 0x6778, 0x6779, 0x6794, 0x6b25,
00910 0x6b80, 0x6b7e, 0x6bde, 0x6c1d, 0x6c93, 0x6cec, 0x6ceb, 0x6cee,
00911 0x6cd9, 0x6cb6, 0x6cd4, 0x6cad, 0x6ce7, 0x6cb7, 0x6cd0, 0x6cc2,
00912 0x6cba, 0x6cc3, 0x6cc6, 0x6ced, 0x6cf2,
00913 /* 0xcd */
00914 0x6cd2, 0x6cdd, 0x6cb4, 0x6c8a, 0x6c9d, 0x6c80, 0x6cde, 0x6cc0,
00915 0x6d30, 0x6ccd, 0x6cc7, 0x6cb0, 0x6cf9, 0x6ccf, 0x6ce9, 0x6cd1,
00916 0x7094, 0x7098, 0x7085, 0x7093, 0x7086, 0x7084, 0x7091, 0x7096,
00917 0x7082, 0x709a, 0x7083, 0x726a, 0x72d6, 0x72cb, 0x72d8, 0x72c9,
00918 0x72dc, 0x72d2, 0x72d4, 0x72da, 0x72cc, 0x72d1, 0x73a4, 0x73a1,
00919 0x73ad, 0x73ae, 0x73a2, 0x73a0, 0x73ac, 0x739d, 0x74dd, 0x74e8,
00920 0x753f, 0x7540, 0x753e, 0x758c, 0x7598, 0x76af, 0x76f3, 0x76f1,
00921 0x76f0, 0x76f5, 0x77f8, 0x77fc, 0x77f9, 0x77fb, 0x77fa, 0x77f7,
00922 0x7942, 0x793f, 0x79c5, 0x7a78, 0x7a7b, 0x7afb, 0x7c75, 0x7cfd,
00923 0x8035, 0x808f, 0x80ae, 0x80a3, 0x80b8, 0x80b5, 0x80ad, 0x8220,
00924 0x82a0, 0x82c0, 0x82ab, 0x829a, 0x8298, 0x829b, 0x82b5, 0x82a7,
00925 0x82ae, 0x82bc, 0x829e, 0x82ba, 0x82b4, 0x82a8, 0x82a1, 0x82a9,
00926 0x82c2, 0x82a4, 0x82c3, 0x82b6, 0x82a2, 0x8670, 0x866f, 0x866d,
00927 0x866e, 0x8c56, 0x8fd2, 0x8fcb, 0x8fd3, 0x8fcd, 0x8fd6, 0x8fd5,
00928 0x8fd7, 0x90b2, 0x90b4, 0x90af, 0x90b3, 0x90b0, 0x9639, 0x963d,
00929 0x963c, 0x963a, 0x9643, 0x9643, 0x4fc5, 0x4fd3, 0x4fb2, 0x4fc9,
00930 0x4fcb, 0x4fc1, 0x4fd4, 0x4fdc, 0x4fd9, 0x4fbb, 0x4fb3, 0x4fdb,
00931 0x4fc7, 0x4fd6, 0x4fba, 0x4fc0, 0x4fb9, 0x4fec, 0x5244, 0x5249,
00932 0x52c0, 0x52c2, 0x533d, 0x537c, 0x5396, 0x5399, 0x5398,
00933 0x54ba, 0x54a1, 0x54ad, 0x54a5, 0x54cf,
00934 /* 0xce */
00935 0x54c3, 0x830d, 0x54b7, 0x54ae, 0x54d6, 0x54b6, 0x54c5, 0x54c6,
00936 0x54a0, 0x5470, 0x54bc, 0x54a2, 0x54be, 0x5472, 0x54de, 0x54b0,
00937 0x57b5, 0x579e, 0x579f, 0x57a4, 0x578c, 0x5797, 0x579d, 0x579b,
00938 0x5794, 0x5798, 0x578f, 0x5799, 0x57a5, 0x579a, 0x5795, 0x58f4,
00939 0x590d, 0x5953, 0x59e1, 0x59de, 0x59ee, 0x5a00, 0x59f1, 0x59dd,
00940 0x59fa, 0x59fd, 0x59fc, 0x59f6, 0x59e4, 0x59f2, 0x59f7, 0x59db,
00941 0x59e9, 0x59f3, 0x59f5, 0x59e0, 0x59fe, 0x59f4, 0x59ed, 0x5ba8,
00942 0x5c4c, 0x5cd0, 0x5cd8, 0x5ccc, 0x5cd7, 0x5ccb, 0x5cdb, 0x5cde,
00943 0x5cda, 0x5cc9, 0x5cc7, 0x5cca, 0x5cd6, 0x5cd3, 0x5cd4, 0x5ccf,
00944 0x5cc8, 0x5cc6, 0x5cce, 0x5cdf, 0x5cf8, 0x5df9, 0x5e21, 0x5e22,
00945 0x5e23, 0x5e20, 0x5e24, 0x5eb0, 0x5ea4, 0x5ea2, 0x5e9b, 0x5ea3,
00946 0x5ea5, 0x5f07, 0x5f2e, 0x5f56, 0x5f86, 0x6037, 0x6039, 0x6054,
00947 0x6072, 0x605e, 0x6045, 0x6053, 0x6047, 0x6049, 0x605b, 0x604c,
00948 0x6040, 0x6042, 0x605f, 0x6024, 0x6044, 0x6058, 0x6066, 0x606e,
00949 0x6242, 0x6243, 0x62cf, 0x630d, 0x630b, 0x62f5, 0x630e, 0x6303,
00950 0x62eb, 0x62f9, 0x630f, 0x630c, 0x62f8, 0x62f6, 0x6300, 0x6313,
00951 0x6314, 0x62fa, 0x6315, 0x62fb, 0x62f0, 0x6541, 0x6543, 0x65aa,
00952 0x65bf, 0x6636, 0x6621, 0x6632, 0x6635, 0x661c, 0x6626, 0x6622,
00953 0x6633, 0x662b, 0x663a, 0x661d, 0x6634, 0x6639, 0x662e, 0x670f,
00954 0x6710, 0x67c1, 0x67f2, 0x67c8, 0x67ba,
00955 /* 0xcf */
00956 0x67dc, 0x67bb, 0x67f8, 0x67d8, 0x67c0, 0x67b7, 0x67c5, 0x67eb,
00957 0x67e4, 0x67df, 0x67b5, 0x67cd, 0x67b3, 0x67f7, 0x67f6, 0x67ee,
00958 0x67e3, 0x67c2, 0x67b9, 0x67ce, 0x67e7, 0x67f0, 0x67b2, 0x67fc,
00959 0x67c6, 0x67ed, 0x67cc, 0x67ae, 0x67e6, 0x67db, 0x67fa, 0x67c9,
00960 0x67ca, 0x67c3, 0x67ea, 0x67cb, 0x6b28, 0x6b82, 0x6b84, 0x6bb6,
00961 0x6bd6, 0x6bd8, 0x6be0, 0x6c20, 0x6c21, 0x6d28, 0x6d34, 0x6d2d,
00962 0x6d1f, 0x6d3c, 0x6d3f, 0x6d12, 0x6d0a, 0x6cda, 0x6d33, 0x6d04,
00963 0x6d19, 0x6d3a, 0x6d1a, 0x6d11, 0x6d00, 0x6d1d, 0x6d42, 0x6d01,
00964 0x6d18, 0x6d37, 0x6d03, 0x6d0f, 0x6d40, 0x6d07, 0x6d20, 0x6d2c,
00965 0x6d08, 0x6d22, 0x6d09, 0x6d10, 0x70b7, 0x709f, 0x70be, 0x70b1,
00966 0x70b0, 0x70a1, 0x70b4, 0x70b5, 0x70a9, 0x7241, 0x7249, 0x724a,
00967 0x726c, 0x7270, 0x7273, 0x7273, 0x726e, 0x72ca, 0x72e8, 0x72eb,
00968 0x72df, 0x72ea, 0x72e6, 0x72e3, 0x7385, 0x73cc, 0x73c2, 0x73c8,
00969 0x73c5, 0x73b9, 0x73b6, 0x73b5, 0x73b4, 0x73eb, 0x73bf, 0x73c7,
00970 0x73be, 0x73c3, 0x73c6, 0x73b8, 0x73cb, 0x74ec, 0x74ee, 0x752e,
00971 0x7547, 0x7548, 0x75a7, 0x75aa, 0x7679, 0x76c4, 0x7708, 0x7703,
00972 0x7704, 0x7705, 0x770a, 0x76f7, 0x76fb, 0x76fa, 0x77e7, 0x77e8,
00973 0x7806, 0x7811, 0x7812, 0x7805, 0x7810, 0x780f, 0x780e, 0x7809,
00974 0x7803, 0x7813, 0x794a, 0x794c, 0x794b, 0x7945, 0x7944, 0x79d5,
00975 0x79cd, 0x79cf, 0x79d6, 0x79ce, 0x7a80,
00976 /* 0xd0 */
00977 0x7a7e, 0x7ad1, 0x7b00, 0x7b01, 0x7c7a, 0x7c78, 0x7c79, 0x7c7f,
00978 0x7c80, 0x7c81, 0x7d03, 0x7d08, 0x7d01, 0x7f58, 0x7f91, 0x7f8d,
00979 0x7fbe, 0x8007, 0x800e, 0x800f, 0x8014, 0x8037, 0x80d8, 0x80c7,
00980 0x80e0, 0x80d1, 0x80c8, 0x80c2, 0x80d0, 0x80c5, 0x80e3, 0x80d9,
00981 0x80dc, 0x80ca, 0x80d5, 0x80c9, 0x80cf, 0x80d7, 0x80e6, 0x80cd,
00982 0x81ff, 0x8221, 0x8294, 0x82d9, 0x82fe, 0x82f9, 0x8307, 0x82e8,
00983 0x8300, 0x82d5, 0x833a, 0x82eb, 0x82d6, 0x82f4, 0x82ec, 0x82e1,
```

```

00984 0x82f2, 0x82f5, 0x830c, 0x82fb, 0x82f6, 0x82f0, 0x82ea, 0x82e4,
00985 0x82e0, 0x82fa, 0x82f3, 0x82ed, 0x8677, 0x8674, 0x867c, 0x8673,
00986 0x8841, 0x884e, 0x8867, 0x886a, 0x8869, 0x88d3, 0x8a04, 0x8a07,
00987 0x8d72, 0x8fe3, 0x8fe1, 0x8fee, 0x8fe0, 0x90f1, 0x90bd, 0x90bf,
00988 0x90d5, 0x90c5, 0x90be, 0x90c7, 0x90cb, 0x90c8, 0x91d4, 0x91d3,
00989 0x9654, 0x964f, 0x9651, 0x9653, 0x964a, 0x964e, 0x501e, 0x5005,
00990 0x5007, 0x5013, 0x5022, 0x5030, 0x501b, 0x4ff5, 0x4ff4, 0x5033,
00991 0x5037, 0x502c, 0x4ff6, 0x4ff7, 0x5017, 0x501c, 0x5020, 0x5027,
00992 0x5035, 0x502f, 0x5031, 0x500e, 0x515a, 0x5194, 0x5193, 0x51ca,
00993 0x51c4, 0x51c5, 0x51c8, 0x51ce, 0x5261, 0x525a, 0x5252, 0x525e,
00994 0x525f, 0x5255, 0x5262, 0x52cd, 0x530e, 0x539e, 0x5526, 0x54e2,
00995 0x5517, 0x5512, 0x54e7, 0x54f3, 0x54e4, 0x551a, 0x54ff, 0x5504,
00996 0x5508, 0x54eb, 0x5511, 0x5505, 0x54f1,
00997 /* 0xd1 */
00998 0x550a, 0x54fb, 0x54f7, 0x54f8, 0x54e0, 0x550e, 0x5503, 0x550b,
00999 0x5701, 0x5702, 0x57cc, 0x5832, 0x57d5, 0x57d2, 0x57ba, 0x57c6,
01000 0x57bd, 0x57bc, 0x57b8, 0x57b6, 0x57bf, 0x57c7, 0x57d0, 0x57b9,
01001 0x57c1, 0x590e, 0x594a, 0x5a19, 0x5a16, 0x5a2d, 0x5a2e, 0x5a15,
01002 0x5a0f, 0x5a17, 0x5a0a, 0x5a1e, 0x5a33, 0x5b6c, 0x5ba7, 0x5bad,
01003 0x5bac, 0x5c03, 0x5c56, 0x5c54, 0x5cec, 0x5cff, 0x5cee, 0x5cf1,
01004 0x5cf7, 0x5d00, 0x5cf9, 0x5e29, 0x5e28, 0x5ea8, 0x5eae, 0x5eaa,
01005 0x5eac, 0x5f33, 0x5f30, 0x5f67, 0x605d, 0x605a, 0x6067, 0x6041,
01006 0x60a2, 0x6088, 0x6080, 0x6092, 0x6081, 0x609d, 0x6083, 0x6095,
01007 0x609b, 0x6097, 0x6087, 0x609c, 0x608e, 0x6219, 0x6246, 0x62f2,
01008 0x6310, 0x6356, 0x632c, 0x6344, 0x6345, 0x6336, 0x6343, 0x63e4,
01009 0x6339, 0x634b, 0x634a, 0x633c, 0x6329, 0x6341, 0x6334, 0x6358,
01010 0x6354, 0x6359, 0x632d, 0x6347, 0x6333, 0x635a, 0x6351, 0x6338,
01011 0x6357, 0x6340, 0x6348, 0x654a, 0x6546, 0x65c6, 0x65c3, 0x65c4,
01012 0x65c2, 0x664a, 0x665f, 0x6647, 0x6651, 0x6712, 0x6713, 0x681f,
01013 0x681a, 0x6849, 0x6832, 0x6833, 0x683b, 0x684b, 0x684f, 0x6816,
01014 0x6831, 0x681c, 0x6835, 0x682b, 0x682d, 0x682f, 0x684e, 0x6844,
01015 0x6834, 0x681d, 0x6812, 0x6814, 0x6826, 0x6828, 0x682e, 0x684d,
01016 0x683a, 0x6825, 0x6820, 0x6b2c, 0x6b2f, 0x6b2d, 0x6b31, 0x6b34,
01017 0x6b6d, 0x8082, 0x6b88, 0x6be6, 0x6be4,
01018 /* 0xd2 */
01019 0x6be8, 0x6be3, 0x6be2, 0x6be7, 0x6c25, 0x6d7a, 0x6d63, 0x6d64,
01020 0x6d76, 0x6d0d, 0x6d61, 0x6d92, 0x6d58, 0x6d62, 0x6d6d, 0x6d6f,
01021 0x6d91, 0x6d8d, 0x6def, 0x6d7f, 0x6d86, 0x6d5e, 0x6d67, 0x6d60,
01022 0x6d97, 0x6d70, 0x6d7c, 0x6d5f, 0x6d82, 0x6d98, 0x6d2f, 0x6d68,
01023 0x6d8b, 0x6d7e, 0x6d80, 0x6d84, 0x6d16, 0x6d83, 0x6d7b, 0x6d7d,
01024 0x6d75, 0x6d90, 0x70dc, 0x70d3, 0x70d1, 0x70dd, 0x70cb, 0x7f39,
01025 0x70e2, 0x70d7, 0x70d2, 0x70de, 0x70e0, 0x70d4, 0x70cd, 0x70c5,
01026 0x70c6, 0x70c7, 0x70da, 0x70ce, 0x70e1, 0x7242, 0x7278, 0x7277,
01027 0x7276, 0x7300, 0x72fa, 0x72f4, 0x72fe, 0x72f6, 0x72f3, 0x72fb,
01028 0x7301, 0x73d3, 0x73d9, 0x73e5, 0x73d6, 0x73bc, 0x73e7, 0x73e3,
01029 0x73e9, 0x73dc, 0x73d2, 0x73db, 0x73d4, 0x73dd, 0x73da, 0x73d7,
01030 0x73d8, 0x73e8, 0x74de, 0x74df, 0x74f4, 0x74f5, 0x7521, 0x755b,
01031 0x755f, 0x75b0, 0x75c1, 0x75bb, 0x75c4, 0x75c0, 0x75bf, 0x75b6,
01032 0x75ba, 0x768a, 0x76c9, 0x771d, 0x771b, 0x7710, 0x7713, 0x7712,
01033 0x7723, 0x7711, 0x7715, 0x7719, 0x771a, 0x7722, 0x7727, 0x7823,
01034 0x782c, 0x7822, 0x7835, 0x782f, 0x7828, 0x782b, 0x7821,
01035 0x7829, 0x7833, 0x782a, 0x7831, 0x7954, 0x795b, 0x794f, 0x795c,
01036 0x7953, 0x7952, 0x7951, 0x79eb, 0x79ec, 0x79e0, 0x79ee, 0x79ed,
01037 0x79ea, 0x79dc, 0x79de, 0x7a86, 0x7a89, 0x7a85, 0x7a8b,
01038 0x7a8c, 0x7a8a, 0x7a87, 0x7ad8, 0x7b10,
01039 /* 0xd3 */
01040 0x7b04, 0x7b13, 0x7b05, 0x7b0f, 0x7b08, 0x7b0a, 0x7b0e, 0x7b09,
01041 0x7b12, 0x7c84, 0x7c91, 0x7c8a, 0x7c8c, 0x7c88, 0x7c8d, 0x7c85,
01042 0x7d1e, 0x7d1d, 0x7d11, 0x7d0e, 0x7d18, 0x7d16, 0x7d13, 0x7d1f,
01043 0x7d12, 0x7d0f, 0x7d0c, 0x7f5c, 0x7f61, 0x7f5e, 0x7f60, 0x7f5d,
01044 0x7f5b, 0x7f96, 0x7f92, 0x7fc3, 0x7fc2, 0x7fc0, 0x8016, 0x803e,
01045 0x8039, 0x80fa, 0x80f2, 0x80f9, 0x80f5, 0x8101, 0x80fb, 0x8100,
01046 0x8201, 0x822f, 0x8225, 0x8333, 0x832d, 0x8344, 0x8319, 0x8351,
01047 0x8325, 0x8356, 0x833f, 0x8341, 0x8326, 0x831c, 0x8322, 0x8342,
01048 0x834e, 0x831b, 0x832a, 0x8308, 0x833c, 0x834d, 0x8316, 0x8324,
01049 0x8320, 0x8337, 0x832f, 0x8329, 0x8347, 0x8345, 0x834c, 0x8353,
01050 0x831e, 0x832c, 0x834b, 0x8327, 0x8348, 0x8653, 0x8652, 0x86a2,
01051 0x86a8, 0x8696, 0x868d, 0x8691, 0x869e, 0x8687, 0x8697, 0x8686,
01052 0x868b, 0x869a, 0x8685, 0x86a5, 0x8699, 0x86a1, 0x86a7, 0x8695,
01053 0x8698, 0x868e, 0x869d, 0x8690, 0x8694, 0x8843, 0x8844, 0x886d,
01054 0x8875, 0x8876, 0x8872, 0x8880, 0x8871, 0x887f, 0x886f, 0x8883,
01055 0x887e, 0x8874, 0x887c, 0x8a12, 0x8c47, 0x8c57, 0x8c7b, 0x8ca4,
01056 0x8ca3, 0x8d76, 0x8d78, 0x8db5, 0x8db7, 0x8db6, 0x8ed1, 0x8ed3,
01057 0x8ffe, 0x8ff5, 0x9002, 0x8fff, 0x9004, 0x8ffc, 0x8ffc, 0x8ffc,
01058 0x90d6, 0x90e0, 0x90d9, 0x90da, 0x90e3, 0x90df, 0x90e5, 0x90d8,
01059 0x90db, 0x90d7, 0x90dc, 0x90e4, 0x9150,
01060 /* 0xd4 */
01061 0x914e, 0x914f, 0x91d5, 0x91e2, 0x91da, 0x965c, 0x965f, 0x96bc,
01062 0x98e3, 0x9adef, 0x9b2f, 0x4e7f, 0x5070, 0x506a, 0x5061, 0x505e,
01063 0x5060, 0x5053, 0x504b, 0x504d, 0x5072, 0x5048, 0x504d, 0x5041,
01064 0x505b, 0x504a, 0x5062, 0x5015, 0x5045, 0x505f, 0x5069, 0x506b,
01065 0x5063, 0x5064, 0x5046, 0x5040, 0x506e, 0x5073, 0x5057, 0x5051,
01066 0x51d0, 0x526b, 0x526d, 0x526c, 0x526e, 0x52d6, 0x52d3, 0x532d,
01067 0x539c, 0x5575, 0x5576, 0x553c, 0x554d, 0x5550, 0x5534, 0x552a,
01068 0x5551, 0x5562, 0x5536, 0x5535, 0x5530, 0x5552, 0x5545, 0x550c,
01069 0x5532, 0x5565, 0x554e, 0x5539, 0x5548, 0x552d, 0x553b, 0x5540,
01070 0x554b, 0x570a, 0x5707, 0x57fb, 0x5814, 0x57e2, 0x57f6, 0x57dc,

```

```
01071 0x57f4, 0x5800, 0x57ed, 0x57fd, 0x5808, 0x57f8, 0x580b, 0x57f3,
01072 0x57cf, 0x5807, 0x57ee, 0x57e3, 0x57f2, 0x57e5, 0x57ec, 0x57e1,
01073 0x580e, 0x57fc, 0x5810, 0x57e7, 0x5801, 0x580c, 0x57f1, 0x57e9,
01074 0x57f0, 0x580d, 0x5804, 0x595c, 0x5a60, 0x5a58, 0x5a55, 0x5a67,
01075 0x5a5e, 0x5a38, 0x5a35, 0x5a6d, 0x5a50, 0x5a5f, 0x5a65, 0x5a6c,
01076 0x5a53, 0x5a64, 0x5a57, 0x5a43, 0x5a5d, 0x5a52, 0x5a44, 0x5a5b,
01077 0x5a48, 0x5a8e, 0x5a3e, 0x5a4d, 0x5a39, 0x5a4c, 0x5a70, 0x5a69,
01078 0x5a47, 0x5a51, 0x5a56, 0x5a42, 0x5a5c, 0x5b72, 0x5b6e, 0x5bc1,
01079 0x5bc0, 0x5c59, 0x5d1e, 0x5d0b, 0x5d1d, 0x5d1a, 0x5d20, 0x5d0c,
01080 0x5d28, 0x5d0d, 0x5d26, 0x5d25, 0x5d0f,
01081 /* 0xd5 */
01082 0x5d30, 0x5d12, 0x5d23, 0x5d1f, 0x5d2e, 0x5e3e, 0x5e34, 0x5eb1,
01083 0x5eb4, 0x5eb9, 0x5eb2, 0x5eb3, 0x5f36, 0x5f38, 0x5f9b, 0x5f96,
01084 0x5f9f, 0x608a, 0x6090, 0x6086, 0x60be, 0x60b0, 0x60ba, 0x60d3,
01085 0x60d4, 0x60cf, 0x60e4, 0x60d9, 0x60dd, 0x60c8, 0x60b1, 0x60db,
01086 0x60b7, 0x60ca, 0x60bf, 0x60c3, 0x60cd, 0x60c0, 0x6332, 0x6365,
01087 0x638a, 0x6382, 0x637d, 0x63bd, 0x639e, 0x63ad, 0x639d, 0x6397,
01088 0x63ab, 0x638e, 0x636f, 0x6387, 0x6390, 0x636e, 0x63af, 0x6375,
01089 0x639c, 0x636d, 0x63ae, 0x637c, 0x63a4, 0x633b, 0x639f, 0x6378,
01090 0x6385, 0x6381, 0x6391, 0x638d, 0x6370, 0x6553, 0x65cd, 0x6665,
01091 0x6661, 0x665b, 0x6659, 0x665c, 0x6662, 0x6718, 0x6879, 0x6887,
01092 0x6890, 0x689c, 0x686d, 0x686e, 0x68ae, 0x68ab, 0x6956, 0x686f,
01093 0x68a3, 0x68ac, 0x68a9, 0x6875, 0x6874, 0x68b2, 0x688f, 0x6877,
01094 0x6892, 0x687c, 0x686b, 0x6872, 0x68aa, 0x6880, 0x6871, 0x687e,
01095 0x689b, 0x6896, 0x688b, 0x68a0, 0x6889, 0x68a4, 0x6878, 0x687b,
01096 0x6891, 0x688c, 0x688a, 0x687d, 0x6b36, 0x6b33, 0x6b37, 0x6b38,
01097 0x6b91, 0x6b8f, 0x6b8d, 0x6b8e, 0x6b8c, 0x6c2a, 0x6dc0, 0x6dab,
01098 0x6db4, 0x6db3, 0x6e74, 0x6dac, 0x6de9, 0x6de2, 0x6db7, 0x6df6,
01099 0x6dd4, 0x6e00, 0x6dc8, 0x6de0, 0x6ddf, 0x6dd6, 0x6dbe, 0x6de5,
01100 0x6ddc, 0x6ddd, 0x6ddb, 0x6ddf4, 0x6dca, 0x6dbd, 0x6ded, 0x6df0,
01101 0x6dba, 0x6dd5, 0x6dc2, 0x6dcf, 0x6dc9,
01102 /* 0xd6 */
01103 0x6dd0, 0x6df2, 0x6dd3, 0x6dfd, 0x6dd7, 0x6dcd, 0x6de3, 0x6dbb,
01104 0x70fa, 0x710d, 0x70f7, 0x7117, 0x70f4, 0x710c, 0x70f0, 0x7104,
01105 0x70f3, 0x7110, 0x70fc, 0x70ff, 0x7106, 0x7113, 0x7100, 0x70f8,
01106 0x70f6, 0x710b, 0x7102, 0x710e, 0x727e, 0x727b, 0x727c, 0x727f,
01107 0x731d, 0x7317, 0x7307, 0x7311, 0x7318, 0x730a, 0x7308, 0x72ff,
01108 0x730f, 0x731e, 0x7388, 0x73f6, 0x73f8, 0x73f5, 0x7404, 0x7401,
01109 0x73fd, 0x7407, 0x7400, 0x73fa, 0x73fc, 0x73ff, 0x740c, 0x740b,
01110 0x73f4, 0x7408, 0x7564, 0x7563, 0x75ce, 0x75d2, 0x75cf, 0x75cb,
01111 0x75cc, 0x75d1, 0x75d0, 0x768f, 0x7689, 0x76d3, 0x7739, 0x772f,
01112 0x772d, 0x7731, 0x7732, 0x7734, 0x7733, 0x773d, 0x7725, 0x773b,
01113 0x7735, 0x7848, 0x7852, 0x7849, 0x784d, 0x784a, 0x784c, 0x7826,
01114 0x7845, 0x7850, 0x7964, 0x7967, 0x7969, 0x796a, 0x7963, 0x796b,
01115 0x7961, 0x79b0, 0x79fa, 0x79f8, 0x79f6, 0x79f7, 0x7a8f, 0x7a94,
01116 0x7a90, 0x7b35, 0x7b47, 0x7b34, 0x7b25, 0x7b30, 0x7b22, 0x7b24,
01117 0x7b33, 0x7b18, 0x7b2a, 0x7b1d, 0x7b31, 0x7b2b, 0x7b2d, 0x7b2f,
01118 0x7b32, 0x7b38, 0x7b1a, 0x7b23, 0x7c94, 0x7c98, 0x7c96, 0x7ca3,
01119 0x7d35, 0x7d3d, 0x7d38, 0x7d36, 0x7d3a, 0x7d45, 0x7d2c, 0x7d29,
01120 0x7d41, 0x7d47, 0x7d3e, 0x7d3f, 0x7d4a, 0x7d3b, 0x7d28, 0x7f63,
01121 0x7f95, 0x7f9c, 0x7f9d, 0x7f9b, 0x7fca, 0x7fcb, 0x7fcd, 0x7fd0,
01122 0x7fd1, 0x7fc7, 0x7fcf, 0x7fc9, 0x801f,
01123 /* 0xd7 */
01124 0x801e, 0x801b, 0x8047, 0x8043, 0x8048, 0x8118, 0x8125, 0x8119,
01125 0x811b, 0x812d, 0x811f, 0x812c, 0x811e, 0x8121, 0x8115, 0x8127,
01126 0x811d, 0x8122, 0x8211, 0x8238, 0x8233, 0x823a, 0x8234, 0x8232,
01127 0x8274, 0x8390, 0x83a3, 0x83a8, 0x838d, 0x837a, 0x8373, 0x83a4,
01128 0x8374, 0x838f, 0x8381, 0x8395, 0x8399, 0x8375, 0x8394, 0x83a9,
01129 0x837d, 0x8383, 0x838c, 0x839d, 0x839b, 0x83aa, 0x838b, 0x837e,
01130 0x83a5, 0x83af, 0x8388, 0x8397, 0x83b0, 0x837f, 0x83a6, 0x8387,
01131 0x83ae, 0x8376, 0x839a, 0x8659, 0x8656, 0x86bf, 0x86b7, 0x86c2,
01132 0x86c1, 0x86c5, 0x86ba, 0x86b0, 0x86c8, 0x86b9, 0x86b3, 0x86b8,
01133 0x86cc, 0x86b4, 0x86bb, 0x86bc, 0x86c3, 0x86bd, 0x86be, 0x8852,
01134 0x8889, 0x8895, 0x88a8, 0x88a2, 0x88aa, 0x889a, 0x8891, 0x88a1,
01135 0x889f, 0x8898, 0x88a7, 0x8899, 0x889b, 0x8897, 0x88a4, 0x88ac,
01136 0x888c, 0x888e, 0x888e, 0x8982, 0x89d6, 0x89d9, 0x89d5, 0x8a30,
01137 0x8a27, 0x8a2c, 0x8a1e, 0x8c39, 0x8c3b, 0x8c5c, 0x8c5d, 0x8c7d,
01138 0x8ca5, 0x8d7d, 0x8d7b, 0x8d79, 0x8dbc, 0x8dc2, 0x8db9, 0x8dbf,
01139 0x8dc1, 0x8ed8, 0x8ede, 0x8edd, 0x8edc, 0x8ed7, 0x8ee0, 0x8ee1,
01140 0x9024, 0x900b, 0x9011, 0x901c, 0x900c, 0x9021, 0x90ef, 0x90ea,
01141 0x90f0, 0x90f4, 0x90f2, 0x90f3, 0x90d4, 0x90eb, 0x90ec, 0x90e9,
01142 0x9156, 0x9158, 0x915a, 0x9153, 0x9155, 0x91ec, 0x91f4, 0x91f1,
01143 0x91f3, 0x91f8, 0x91e4, 0x91f9, 0x91ea,
01144 /* 0xd8 */
01145 0x91eb, 0x91f7, 0x91e8, 0x91ee, 0x957a, 0x9586, 0x9588, 0x967c,
01146 0x966d, 0x966b, 0x9671, 0x966f, 0x96bf, 0x976a, 0x9804, 0x98e5,
01147 0x9997, 0x509b, 0x5095, 0x5094, 0x509e, 0x508b, 0x50a3, 0x5083,
01148 0x508c, 0x508e, 0x509d, 0x5068, 0x509c, 0x5092, 0x5082, 0x5087,
01149 0x515f, 0x51d4, 0x5312, 0x5311, 0x53a4, 0x53a7, 0x5591, 0x55a8,
01150 0x55a5, 0x55ad, 0x5577, 0x5645, 0x55a2, 0x5593, 0x5588, 0x558f,
01151 0x55b5, 0x5581, 0x55a3, 0x5592, 0x55a4, 0x557d, 0x558c, 0x55a6,
01152 0x557f, 0x5595, 0x55a1, 0x558e, 0x570c, 0x5829, 0x5837, 0x5819,
01153 0x581e, 0x5827, 0x5823, 0x5828, 0x57f5, 0x5848, 0x5825, 0x581c,
01154 0x581b, 0x5833, 0x583f, 0x5836, 0x582e, 0x5839, 0x5838, 0x582d,
01155 0x582c, 0x583b, 0x5961, 0x5aaf, 0x5a94, 0x5a9f, 0x5a7a, 0x5aa2,
01156 0x5a9e, 0x5a78, 0x5aa6, 0x5a7c, 0x5aa5, 0x5aac, 0x5a95, 0x5aae,
01157 0x5a37, 0x5a84, 0x5a8a, 0x5a97, 0x5a83, 0x5a8b, 0x5aa9, 0x5a7b,
```

```

01158 0x5a7d, 0x5a8c, 0x5a9c, 0x5a8f, 0x5a93, 0x5a9d, 0x5bea, 0x5bcd,
01159 0x5bcb, 0x5bd4, 0x5bd1, 0x5bca, 0x5bce, 0x5c0c, 0x5c30, 0x5d37,
01160 0x5d43, 0x5d6b, 0x5d41, 0x5d4b, 0x5d3f, 0x5d35, 0x5d51, 0x5d4e,
01161 0x5d55, 0x5d33, 0x5d3a, 0x5d52, 0x5d3d, 0x5d31, 0x5d59, 0x5d42,
01162 0x5d39, 0x5d49, 0x5d38, 0x5d3c, 0x5d32, 0x5d36, 0x5d40, 0x5d45,
01163 0x5e44, 0x5e41, 0x5f58, 0x5fa6, 0x5fab, 0x60c9, 0x60b9,
01164 0x60cc, 0x60e2, 0x60ce, 0x60c4, 0x6114,
01165 /* 0xd9 */
01166 0x60f2, 0x610a, 0x6116, 0x6105, 0x60f5, 0x6113, 0x60f8, 0x60fc,
01167 0x60fe, 0x60c1, 0x6103, 0x6118, 0x611d, 0x6110, 0x60ff, 0x6104,
01168 0x610b, 0x624a, 0x6394, 0x63b1, 0x63b0, 0x63ce, 0x63e5, 0x63e8,
01169 0x63ef, 0x63c3, 0x649d, 0x63f3, 0x63ca, 0x63e0, 0x63f6, 0x63d5,
01170 0x63f2, 0x63f5, 0x6461, 0x63df, 0x63be, 0x63dd, 0x63dc, 0x63c4,
01171 0x63d8, 0x63d3, 0x63c2, 0x63c7, 0x63cc, 0x63cb, 0x63c8, 0x63f0,
01172 0x63d7, 0x63d9, 0x6532, 0x6567, 0x656a, 0x6564, 0x655c, 0x6568,
01173 0x6565, 0x658c, 0x659d, 0x659e, 0x65ae, 0x65d0, 0x65d2, 0x667c,
01174 0x666c, 0x667b, 0x6680, 0x6671, 0x6679, 0x666a, 0x6672, 0x6701,
01175 0x690c, 0x68d3, 0x6904, 0x68dc, 0x692a, 0x68ec, 0x68ea, 0x68f1,
01176 0x690f, 0x68d6, 0x68f7, 0x68eb, 0x68e4, 0x68f6, 0x6913, 0x6910,
01177 0x68f3, 0x68e1, 0x6907, 0x68cc, 0x6908, 0x6970, 0x68b4, 0x6911,
01178 0x68ef, 0x68c6, 0x6914, 0x68f8, 0x68d0, 0x68fd, 0x68fc, 0x68e8,
01179 0x690b, 0x690a, 0x6917, 0x68ce, 0x68c8, 0x68dd, 0x68de, 0x68e6,
01180 0x68f4, 0x68d1, 0x6906, 0x68d4, 0x68e9, 0x6915, 0x6925, 0x68c7,
01181 0x6b39, 0x6b3b, 0x6b3f, 0x6b3c, 0x6b94, 0x6b97, 0x6b99, 0x6b95,
01182 0x6bbd, 0x6bf0, 0x6bf2, 0x6bf3, 0x6c30, 0x6dfc, 0x6e46, 0x6e47,
01183 0x6e1f, 0x6e49, 0x6e88, 0x6e3c, 0x6e3d, 0x6e45, 0x6e62, 0x6e2b,
01184 0x6e3f, 0x6e41, 0x6e5d, 0x6e73, 0x6e1c, 0x6e33, 0x6e4b, 0x6e40,
01185 0x6e51, 0x6e3b, 0x6e03, 0x6e2e, 0x6e5e,
01186 /* 0xda */
01187 0x6e68, 0x6e5c, 0x6e61, 0x6e31, 0x6e28, 0x6e60, 0x6e71, 0x6e6b,
01188 0x6e39, 0x6e22, 0x6e30, 0x6e53, 0x6e65, 0x6e27, 0x6e78, 0x6e64,
01189 0x6e77, 0x6e55, 0x6e79, 0x6e52, 0x6e66, 0x6e35, 0x6e36, 0x6e5a,
01190 0x7120, 0x711e, 0x712f, 0x712f, 0x70fb, 0x712e, 0x7131, 0x7123, 0x7125,
01191 0x7122, 0x7132, 0x711f, 0x7128, 0x713a, 0x711b, 0x724b, 0x725a,
01192 0x7288, 0x7289, 0x7286, 0x7285, 0x728b, 0x7312, 0x730b, 0x7330,
01193 0x7322, 0x7331, 0x7333, 0x7327, 0x7332, 0x732d, 0x7326, 0x7323,
01194 0x7335, 0x730c, 0x742e, 0x742c, 0x7430, 0x742b, 0x7416, 0x741a,
01195 0x7421, 0x742d, 0x7431, 0x7424, 0x7423, 0x741d, 0x7429, 0x7420,
01196 0x7432, 0x74fb, 0x752f, 0x756f, 0x756c, 0x75e7, 0x75da, 0x75e1,
01197 0x75e6, 0x75dd, 0x75df, 0x75e4, 0x75d7, 0x7695, 0x7692, 0x76da,
01198 0x7746, 0x7747, 0x7744, 0x774d, 0x7745, 0x774a, 0x774e, 0x774b,
01199 0x774c, 0x77de, 0x77ec, 0x7860, 0x7864, 0x7865, 0x785c, 0x786d,
01200 0x7871, 0x786a, 0x786e, 0x7870, 0x7869, 0x7868, 0x785e, 0x7862,
01201 0x7974, 0x7973, 0x7972, 0x7970, 0x7a02, 0x7a0a, 0x7a03, 0x7a0c,
01202 0x7a04, 0x7a99, 0x7ae6, 0x7ae4, 0x7b4a, 0x7b3b, 0x7b44, 0x7b48,
01203 0x7b4c, 0x7b4e, 0x7b40, 0x7b58, 0x7b45, 0x7ca2, 0x7c9e, 0x7ca8,
01204 0x7ca1, 0x7d58, 0x7d6f, 0x7d63, 0x7d53, 0x7d56, 0x7d67, 0x7d6a,
01205 0x7d4f, 0x7d6d, 0x7d5c, 0x7d6b, 0x7d52, 0x7d54, 0x7d69, 0x7d51,
01206 0x7d5f, 0x7d4e, 0x7f3e, 0x7f3f, 0x7f65,
01207 /* 0xdb */
01208 0x7f66, 0x7fa2, 0x7fa0, 0x7fa1, 0x7fd7, 0x8051, 0x804f, 0x8050,
01209 0x80fe, 0x80d4, 0x8143, 0x814a, 0x8152, 0x814f, 0x8147, 0x813d,
01210 0x814d, 0x813a, 0x81e6, 0x81ee, 0x81f7, 0x81f8, 0x81f9, 0x8204,
01211 0x823c, 0x823b, 0x823f, 0x8275, 0x833b, 0x83cf, 0x83f9, 0x8423,
01212 0x83c0, 0x83e8, 0x8412, 0x83e7, 0x83e4, 0x83fc, 0x83f6, 0x8410,
01213 0x83c6, 0x83c8, 0x83eb, 0x83e3, 0x83bf, 0x8401, 0x83dd, 0x83e5,
01214 0x83d8, 0x83ff, 0x83e1, 0x83cb, 0x83ce, 0x83d6, 0x83f5, 0x83c9,
01215 0x8409, 0x840f, 0x83de, 0x8411, 0x8406, 0x83c2, 0x83f3, 0x83d5,
01216 0x83fa, 0x83c7, 0x83d1, 0x83ea, 0x8413, 0x83c3, 0x83ec, 0x83ee,
01217 0x83c4, 0x83fb, 0x83d7, 0x83e2, 0x841b, 0x83db, 0x83fe, 0x86d8,
01218 0x86e2, 0x86e6, 0x86d3, 0x86e3, 0x86da, 0x86ea, 0x86dd, 0x86eb,
01219 0x86dc, 0x86ec, 0x86e9, 0x86d7, 0x86e8, 0x86d1, 0x8848, 0x8856,
01220 0x8855, 0x88ba, 0x88d7, 0x88b9, 0x88b8, 0x88c0, 0x88be, 0x88b6,
01221 0x88bc, 0x88b7, 0x88bd, 0x88b2, 0x8901, 0x88c9, 0x8995, 0x8998,
01222 0x8997, 0x89dd, 0x89da, 0x89db, 0x8a4e, 0x8a4d, 0x8a39, 0x8a59,
01223 0x8a40, 0x8a57, 0x8a58, 0x8a44, 0x8a45, 0x8a52, 0x8a48, 0x8a51,
01224 0x8a4a, 0x8a4c, 0x8a4f, 0x8c5f, 0x8c81, 0x8c80, 0x8cba, 0x8cbe,
01225 0x8cb0, 0x8cb9, 0x8cb5, 0x8d84, 0x8d80, 0x8d89, 0x8dd8, 0x8dd3,
01226 0x8dcd, 0x8dc7, 0x8dd6, 0x8ddc, 0x8dcf, 0x8dd5, 0x8dd9, 0x8dc8,
01227 0x8dd7, 0x8dc5, 0x8eef, 0x8ef7, 0x8efa,
01228 /* 0xdc */
01229 0x8ef9, 0x8ee6, 0x8eee, 0x8ee5, 0x8ef5, 0x8ee7, 0x8ee8, 0x8ef6,
01230 0x8eeb, 0x8ef1, 0x8eec, 0x8ef4, 0x8ee9, 0x902d, 0x9034, 0x902f,
01231 0x9106, 0x912c, 0x9104, 0x90ff, 0x90fc, 0x9108, 0x90f9, 0x90fb,
01232 0x9101, 0x9100, 0x9107, 0x9105, 0x9103, 0x9161, 0x9164, 0x915f,
01233 0x9162, 0x9160, 0x9201, 0x920a, 0x9225, 0x9203, 0x921a, 0x9226,
01234 0x920f, 0x920c, 0x9200, 0x9212, 0x91ff, 0x91fd, 0x9206, 0x9204,
01235 0x9227, 0x9202, 0x921c, 0x9224, 0x9219, 0x9217, 0x9205, 0x9216,
01236 0x957b, 0x958d, 0x958c, 0x9590, 0x9687, 0x967e, 0x9688, 0x9689,
01237 0x9683, 0x9680, 0x96c2, 0x96c8, 0x96c3, 0x96f1, 0x96f0, 0x976c,
01238 0x9770, 0x976e, 0x9807, 0x98a9, 0x98eb, 0x9ce6, 0x9ef9, 0x4e83,
01239 0x4e84, 0x4eb6, 0x50bd, 0x50bf, 0x50c6, 0x50ae, 0x50c4, 0x50ca,
01240 0x50b4, 0x50c8, 0x50c2, 0x50b0, 0x50c1, 0x50ba, 0x50b1, 0x50cb,
01241 0x50c9, 0x50b6, 0x50b8, 0x51d7, 0x527a, 0x5278, 0x527b, 0x527c,
01242 0x55c3, 0x55db, 0x55cc, 0x55d0, 0x55cb, 0x55ca, 0x55dd, 0x55c0,
01243 0x55d4, 0x55c4, 0x55e9, 0x55bf, 0x55d2, 0x558d, 0x55cf, 0x55d5,
01244 0x55e2, 0x55d6, 0x55c8, 0x55f2, 0x55cd, 0x55d9, 0x55c2, 0x5714,

```



```
01245 0x5853, 0x5868, 0x5864, 0x584f, 0x584d, 0x5849, 0x586f, 0x5855,
01246 0x584e, 0x585d, 0x5859, 0x5865, 0x585b, 0x583d, 0x5863, 0x5871,
01247 0x58fc, 0x5ac7, 0x5ac4, 0x5acb, 0x5aba, 0x5ab8, 0x5ab1, 0x5ab5,
01248 0x5ab0, 0x5abf, 0x5ac8, 0x5abb, 0x5ac6,
01249 /* 0xdd */
01250 0x5ab7, 0x5ac0, 0x5aca, 0x5ab4, 0x5ab6, 0x5acd, 0x5ab9, 0x5a90,
01251 0x5bd6, 0x5bd8, 0x5bd9, 0x5c1f, 0x5c33, 0x5d71, 0x5d63, 0x5d4a,
01252 0x5d65, 0x5d72, 0x5d6c, 0x5d5e, 0x5d68, 0x5d67, 0x5d62, 0x5df0,
01253 0x5e4f, 0x5e4e, 0x5e4a, 0x5e4d, 0x5e4b, 0x5ec5, 0x5ecc, 0x5ec6,
01254 0x5ecb, 0x5ec7, 0x5f40, 0x5faf, 0x5fad, 0x60f7, 0x6149, 0x614a,
01255 0x612b, 0x6145, 0x6136, 0x6132, 0x612e, 0x6146, 0x612f, 0x614f,
01256 0x6129, 0x6140, 0x6220, 0x9168, 0x6223, 0x6225, 0x6224, 0x63c5,
01257 0x63f1, 0x63eb, 0x6410, 0x6412, 0x6409, 0x6420, 0x6424, 0x6433,
01258 0x6443, 0x641f, 0x6415, 0x6418, 0x6439, 0x6437, 0x6422, 0x6423,
01259 0x640c, 0x6426, 0x6430, 0x6430, 0x6428, 0x6441, 0x6435, 0x642f, 0x640a,
01260 0x641a, 0x6440, 0x6425, 0x6427, 0x640b, 0x63e7, 0x641b, 0x642e,
01261 0x6421, 0x640e, 0x656f, 0x6592, 0x65d3, 0x6686, 0x668c, 0x6695,
01262 0x6690, 0x668b, 0x668a, 0x6699, 0x6694, 0x6678, 0x6720, 0x6966,
01263 0x695f, 0x6938, 0x694e, 0x6962, 0x6971, 0x693f, 0x6945, 0x696a,
01264 0x6939, 0x6942, 0x6957, 0x6959, 0x697a, 0x6948, 0x6949, 0x6935,
01265 0x696c, 0x6932, 0x693d, 0x6965, 0x68f0, 0x6978, 0x6934, 0x6969,
01266 0x6940, 0x696f, 0x6944, 0x6976, 0x6958, 0x6941, 0x6974, 0x694c,
01267 0x693b, 0x694b, 0x6937, 0x695c, 0x694f, 0x6951, 0x6932, 0x6952,
01268 0x692f, 0x697b, 0x693c, 0x6b46, 0x6b45, 0x6b43, 0x6b42, 0x6b48,
01269 0x6b41, 0x6b9b, 0xfa0d, 0x6bfb, 0x6bfc,
01270 /* 0xde */
01271 0x6bf9, 0x6bf7, 0x6bf8, 0x6e9b, 0x6ed6, 0x6ec8, 0x6e8f, 0x6ec0,
01272 0x6e9f, 0x6e93, 0x6e94, 0x6ea0, 0x6eb1, 0x6eb9, 0x6ec6, 0x6ed2,
01273 0x6ebd, 0x6ec1, 0x6e9e, 0x6ec9, 0x6eb7, 0x6eb0, 0x6ecd, 0x6ea6,
01274 0x6ecf, 0x6eb2, 0x6ebe, 0x6ec3, 0x6edc, 0x6ed8, 0x6e99, 0x6e92,
01275 0x6e8e, 0x6e8d, 0x6ea4, 0x6ea1, 0x6ebf, 0x6eb3, 0x6ed0, 0x6eca,
01276 0x6e97, 0x6eae, 0x6ea3, 0x7147, 0x7154, 0x7152, 0x7163, 0x7160,
01277 0x7141, 0x715d, 0x7162, 0x7172, 0x7178, 0x716a, 0x7161, 0x7142,
01278 0x7158, 0x7143, 0x714b, 0x7170, 0x715f, 0x7150, 0x7153, 0x7144,
01279 0x714d, 0x715a, 0x724f, 0x728d, 0x728c, 0x7291, 0x7290, 0x728e,
01280 0x733c, 0x7342, 0x733b, 0x733a, 0x7340, 0x734a, 0x7349, 0x7444,
01281 0x744a, 0x744b, 0x7452, 0x7451, 0x7457, 0x7440, 0x744f, 0x7450,
01282 0x744e, 0x7442, 0x7446, 0x744d, 0x7454, 0x74e1, 0x74ff, 0x74fe,
01283 0x74fd, 0x751d, 0x7579, 0x7577, 0x6983, 0x75ef, 0x760f, 0x7603,
01284 0x75f7, 0x75fe, 0x75fc, 0x75f9, 0x75f8, 0x7610, 0x75fb, 0x75f6,
01285 0x75ed, 0x75f5, 0x75fd, 0x7699, 0x76b5, 0x76dd, 0x7755, 0x775f,
01286 0x7760, 0x7752, 0x7756, 0x775a, 0x7769, 0x7767, 0x7754, 0x7759,
01287 0x776d, 0x77e0, 0x7887, 0x789a, 0x7894, 0x788f, 0x7884, 0x7895,
01288 0x7885, 0x7886, 0x78a1, 0x7883, 0x7879, 0x7899, 0x7880, 0x7896,
01289 0x787b, 0x797c, 0x7982, 0x797d, 0x7979, 0x7a11, 0x7a18, 0x7a19,
01290 0x7a12, 0x7a17, 0x7a15, 0x7a22, 0x7a13,
01291 /* 0xdf */
01292 0x7a1b, 0x7a10, 0x7aa3, 0x7aa2, 0x7a9e, 0x7aeb, 0x7b66, 0x7b64,
01293 0x7b6d, 0x7b74, 0x7b69, 0x7b72, 0x7b65, 0x7b73, 0x7b71, 0x7b70,
01294 0x7b61, 0x7b78, 0x7b76, 0x7b63, 0x7cb2, 0x7cb4, 0x7caf, 0x7d88,
01295 0x7d86, 0x7d80, 0x7d8d, 0x7d7f, 0x7d85, 0x7d7a, 0x7d8e, 0x7d7b,
01296 0x7d83, 0x7d7c, 0x7d8c, 0x7d94, 0x7d84, 0x7d7d, 0x7d92, 0x7fd6,
01297 0x7fb6, 0x7f67, 0x7f68, 0x7f6c, 0x7fa6, 0x7fa5, 0x7fa7, 0x7fdb,
01298 0x7fdc, 0x8021, 0x8164, 0x8160, 0x8177, 0x815c, 0x8169, 0x815b,
01299 0x8162, 0x8172, 0x6721, 0x815e, 0x8176, 0x8167, 0x816f, 0x8144,
01300 0x8161, 0x821d, 0x8249, 0x8244, 0x8240, 0x8242, 0x8245, 0x84f1,
01301 0x843f, 0x8456, 0x8476, 0x8479, 0x848f, 0x848d, 0x8465, 0x8451,
01302 0x8440, 0x8486, 0x8467, 0x8430, 0x844d, 0x847d, 0x845a, 0x8459,
01303 0x8474, 0x8473, 0x845d, 0x8507, 0x845e, 0x8437, 0x843a, 0x8434,
01304 0x847a, 0x8443, 0x8478, 0x8432, 0x8445, 0x8429, 0x83d9, 0x844b,
01305 0x842f, 0x8442, 0x842d, 0x845f, 0x8470, 0x8439, 0x844e, 0x844c,
01306 0x8452, 0x846f, 0x84c5, 0x848e, 0x843b, 0x8447, 0x8436, 0x8433,
01307 0x8468, 0x847e, 0x8444, 0x842b, 0x8460, 0x8454, 0x846e, 0x8450,
01308 0x870b, 0x8704, 0x86f7, 0x870c, 0x86fa, 0x86d6, 0x86f5, 0x874d,
01309 0x86f8, 0x870e, 0x8709, 0x8701, 0x86f6, 0x870d, 0x8705, 0x88d6,
01310 0x88cb, 0x88cd, 0x88ce, 0x88de, 0x88db, 0x88da, 0x88cc, 0x88d0,
01311 0x8985, 0x899b, 0x89df, 0x89e5, 0x89e4,
01312 /* 0xe0 */
01313 0x89e1, 0x89e0, 0x89e2, 0x89dc, 0x89e6, 0x8a76, 0x8a86, 0x8a7f,
01314 0x8a61, 0x8a3f, 0x8a77, 0x8a82, 0x8a84, 0x8a75, 0x8a83, 0x8a81,
01315 0x8a74, 0x8a7a, 0x8c3c, 0x8c4b, 0x8c4a, 0x8c65, 0x8c64, 0x8c66,
01316 0x8c86, 0x8c84, 0x8c85, 0x8ccc, 0x8d68, 0x8d69, 0x8d91, 0x8d8c,
01317 0x8d8e, 0x8d8f, 0x8d8d, 0x8d93, 0x8d94, 0x8d90, 0x8d92, 0x8df0,
01318 0x8de0, 0x8dec, 0x8df1, 0x8dee, 0x8dd0, 0x8de9, 0x8de3, 0x8de2,
01319 0x8de7, 0x8df2, 0x8deb, 0x8df4, 0x8f06, 0x8eff, 0x8f01, 0x8f00,
01320 0x8f05, 0x8f07, 0x8f08, 0x8f02, 0x8f0b, 0x9052, 0x903f, 0x9044,
01321 0x9049, 0x903d, 0x9110, 0x910d, 0x910f, 0x9111, 0x9116, 0x9114,
01322 0x910b, 0x910e, 0x916e, 0x916f, 0x9248, 0x9252, 0x9230, 0x923a,
01323 0x9266, 0x9233, 0x9265, 0x925e, 0x9283, 0x922e, 0x924a, 0x9246,
01324 0x926d, 0x926c, 0x924f, 0x9260, 0x9267, 0x926f, 0x9236, 0x9261,
01325 0x9270, 0x9231, 0x9254, 0x9263, 0x9250, 0x9272, 0x924e, 0x9253,
01326 0x924c, 0x9256, 0x9232, 0x959f, 0x959c, 0x959e, 0x959b, 0x9692,
01327 0x9693, 0x9691, 0x9697, 0x9699, 0x96ce, 0x96fa, 0x96fd, 0x96f8, 0x96f5,
01328 0x9773, 0x9777, 0x9778, 0x9772, 0x980f, 0x980d, 0x980e, 0x98ac,
01329 0x98f6, 0x98f9, 0x99af, 0x99b2, 0x99b0, 0x99b5, 0x9aad, 0x9aab,
01330 0x9b5b, 0x9ce9, 0x9ced, 0x9ce7, 0x9e80, 0x9efd, 0x50e6, 0x50d4,
01331 0x50d7, 0x50e8, 0x50f3, 0x50db, 0x50ea, 0x50dd, 0x50e4, 0x50d3,
```

```

01332 0x50ec, 0x50f0, 0x50ef, 0x50e3, 0x50e0,
01333 /* 0xe1 */
01334 0x51d8, 0x5280, 0x5281, 0x52e9, 0x52eb, 0x5330, 0x53ac, 0x5627,
01335 0x5615, 0x560c, 0x5612, 0x55fc, 0x560f, 0x561c, 0x5601, 0x5613,
01336 0x5602, 0x55fa, 0x561d, 0x5604, 0x55ff, 0x55f9, 0x5889, 0x587c,
01337 0x5890, 0x5898, 0x5886, 0x5881, 0x587f, 0x5874, 0x588b, 0x587a,
01338 0x5887, 0x5891, 0x588e, 0x5876, 0x5882, 0x5888, 0x587b, 0x5894,
01339 0x588f, 0x58fe, 0x596b, 0x5adc, 0x5aee, 0x5ae5, 0x5ad5, 0x5aea,
01340 0x5ada, 0x5aed, 0x5aeb, 0x5af3, 0x5ae2, 0x5ae0, 0x5adb, 0x5aec,
01341 0x5ade, 0x5add, 0x5ad9, 0x5ae8, 0x5adf, 0x5b77, 0x5be0, 0x5be3,
01342 0x5c63, 0x5d82, 0x5d80, 0x5d7d, 0x5d86, 0x5d7a, 0x5d81, 0x5d77,
01343 0x5d8a, 0x5d89, 0x5d88, 0x5d7e, 0x5d7c, 0x5d8d, 0x5d79, 0x5d7f,
01344 0x5e58, 0x5e59, 0x5e53, 0x5ed8, 0x5ed1, 0x5ed7, 0x5ece, 0x5edc,
01345 0x5ed5, 0x5ed9, 0x5ed2, 0x5ed4, 0x5f44, 0x5f43, 0x5f6f, 0x5fb6,
01346 0x612c, 0x6128, 0x6141, 0x615e, 0x6171, 0x6173, 0x6152, 0x6153,
01347 0x6172, 0x616c, 0x6180, 0x6174, 0x6154, 0x617a, 0x615b, 0x6165,
01348 0x613b, 0x616a, 0x6161, 0x6156, 0x6229, 0x6227, 0x622b, 0x642b,
01349 0x644d, 0x645b, 0x645d, 0x6474, 0x6476, 0x6472, 0x6473, 0x647d,
01350 0x6475, 0x6466, 0x64a6, 0x644e, 0x6482, 0x645e, 0x645c, 0x644b,
01351 0x6453, 0x6460, 0x6450, 0x647f, 0x643f, 0x646c, 0x646b, 0x6459,
01352 0x6465, 0x6477, 0x6573, 0x65a0, 0x66a1, 0x66a0, 0x669f, 0x6705,
01353 0x6704, 0x6722, 0x69b1, 0x69b6, 0x69c9,
01354 /* 0xe2 */
01355 0x69a0, 0x69ce, 0x6996, 0x69b0, 0x69ac, 0x69bc, 0x6991, 0x6999,
01356 0x698e, 0x69a7, 0x698d, 0x69a9, 0x69be, 0x69af, 0x69bf, 0x69c4,
01357 0x69bd, 0x69a4, 0x69d4, 0x69b9, 0x69ca, 0x699a, 0x69cf, 0x69b3,
01358 0x6993, 0x69aa, 0x69d9, 0x699e, 0x69d9, 0x6997, 0x6990, 0x69c2,
01359 0x69b5, 0x69a5, 0x69c6, 0x6b4a, 0x6b4d, 0x6b4b, 0x6b9e, 0x6b9f,
01360 0x6ba0, 0x6bc3, 0x6bc4, 0x6bfe, 0x6ece, 0x6ef5, 0x6ef1, 0x6f03,
01361 0x6f25, 0x6ef8, 0x6ef3, 0x6efb, 0x6f2e, 0x6f09, 0x6f4e, 0x6f19,
01362 0x6f1a, 0x6f27, 0x6f18, 0x6f3b, 0x6f12, 0x6eed, 0x6f0a, 0x6f36,
01363 0x6f73, 0x6ef9, 0x6eee, 0x6f2d, 0x6f40, 0x6f30, 0x6f3c, 0x6f35,
01364 0x6eeb, 0x6f07, 0x6f0e, 0x6f43, 0x6f05, 0x6efd, 0x6ef6, 0x6f39,
01365 0x6f1c, 0x6efc, 0x6f3a, 0x6f1f, 0x6f0d, 0x6f1e, 0x6f08, 0x6f21,
01366 0x7187, 0x7190, 0x7189, 0x7180, 0x7185, 0x7182, 0x718f, 0x717b,
01367 0x7186, 0x7181, 0x7197, 0x7244, 0x7253, 0x7297, 0x7295, 0x7293,
01368 0x7343, 0x734d, 0x7351, 0x734c, 0x7462, 0x7473, 0x7471, 0x7475,
01369 0x7472, 0x7467, 0x746e, 0x7500, 0x7502, 0x7503, 0x757d, 0x7590,
01370 0x7616, 0x7608, 0x760c, 0x7615, 0x7611, 0x760a, 0x7614, 0x76b8,
01371 0x7781, 0x777c, 0x7785, 0x7782, 0x776e, 0x7780, 0x776f, 0x777e,
01372 0x7783, 0x78b2, 0x78aa, 0x78b4, 0x78ad, 0x78a8, 0x787e, 0x78ab,
01373 0x789e, 0x78a5, 0x78a0, 0x78ac, 0x78a2, 0x78a4, 0x7998, 0x798a,
01374 0x798b, 0x7996, 0x7995, 0x7994, 0x7993,
01375 /* 0xe3 */
01376 0x7997, 0x7988, 0x7992, 0x7990, 0x7a2b, 0x7a4a, 0x7a30, 0x7a2f,
01377 0x7a28, 0x7a26, 0x7aa8, 0x7aab, 0x7aac, 0x7aee, 0x7b88, 0x7b9c,
01378 0x7b8a, 0x7b91, 0x7b90, 0x7b96, 0x7b8d, 0x7b8c, 0x7b9b, 0x7b8e,
01379 0x7b85, 0x7b98, 0x5284, 0x7b99, 0x7ba4, 0x7b82, 0x7cbb, 0x7cbf,
01380 0x7cbc, 0x7cba, 0x7da7, 0x7db7, 0x7dc2, 0x7da3, 0x7daa, 0x7dc1,
01381 0x7dc0, 0x7dc5, 0x7d9d, 0x7dce, 0x7dc4, 0x7dc6, 0x7dcb, 0x7dcc,
01382 0x7daf, 0x7db9, 0x7d96, 0x7dbc, 0x7d9f, 0x7da6, 0x7dae, 0x7da9,
01383 0x7da1, 0x7dc9, 0x7ef3, 0x7fe2, 0x7fe3, 0x7fe5, 0x7fde, 0x8024,
01384 0x805d, 0x805c, 0x8189, 0x8186, 0x8183, 0x8187, 0x818d, 0x818c,
01385 0x818b, 0x8215, 0x8497, 0x84a4, 0x84a1, 0x849f, 0x84ba, 0x84ce,
01386 0x84c2, 0x84ac, 0x84ae, 0x84ab, 0x84b9, 0x84b4, 0x84c1, 0x84cd,
01387 0x84aa, 0x849a, 0x84b1, 0x84d0, 0x849d, 0x84a7, 0x84bb, 0x84a2,
01388 0x8494, 0x84c7, 0x84cc, 0x849b, 0x84a9, 0x84af, 0x84a8, 0x84d6,
01389 0x8498, 0x84b6, 0x84cf, 0x84a0, 0x84d7, 0x84d4, 0x84d2, 0x84db,
01390 0x84b0, 0x8491, 0x8661, 0x8733, 0x8723, 0x8728, 0x876b, 0x8740,
01391 0x872e, 0x871e, 0x8719, 0x871b, 0x8743, 0x872c, 0x8741,
01392 0x873e, 0x8746, 0x8720, 0x8732, 0x872a, 0x872d, 0x873c, 0x8712,
01393 0x873a, 0x8731, 0x8735, 0x8742, 0x8726, 0x8727, 0x8738, 0x8724,
01394 0x871a, 0x8730, 0x8711, 0x88f7, 0x88e7, 0x88f1, 0x88f2, 0x88fa,
01395 0x88fe, 0x88ee, 0x88fc, 0x88f6, 0x88fb,
01396 /* 0xe4 */
01397 0x88f0, 0x88ec, 0x88eb, 0x899d, 0x89a1, 0x899f, 0x899e, 0x89e9,
01398 0x89eb, 0x89e8, 0x8aab, 0x8a99, 0x8a8b, 0x8a92, 0x8a8f, 0x8a96,
01399 0x8c3d, 0x8c68, 0x8c69, 0x8cd5, 0x8ccf, 0x8cd7, 0x8d96, 0x8e09,
01400 0x8e02, 0x8dff, 0x8e0d, 0x8dfd, 0x8e0a, 0x8e03, 0x8e07, 0x8e06,
01401 0x8e05, 0x8dfe, 0x8e00, 0x8e04, 0x8f10, 0x8f11, 0x8f0e, 0x8f0d,
01402 0x9123, 0x911c, 0x9120, 0x9122, 0x911f, 0x911d, 0x911a, 0x9124,
01403 0x9121, 0x911b, 0x917a, 0x9172, 0x9179, 0x9173, 0x92a5, 0x92a4,
01404 0x9276, 0x929b, 0x927a, 0x92a0, 0x9294, 0x92aa, 0x928d, 0x92a6,
01405 0x929a, 0x92ab, 0x9279, 0x9297, 0x927f, 0x92a3, 0x92ee, 0x928e,
01406 0x9282, 0x9295, 0x92a2, 0x927d, 0x9288, 0x92a1, 0x928a, 0x9286,
01407 0x928c, 0x9299, 0x92a7, 0x927e, 0x9287, 0x92a9, 0x929d, 0x928b,
01408 0x922d, 0x969e, 0x96a1, 0x96ff, 0x9758, 0x977d, 0x977a, 0x977e,
01409 0x9783, 0x9780, 0x9782, 0x977b, 0x9784, 0x9781, 0x977f, 0x97ce,
01410 0x97cd, 0x9816, 0x98ad, 0x98ae, 0x9902, 0x9900, 0x9907, 0x999d,
01411 0x999c, 0x99c3, 0x99b9, 0x99bb, 0x99ba, 0x99c2, 0x99bd, 0x99c7,
01412 0x9ab1, 0x9ae3, 0x9ae7, 0x9b3e, 0x9b3f, 0x9b60, 0x9b61, 0x9b5f,
01413 0x9cf1, 0x9cf2, 0x9cf5, 0x9ea7, 0x50ff, 0x5103, 0x5130, 0x50f8,
01414 0x5106, 0x5107, 0x50f6, 0x50fe, 0x510b, 0x510c, 0x50fd, 0x510a,
01415 0x528b, 0x528c, 0x52f1, 0x52ef, 0x5648, 0x5642, 0x564c, 0x5635,
01416 0x5641, 0x564a, 0x5649, 0x5646, 0x5658,
01417 /* 0xe5 */
01418 0x565a, 0x5640, 0x5633, 0x563d, 0x562c, 0x563e, 0x5638, 0x562a,

```



```
01419 0x563a, 0x571a, 0x58ab, 0x589d, 0x58b1, 0x58a0, 0x58a3, 0x58af,
01420 0x58ac, 0x58a5, 0x58a1, 0x58ff, 0x5aff, 0x5af4, 0x5afd, 0x5af7,
01421 0x5af6, 0x5b03, 0x5af8, 0x5bf8, 0x5bf0, 0x5bf1, 0x5bf2, 0x5bf3,
01422 0x5b0f, 0x5c67, 0x5d99, 0x5d97, 0x5d9f, 0x5d92, 0x5da2, 0x5d93,
01423 0x5d95, 0x5da0, 0x5d9c, 0x5da1, 0x5d9a, 0x5d9e, 0x5e69, 0x5e5d,
01424 0x5e60, 0x5e5c, 0x7df3, 0x5edb, 0x5ede, 0x5ee1, 0x5f49, 0x5fb2,
01425 0x618b, 0x6183, 0x6179, 0x61b1, 0x61b0, 0x61a2, 0x6189, 0x619b,
01426 0x6193, 0x61af, 0x61ad, 0x619f, 0x6192, 0x61aa, 0x61a1, 0x618d,
01427 0x6166, 0x61b3, 0x622d, 0x646e, 0x6470, 0x6496, 0x64a0, 0x6485,
01428 0x6497, 0x649c, 0x648f, 0x648b, 0x648a, 0x648c, 0x64a3, 0x649f,
01429 0x6468, 0x64b1, 0x6498, 0x6576, 0x657a, 0x6579, 0x657b, 0x65b2,
01430 0x65b3, 0x66b5, 0x66b0, 0x66a9, 0x66b2, 0x66b7, 0x66aa, 0x66af,
01431 0x6a00, 0x6a06, 0x6a17, 0x69e5, 0x69f8, 0x6a15, 0x69f1, 0x69e4,
01432 0x6a20, 0x69ff, 0x69ec, 0x69e2, 0x6a1b, 0x6a1d, 0x69fe, 0x6a27,
01433 0x69f2, 0x69ee, 0x6a14, 0x69f7, 0x69e7, 0x6a40, 0x6a08, 0x69e6,
01434 0x69fb, 0x6a0d, 0x69fc, 0x69eb, 0x6a09, 0x6a04, 0x6a18, 0x6a25,
01435 0x6a0f, 0x69f6, 0x6a26, 0x6a07, 0x69f4, 0x6a16, 0x6b51, 0x6ba5,
01436 0x6ba3, 0x6ba2, 0x6ba6, 0x6c01, 0x6c00, 0x6bff, 0x6c02, 0x6f41,
01437 0x6f26, 0x6f7e, 0x6f87, 0x6fc6, 0x6f92,
01438 /* 0xe6 */
01439 0x6f8d, 0x6f89, 0x6f8c, 0x6f62, 0x6f4f, 0x6f85, 0x6f5a, 0x6f96,
01440 0x6f76, 0x6f6c, 0x6f82, 0x6f55, 0x6f72, 0x6f52, 0x6f50, 0x6f57,
01441 0x6f94, 0x6f93, 0x6f5d, 0x6f00, 0x6f61, 0x6f6b, 0x6f7d, 0x6f67,
01442 0x6f90, 0x6f53, 0x6f8b, 0x6f69, 0x6f7f, 0x6f95, 0x6f63, 0x6f77,
01443 0x6f6a, 0x6f7b, 0x71b2, 0x71af, 0x719b, 0x71b0, 0x71a0, 0x719a,
01444 0x71a9, 0x71b5, 0x719d, 0x71a5, 0x719e, 0x71a4, 0x71a1, 0x71aa,
01445 0x719c, 0x71a7, 0x71b3, 0x7298, 0x729a, 0x7358, 0x7352, 0x735e,
01446 0x735f, 0x7360, 0x735d, 0x735b, 0x7361, 0x735a, 0x7359, 0x7362,
01447 0x7487, 0x7489, 0x748a, 0x7486, 0x7481, 0x747d, 0x7485, 0x7488,
01448 0x747c, 0x7479, 0x7508, 0x7507, 0x757e, 0x7625, 0x761e, 0x7619,
01449 0x761d, 0x761c, 0x7623, 0x761a, 0x7628, 0x761b, 0x769c, 0x769d,
01450 0x769e, 0x769b, 0x778d, 0x778f, 0x7789, 0x7788, 0x78cd, 0x78bb,
01451 0x78cf, 0x78cc, 0x78d1, 0x78ce, 0x78d4, 0x78c8, 0x78c3, 0x78c4,
01452 0x78c9, 0x799a, 0x79a1, 0x79a0, 0x799c, 0x79a2, 0x799b, 0x6b76,
01453 0x7a39, 0x7ab2, 0x7ab4, 0x7ab3, 0x7bb7, 0x7bcb, 0x7bbe, 0x7bac,
01454 0x7bce, 0x7baf, 0x7bb9, 0x7bca, 0x7bb5, 0x7cc5, 0x7cc8, 0x7ccc,
01455 0x7ccb, 0x7df7, 0x7ddb, 0x7dea, 0x7de7, 0x7dd7, 0x7del, 0x7e03,
01456 0x7dfa, 0x7de6, 0x7df6, 0x7df1, 0x7df0, 0x7dee, 0x7ddc, 0x7f76,
01457 0x7fac, 0x7fb0, 0x7fad, 0x7fed, 0x7feb, 0x7fea, 0x7fec, 0x7fe6,
01458 0x7fe8, 0x8064, 0x8067, 0x81a3, 0x819f,
01459 /* 0xe7 */
01460 0x819e, 0x8195, 0x81a2, 0x8199, 0x8197, 0x8216, 0x824f, 0x8253,
01461 0x8252, 0x8250, 0x824e, 0x8251, 0x8524, 0x853b, 0x850f, 0x8500,
01462 0x8529, 0x850e, 0x8509, 0x850d, 0x851f, 0x850a, 0x8527, 0x851c,
01463 0x84fb, 0x852b, 0x84fa, 0x8508, 0x850c, 0x84f4, 0x852a, 0x84f2,
01464 0x8515, 0x84f7, 0x84eb, 0x84f3, 0x84fc, 0x8512, 0x84ea, 0x84e9,
01465 0x8516, 0x84fe, 0x8528, 0x851d, 0x852e, 0x8502, 0x84fd, 0x851e,
01466 0x84f6, 0x8531, 0x8526, 0x84e7, 0x84e8, 0x84f0, 0x84ef, 0x84f9,
01467 0x8518, 0x8520, 0x8530, 0x850b, 0x8519, 0x852f, 0x8662, 0x8756,
01468 0x8763, 0x8764, 0x8777, 0x87e1, 0x8773, 0x8758, 0x8754, 0x875b,
01469 0x8752, 0x8761, 0x875a, 0x8751, 0x875e, 0x876d, 0x876a, 0x8750,
01470 0x874e, 0x875f, 0x875d, 0x876f, 0x876c, 0x877a, 0x876e, 0x875c,
01471 0x8765, 0x874f, 0x877b, 0x8775, 0x8762, 0x8767, 0x8769, 0x885a,
01472 0x8905, 0x890c, 0x8914, 0x890b, 0x8917, 0x8918, 0x8919, 0x8906,
01473 0x8916, 0x8911, 0x890e, 0x8909, 0x89a2, 0x89a4, 0x89a3, 0x89ed,
01474 0x89f0, 0x89ec, 0x8acf, 0x8ac6, 0x8ab8, 0x8ad3, 0x8ad1, 0x8ad4,
01475 0x8ad5, 0x8abb, 0x8ad7, 0x8abe, 0x8ac0, 0x8ac5, 0x8ad8, 0x8ac3,
01476 0x8aba, 0x8abd, 0x8ad9, 0x8c3e, 0x8c4d, 0x8c8f, 0x8ce5, 0x8cdf,
01477 0x8cd9, 0x8ce8, 0x8cda, 0x8cdd, 0x8ce7, 0x8da0, 0x8d9c, 0x8dal,
01478 0x8d9b, 0x8e20, 0x8e23, 0x8e25, 0x8e24, 0x8e2e, 0x8e15, 0x8e1b,
01479 0x8e16, 0x8e11, 0x8e19, 0x8e26, 0x8e27,
01480 /* 0xe8 */
01481 0x8e14, 0x8e12, 0x8e18, 0x8e13, 0x8e1c, 0x8e17, 0x8e1a, 0x8f2c,
01482 0x8f24, 0x8f18, 0x8f1a, 0x8f20, 0x8f23, 0x8f16, 0x8f17, 0x9073,
01483 0x9070, 0x906f, 0x9067, 0x906b, 0x912f, 0x912b, 0x9129, 0x912a,
01484 0x9132, 0x9126, 0x912e, 0x9185, 0x9186, 0x918a, 0x9181, 0x9182,
01485 0x9184, 0x9180, 0x92d0, 0x92c3, 0x92c4, 0x92c0, 0x92d9, 0x92b6,
01486 0x92cf, 0x92f1, 0x92df, 0x92d8, 0x92e9, 0x92d7, 0x92dd, 0x92cc,
01487 0x92ef, 0x92c2, 0x92e8, 0x92ca, 0x92c8, 0x92ce, 0x92e6, 0x92cd,
01488 0x92d5, 0x92c9, 0x92e0, 0x92de, 0x92e7, 0x92d1, 0x92d3, 0x92b5,
01489 0x92e1, 0x92c6, 0x92b4, 0x957c, 0x95ac, 0x95ab, 0x95ae, 0x95b0,
01490 0x96a4, 0x96a2, 0x96d3, 0x9705, 0x9708, 0x9702, 0x975a, 0x978a,
01491 0x978e, 0x9788, 0x97d0, 0x97cf, 0x981e, 0x981d, 0x9826, 0x9829,
01492 0x9828, 0x9820, 0x981b, 0x9827, 0x98b2, 0x9908, 0x98fa, 0x9911,
01493 0x9914, 0x9916, 0x9917, 0x9915, 0x99dc, 0x99cd, 0x99cf, 0x99d3,
01494 0x99d4, 0x99ce, 0x99c9, 0x99d6, 0x99d8, 0x99cb, 0x99d7, 0x99cc,
01495 0x9ab3, 0x9aec, 0x9aeb, 0x9af3, 0x9af2, 0x9af1, 0x9b46, 0x9b43,
01496 0x9b67, 0x9b74, 0x9b71, 0x9b66, 0x9b76, 0x9b75, 0x9b70, 0x9b68,
01497 0x9b64, 0x9b6c, 0x9cfc, 0x9cfa, 0x9cfd, 0x9cff, 0x9cf7, 0x9d07,
01498 0x9d00, 0x9cf9, 0x9cfb, 0x9d08, 0x9d05, 0x9d04, 0x9e83, 0x9ed3,
01499 0x9f0f, 0x9f10, 0x511c, 0x5113, 0x5117, 0x511a, 0x5111, 0x51de,
01500 0x5334, 0x53e1, 0x5670, 0x5660, 0x566e,
01501 /* 0xe9 */
01502 0x5673, 0x5666, 0x5663, 0x566d, 0x5672, 0x565e, 0x5677, 0x571c,
01503 0x571b, 0x58c8, 0x58bd, 0x58c9, 0x58bf, 0x58ba, 0x58c2, 0x58bc,
01504 0x58c6, 0x5b17, 0x5b19, 0x5b1b, 0x5b21, 0x5b14, 0x5b13, 0x5b10,
01505 0x5b16, 0x5b28, 0x5b1a, 0x5b20, 0x5b1e, 0x5bef, 0x5dac, 0x5db1,
```

```

01506 0x5da9, 0x5da7, 0x5db5, 0x5db0, 0x5dae, 0x5daa, 0x5da8, 0x5db2,
01507 0x5dad, 0x5daf, 0x5db4, 0x5e67, 0x5e68, 0x5e66, 0x5e6f, 0x5ee9,
01508 0x5ee7, 0x5ee6, 0x5ee8, 0x5ee5, 0x5f4b, 0x5fbc, 0x619d, 0x61a8,
01509 0x6196, 0x61c5, 0x61b4, 0x61c6, 0x61c1, 0x61cc, 0x61ba, 0x61bf,
01510 0x61b8, 0x618c, 0x64d7, 0x64d6, 0x64d0, 0x64cf, 0x64c9, 0x64bd,
01511 0x6489, 0x64c3, 0x64db, 0x64f3, 0x64d9, 0x6533, 0x657f, 0x657c,
01512 0x65a2, 0x66c8, 0x66be, 0x66c0, 0x66ca, 0x66cb, 0x66cf, 0x66bd,
01513 0x66bb, 0x66ba, 0x66cc, 0x6723, 0x6a34, 0x6a66, 0x6a49, 0x6a67,
01514 0x6a32, 0x6a68, 0x6a3e, 0x6a5d, 0x6a6d, 0x6a76, 0x6a5b, 0x6a51,
01515 0x6a28, 0x6a5a, 0x6a3b, 0x6a3f, 0x6a41, 0x6a6a, 0x6a64, 0x6a50,
01516 0x6a4f, 0x6a54, 0x6a6f, 0x6a69, 0x6a60, 0x6a3c, 0x6a5e, 0x6a56,
01517 0x6a55, 0x6a4d, 0x6a4e, 0x6a46, 0x6b55, 0x6b54, 0x6b56, 0x6ba7,
01518 0x6baa, 0x6bab, 0x6bc8, 0x6bc7, 0x6c04, 0x6c03, 0x6c06, 0x6fad,
01519 0x6fcb, 0x6fa3, 0x6fc7, 0x6fbc, 0x6fce, 0x6fc8, 0x6f5e, 0x6fc4,
01520 0x6fbd, 0x6f9e, 0x6fca, 0x6fa8, 0x7004, 0x6fa5, 0x6fae, 0x6fba,
01521 0x6fac, 0x6faa, 0x6fcf, 0x6fbf, 0x6fb8,
01522 /* 0xea */
01523 0x6fa2, 0x6fc9, 0x6fab, 0x6fcd, 0x6faf, 0x6fb2, 0x6fb0, 0x71c5,
01524 0x71c2, 0x71bf, 0x71b8, 0x71d6, 0x71c0, 0x71c1, 0x71cb, 0x71d4,
01525 0x71ca, 0x71c7, 0x71cf, 0x71bd, 0x71d8, 0x71bc, 0x71c6, 0x71da,
01526 0x71db, 0x729d, 0x729e, 0x7369, 0x7366, 0x7367, 0x736c, 0x7365,
01527 0x736b, 0x736a, 0x747f, 0x749a, 0x74a0, 0x7494, 0x7492, 0x7495,
01528 0x74a1, 0x750b, 0x7580, 0x762f, 0x762d, 0x7631, 0x763d, 0x7633,
01529 0x763c, 0x7635, 0x7632, 0x7630, 0x76bb, 0x76e6, 0x779a, 0x779d,
01530 0x77a1, 0x779c, 0x779b, 0x77a2, 0x77a3, 0x7795, 0x7799, 0x7797,
01531 0x78dd, 0x78e9, 0x78e5, 0x78ea, 0x78de, 0x78e3, 0x78db, 0x78e1,
01532 0x78e2, 0x78ed, 0x78df, 0x78e0, 0x79a4, 0x7a44, 0x7a48, 0x7a47,
01533 0x7ab6, 0x7ab8, 0x7ab5, 0x7ab1, 0x7ab7, 0x7bde, 0x7be3, 0x7be7,
01534 0x7bdd, 0x7bd5, 0x7be5, 0x7bda, 0x7be8, 0x7bf9, 0x7bd4, 0x7bea,
01535 0x7be2, 0x7bdc, 0x7beb, 0x7bd8, 0x7bdf, 0x7cd2, 0x7cd4, 0x7cd7,
01536 0x7cd0, 0x7cd1, 0x7e12, 0x7e21, 0x7e17, 0x7e0c, 0x7e1f, 0x7e20,
01537 0x7e13, 0x7e0e, 0x7e1c, 0x7e15, 0x7e1a, 0x7e22, 0x7e0b, 0x7e0f,
01538 0x7e16, 0x7e0d, 0x7e14, 0x7e25, 0x7e24, 0x7f43, 0x7f7b, 0x7f7c,
01539 0x7f7a, 0x7fb1, 0x7fef, 0x802a, 0x8029, 0x806c, 0x81b1, 0x81a6,
01540 0x81ae, 0x81b9, 0x81b5, 0x81ab, 0x81b0, 0x81ac, 0x81b4, 0x81b2,
01541 0x81b7, 0x81a7, 0x81f2, 0x8255, 0x8256, 0x8257, 0x8556, 0x8545,
01542 0x856b, 0x854d, 0x8553, 0x8561, 0x8558,
01543 /* 0xeb */
01544 0x8540, 0x8546, 0x854e, 0x8541, 0x8562, 0x8544, 0x8551, 0x8547,
01545 0x8563, 0x853e, 0x855b, 0x8571, 0x854e, 0x856e, 0x8575, 0x8555,
01546 0x8567, 0x8560, 0x858c, 0x8566, 0x855d, 0x8554, 0x8565, 0x856c,
01547 0x8663, 0x8665, 0x8664, 0x879b, 0x879f, 0x8797, 0x8793, 0x8792,
01548 0x8788, 0x8781, 0x8796, 0x8798, 0x8779, 0x8787, 0x87a3, 0x8785,
01549 0x8790, 0x8791, 0x879d, 0x8784, 0x8794, 0x879c, 0x879a, 0x8789,
01550 0x891e, 0x8926, 0x892d, 0x892e, 0x892f, 0x892c, 0x891f, 0x89f1, 0x8ae0, 0x8ae2,
01551 0x8af2, 0x8af4, 0x8af5, 0x8add, 0x8b14, 0x8ae4, 0x8adf, 0x8af0,
01552 0x8ac8, 0x8ade, 0x8ae1, 0x8ae8, 0x8aff, 0x8aef, 0x8afb, 0x8c91,
01553 0x8c92, 0x8c90, 0x8cf5, 0x8cee, 0x8cf1, 0x8cf0, 0x8cf3, 0x8dc6,
01554 0x8d6e, 0x8da5, 0x8da7, 0x8e33, 0x8e3e, 0x8e38, 0x8e40, 0x8e45,
01555 0x8e36, 0x8e3c, 0x8e3d, 0x8e41, 0x8e30, 0x8e3f, 0x8ebd, 0x8f36,
01556 0x8f2e, 0x8f35, 0x8f32, 0x8f39, 0x8f37, 0x8f34, 0x9076, 0x9079,
01557 0x907b, 0x9086, 0x90fa, 0x9133, 0x9135, 0x9136, 0x9193, 0x9190,
01558 0x9191, 0x918d, 0x918f, 0x9327, 0x931e, 0x9308, 0x931f, 0x9306,
01559 0x930f, 0x937a, 0x9338, 0x933c, 0x931b, 0x9323, 0x9312, 0x9301,
01560 0x9346, 0x932d, 0x930e, 0x930d, 0x92cb, 0x931d, 0x92fa, 0x9325,
01561 0x9313, 0x92f9, 0x92f7, 0x9334, 0x9302, 0x9324, 0x92ff, 0x9329,
01562 0x9339, 0x9335, 0x932a, 0x9314, 0x930c,
01563 /* 0xec */
01564 0x930b, 0x92fe, 0x9309, 0x9300, 0x92fb, 0x9316, 0x95bc, 0x95cd,
01565 0x95be, 0x95b9, 0x95ba, 0x95b6, 0x95bf, 0x95b5, 0x95bd, 0x96a9,
01566 0x96d4, 0x970b, 0x9712, 0x9710, 0x9799, 0x9797, 0x9794, 0x97f0,
01567 0x97f8, 0x9835, 0x982f, 0x9832, 0x9924, 0x991f, 0x9927, 0x9929,
01568 0x999e, 0x99ee, 0x99ec, 0x99e5, 0x99e4, 0x99f0, 0x99e3, 0x99ea,
01569 0x99e9, 0x99e7, 0x99ab, 0x99bf, 0x99ab, 0x99ab, 0x99af, 0x99af,
01570 0x99af, 0x99af, 0x99b3, 0x99b8, 0x99b5, 0x99b8, 0x99b7, 0x99b7,
01571 0x99b7, 0x99b8, 0x99b9, 0x99b2, 0x99b9, 0x99b7, 0x99b9, 0x99b7,
01572 0x99b8, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01573 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01574 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01575 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01576 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01577 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01578 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01579 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01580 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01581 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01582 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01583 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01584 0x99d1, 0x99d2, 0x99d1, 0x99d0, 0x99d1, 0x99d1, 0x99d1, 0x99d1,
01585 /* 0xed */
01586 0x6a95, 0x6a83, 0x6aa8, 0x6aa4, 0x6a91, 0x6a7f, 0x6aa6, 0x6a9a,
01587 0x6a85, 0x6a8c, 0x6a92, 0x6b5b, 0x6bad, 0x6c09, 0x6fcc, 0x6fa9,
01588 0x6ff4, 0x6ffd, 0x6fe3, 0x6fed, 0x6fed, 0x6fed, 0x6fed, 0x6fed,
01589 0x6ff2, 0x6ffd, 0x6fe2, 0x6fe8, 0x71e1, 0x71f1, 0x71e8, 0x71f2,
01590 0x71e4, 0x71f0, 0x71e2, 0x7373, 0x736e, 0x736f, 0x7497, 0x74b2,
01591 0x74ab, 0x7490, 0x74ad, 0x74b1, 0x74a5, 0x74af, 0x7510,
01592 0x7511, 0x7512, 0x750f, 0x7584, 0x7643, 0x7648, 0x7649, 0x7647,

```

```
01593 0x76a4, 0x76e9, 0x77b5, 0x77ab, 0x77b2, 0x77b7, 0x77b6, 0x77b4,
01594 0x77b1, 0x77a8, 0x77f0, 0x78f3, 0x78fd, 0x7902, 0x78fb, 0x78fc,
01595 0x78f2, 0x7905, 0x78f9, 0x78fe, 0x7904, 0x79ab, 0x79a8, 0x7a5c,
01596 0x7a5b, 0x7a56, 0x7a58, 0x7a54, 0x7a5a, 0x7abe, 0x7ac0, 0x7ac1,
01597 0x7c05, 0x7c0f, 0x7bf2, 0x7c00, 0x7bfb, 0x7bfb, 0x7c0e, 0x7bf4,
01598 0x7c0b, 0x7bfb, 0x7bf3, 0x7c02, 0x7c09, 0x7c03, 0x7c01, 0x7bfb,
01599 0x7c06, 0x7bf0, 0x7bf1, 0x7c10, 0x7c0a, 0x7ce8, 0x7e2d, 0x7e3c,
01600 0x7e42, 0x7e33, 0x9848, 0x7e38, 0x7e2a, 0x7e49, 0x7e40, 0x7e47,
01601 0x7e29, 0x7e4c, 0x7e30, 0x7e3b, 0x7e36, 0x7e44, 0x7e3a, 0x7f45,
01602 0x7f7f, 0x7f7e, 0x7f7d, 0x7ff4, 0x7ff2, 0x802c, 0x81bb, 0x81c4,
01603 0x81cc, 0x81ca, 0x81c5, 0x81c7, 0x81bc, 0x81e9, 0x825b, 0x825a,
01604 0x825c, 0x8583, 0x8580, 0x858f, 0x85a7, 0x8595, 0x85a0, 0x858b,
01605 0x85a3, 0x857b, 0x85a4, 0x859a, 0x859e,
01606 /* 0xee */
01607 0x8577, 0x857c, 0x8589, 0x85a1, 0x857a, 0x8578, 0x8557, 0x858e,
01608 0x8596, 0x8586, 0x858d, 0x8599, 0x859d, 0x8581, 0x85a2, 0x8582,
01609 0x8588, 0x8585, 0x8579, 0x8576, 0x8598, 0x8590, 0x859f, 0x8668,
01610 0x87be, 0x87aa, 0x87ad, 0x87c5, 0x87b0, 0x87ac, 0x87b9, 0x87b5,
01611 0x87bc, 0x87ae, 0x87c9, 0x87c3, 0x87c2, 0x87cc, 0x87b7, 0x87af,
01612 0x87c4, 0x87ca, 0x87b4, 0x87b6, 0x87bf, 0x87b8, 0x87bd, 0x87de,
01613 0x87b2, 0x8935, 0x8933, 0x893c, 0x893e, 0x8941, 0x8952, 0x8937,
01614 0x8942, 0x89ad, 0x89af, 0x89ae, 0x89f2, 0x89f3, 0x8b1e, 0x8b18,
01615 0x8b16, 0x8b11, 0x8b05, 0x8b0b, 0x8b22, 0x8b0f, 0x8b12, 0x8b15,
01616 0x8b07, 0x8b0d, 0x8b08, 0x8b06, 0x8b1c, 0x8b13, 0x8b1a, 0x8c4f,
01617 0x8c70, 0x8c72, 0x8c71, 0x8c6f, 0x8c95, 0x8c94, 0x8cf9, 0x8d6f,
01618 0x8e4e, 0x8e4d, 0x8e53, 0x8e50, 0x8e4c, 0x8e47, 0x8f43, 0x8f40,
01619 0x9085, 0x907e, 0x9138, 0x913e, 0x91a2, 0x919b, 0x9199, 0x919f,
01620 0x91a1, 0x919d, 0x91a0, 0x93a1, 0x9383, 0x93af, 0x9364, 0x9356,
01621 0x9347, 0x937c, 0x9358, 0x935c, 0x9376, 0x9349, 0x9350, 0x9351,
01622 0x9360, 0x936d, 0x938f, 0x934c, 0x936a, 0x9379, 0x9357, 0x9355,
01623 0x9352, 0x934f, 0x9371, 0x9377, 0x937b, 0x9361, 0x935e, 0x9363,
01624 0x9367, 0x9380, 0x934e, 0x9359, 0x95c7, 0x95c0, 0x95c9, 0x95c3,
01625 0x95c5, 0x95b7, 0x96ae, 0x96b0, 0x96ac, 0x9720, 0x971f, 0x9718,
01626 0x971d, 0x9719, 0x979a, 0x97a1, 0x979c,
01627 /* 0xef */
01628 0x979e, 0x979d, 0x97d5, 0x97d4, 0x97f1, 0x9841, 0x9844, 0x984a,
01629 0x9849, 0x9845, 0x9843, 0x9925, 0x992b, 0x992c, 0x992a, 0x9933,
01630 0x9932, 0x992f, 0x992d, 0x9931, 0x9930, 0x9998, 0x99a3, 0x99a1,
01631 0x9a02, 0x99fa, 0x99f4, 0x99f7, 0x99f9, 0x99f6, 0x99fb,
01632 0x99fd, 0x99fe, 0x99fc, 0x9a03, 0x9abe, 0x9afe, 0x9afd, 0x9b01,
01633 0x9afc, 0x9b48, 0x9b9a, 0x9ba8, 0x9b9e, 0x9b9b, 0x9ba6, 0x9ba1,
01634 0x9ba5, 0x9ba4, 0x9b86, 0x9ba2, 0x9ba0, 0x9baf, 0x9d33, 0x9d41,
01635 0x9d67, 0x9d36, 0x9d2e, 0x9d2f, 0x9d31, 0x9d38, 0x9d30, 0x9d45,
01636 0x9d42, 0x9d43, 0x9d3e, 0x9d37, 0x9d40, 0x9d3d, 0x7ff5, 0x9d2d,
01637 0x9e8a, 0x9e89, 0x9e8d, 0x9eb0, 0x9ec8, 0x9eda, 0x9efb, 0x9eff,
01638 0x9f24, 0x9f23, 0x9f22, 0x9f54, 0x9fa0, 0x5131, 0x512d, 0x512e,
01639 0x5698, 0x569c, 0x5697, 0x569a, 0x569d, 0x5699, 0x5970, 0x5b3c,
01640 0x5c69, 0x5c6a, 0x5dc0, 0x5e6d, 0x5e6e, 0x61d8, 0x61df, 0x61ed,
01641 0x61ee, 0x61f1, 0x61ea, 0x61f0, 0x61eb, 0x61d6, 0x61e9, 0x64ff,
01642 0x6504, 0x64fd, 0x64f8, 0x6501, 0x6503, 0x64fc, 0x6594, 0x65db,
01643 0x66da, 0x66db, 0x66d8, 0x6ac5, 0x6ab9, 0x6abd, 0x6ae1, 0x6ac6,
01644 0x6aba, 0x6ab6, 0x6ab7, 0x6ac7, 0x6ab4, 0x6aad, 0x6b5e, 0x6bc9,
01645 0x6c0b, 0x7007, 0x700c, 0x700d, 0x7001, 0x7005, 0x7014, 0x700e,
01646 0x6fff, 0x7000, 0x6ffb, 0x7026, 0x6ffc, 0x6fff, 0x700a, 0x7201,
01647 0x71ff, 0x71f9, 0x7203, 0x71fd, 0x7376,
01648 /* 0xf0 */
01649 0x74b8, 0x74c0, 0x74b5, 0x74c1, 0x74be, 0x74b6, 0x74bb, 0x74c2,
01650 0x7514, 0x7513, 0x765c, 0x7664, 0x7659, 0x7650, 0x7653, 0x7657,
01651 0x765a, 0x76a6, 0x76bd, 0x76ec, 0x77c2, 0x77ba, 0x78ff, 0x790c,
01652 0x7913, 0x7914, 0x7909, 0x7910, 0x7912, 0x7911, 0x79ad, 0x79ac,
01653 0x7a5f, 0x7c1c, 0x7c29, 0x7c19, 0x7c20, 0x7c1f, 0x7c2d, 0x7c1d,
01654 0x7c26, 0x7c28, 0x7c22, 0x7c25, 0x7c30, 0x7e5c, 0x7e50, 0x7e56,
01655 0x7e63, 0x7e58, 0x7e62, 0x7e5f, 0x7e51, 0x7e60, 0x7e57, 0x7e53,
01656 0x7fb5, 0x7fb3, 0x7ff7, 0x7ff8, 0x8075, 0x81d1, 0x81d2, 0x81d0,
01657 0x825f, 0x825e, 0x85b4, 0x85c6, 0x85c0, 0x85c3, 0x85c2, 0x85b3,
01658 0x85b5, 0x85bd, 0x85c7, 0x85c4, 0x85bf, 0x85cb, 0x85ce, 0x85c8,
01659 0x85c5, 0x85b1, 0x85b6, 0x85d2, 0x8624, 0x85b8, 0x85b7, 0x85be,
01660 0x8669, 0x87e7, 0x87e6, 0x87e2, 0x87db, 0x87eb, 0x87ea, 0x87e5,
01661 0x87df, 0x87f3, 0x87e4, 0x87d4, 0x87dc, 0x87d3, 0x87ed, 0x87d8,
01662 0x87e3, 0x87a4, 0x87d7, 0x87d9, 0x8801, 0x87f4, 0x87e8, 0x87dd,
01663 0x8953, 0x894b, 0x894f, 0x894c, 0x8946, 0x8950, 0x8951, 0x8949,
01664 0x8b2a, 0x8b27, 0x8b23, 0x8b33, 0x8b30, 0x8b35, 0x8b47, 0x8b2f,
01665 0x8b3c, 0x8b3e, 0x8b31, 0x8b25, 0x8b37, 0x8b26, 0x8b36, 0x8b2e,
01666 0x8b24, 0x8b3b, 0x8b3d, 0x8b3a, 0x8c42, 0x8c75, 0x8c99, 0x8c98,
01667 0x8c97, 0x8cfe, 0x8d04, 0x8d02, 0x8d00, 0x8e5c, 0x8e62, 0x8e60,
01668 0x8e57, 0x8e56, 0x8e5e, 0x8e65, 0x8e67,
01669 /* 0xf1 */
01670 0x8e5b, 0x8e5a, 0x8e61, 0x8e5d, 0x8e69, 0x8e54, 0x8f46, 0x8f47,
01671 0x8f48, 0x8f4b, 0x9128, 0x913a, 0x913b, 0x913e, 0x91a8, 0x91a5,
01672 0x91a7, 0x91af, 0x91aa, 0x93b5, 0x938c, 0x9392, 0x93b7, 0x939b,
01673 0x939d, 0x9389, 0x93a7, 0x938e, 0x93aa, 0x939e, 0x93a6, 0x9395,
01674 0x9388, 0x9399, 0x939f, 0x938d, 0x93b1, 0x9391, 0x93b2, 0x93a4,
01675 0x93a8, 0x93b4, 0x93a3, 0x93a5, 0x95d2, 0x95d3, 0x95d1, 0x96b3,
01676 0x96d7, 0x96da, 0x5dc2, 0x96df, 0x96d8, 0x96dd, 0x9723, 0x9722,
01677 0x9725, 0x97ac, 0x97ae, 0x97a8, 0x97ab, 0x97a4, 0x97aa, 0x97a2,
01678 0x97a5, 0x97d7, 0x97d9, 0x97d6, 0x97d8, 0x97fa, 0x9850, 0x9851,
01679 0x9852, 0x98b8, 0x9941, 0x993c, 0x993a, 0x9a0f, 0x9a0b, 0x9a09,
```

```

01680 0x9a0d, 0x9a04, 0x9a11, 0x9a0a, 0x9a05, 0x9a07, 0x9a06, 0x9ac0,
01681 0x9adc, 0x9b08, 0x9b04, 0x9b05, 0x9b29, 0x9b35, 0x9b4a, 0x9b4c,
01682 0x9b4b, 0x9bc7, 0x9bc6, 0x9bc3, 0x9bbf, 0x9bc1, 0x9bb5, 0x9bb8,
01683 0x9bd3, 0x9bb6, 0x9bc4, 0x9bb9, 0x9bbd, 0x9d5c, 0x9d53, 0x9d4f,
01684 0x9d4a, 0x9d5b, 0x9d4b, 0x9d59, 0x9d56, 0x9d4c, 0x9d5f, 0x9d52,
01685 0x9d54, 0x9d5f, 0x9d58, 0x9d5a, 0x9e8e, 0x9e8c, 0x9edf, 0x9f01,
01686 0x9f00, 0x9f16, 0x9f25, 0x9f2b, 0x9f2a, 0x9f29, 0x9f28, 0x9f4c,
01687 0x9f55, 0x5134, 0x5135, 0x5296, 0x52f7, 0x53b4, 0x56ab, 0x56ad,
01688 0x56a6, 0x56a7, 0x56aa, 0x56ac, 0x58da, 0x58dd, 0x58db, 0x5912,
01689 0x5b3d, 0x5b3e, 0x5b3f, 0x5dc3, 0x5e70,
01690 /* 0xf2 */
01691 0x5fbf, 0x61fb, 0x6507, 0x6510, 0x650d, 0x6509, 0x650c, 0x650e,
01692 0x6584, 0x65de, 0x65dd, 0x66de, 0x6ae7, 0x6ae0, 0x6acc, 0x6ad1,
01693 0x6ad9, 0x6acb, 0x6adf, 0x6adc, 0x6ad0, 0x6aeb, 0x6acf, 0x6acd,
01694 0x6ade, 0x6b60, 0x6bb0, 0x6c0c, 0x7019, 0x7027, 0x7020, 0x7016,
01695 0x702b, 0x7021, 0x7022, 0x7023, 0x7029, 0x7017, 0x7024, 0x701c,
01696 0x702a, 0x720c, 0x720a, 0x7207, 0x7202, 0x7205, 0x72a5, 0x72a6,
01697 0x72a4, 0x72a3, 0x72a1, 0x74cb, 0x74c5, 0x74b7, 0x74c3, 0x7516,
01698 0x7660, 0x77c9, 0x77ca, 0x77c4, 0x77f1, 0x791d, 0x791b, 0x7921,
01699 0x791c, 0x7917, 0x791e, 0x79b0, 0x7a67, 0x7a68, 0x7c33, 0x7c3c,
01700 0x7c39, 0x7c2c, 0x7c3b, 0x7cec, 0x7cea, 0x7e76, 0x7e75, 0x7e78,
01701 0x7e70, 0x7e77, 0x7e6f, 0x7e7a, 0x7e72, 0x7e74, 0x7e68, 0x7f4b,
01702 0x7f4a, 0x7f83, 0x7f86, 0x7fb7, 0x7ffd, 0x7ffe, 0x8078, 0x81d7,
01703 0x81d5, 0x8264, 0x8261, 0x8263, 0x85eb, 0x85f1, 0x85ed, 0x85d9,
01704 0x85e1, 0x85e8, 0x85da, 0x85d7, 0x85ec, 0x85f2, 0x85f8, 0x85d8,
01705 0x85df, 0x85e3, 0x85dc, 0x85d1, 0x85f0, 0x85e6, 0x85ef, 0x85de,
01706 0x85e2, 0x8800, 0x87fa, 0x8803, 0x87f6, 0x87f7, 0x8809, 0x880c,
01707 0x880b, 0x8806, 0x87fc, 0x8808, 0x87ff, 0x880a, 0x8802, 0x8962,
01708 0x895a, 0x895b, 0x8957, 0x8961, 0x895c, 0x8958, 0x895d, 0x8959,
01709 0x8988, 0x89b7, 0x89b6, 0x89f6, 0x8b50, 0x8b48, 0x8b4a, 0x8b40,
01710 0x8b53, 0x8b56, 0x8b54, 0x8b4b, 0x8b55,
01711 /* 0xf3 */
01712 0x8b51, 0x8b42, 0x8b52, 0x8b57, 0x8c43, 0x8c77, 0x8c76, 0x8c9a,
01713 0x8d06, 0x8d07, 0x8d09, 0x8dac, 0x8daa, 0x8dad, 0x8dab, 0x8e6d,
01714 0x8e78, 0x8e73, 0x8e6a, 0x8e6f, 0x8e7b, 0x8ec2, 0x8f52, 0x8f51,
01715 0x8f4f, 0x8f53, 0x8fb4, 0x9140, 0x913f, 0x91b0, 0x91ad, 0x91ad,
01716 0x93de, 0x93c7, 0x93cf, 0x93c2, 0x93da, 0x93d0, 0x93f9, 0x93ec,
01717 0x93cc, 0x93d9, 0x93a9, 0x93e6, 0x93ca, 0x93d4, 0x93ee, 0x93e3,
01718 0x93d5, 0x93c4, 0x93ce, 0x93c0, 0x93d2, 0x93e7, 0x957d, 0x95da,
01719 0x95db, 0x96e1, 0x9729, 0x972b, 0x972c, 0x9728, 0x9726, 0x97b3,
01720 0x97b7, 0x97b6, 0x97dd, 0x97de, 0x97df, 0x985c, 0x9859, 0x985d,
01721 0x9857, 0x98bf, 0x98bd, 0x98bb, 0x98be, 0x9948, 0x9947, 0x9943,
01722 0x99a6, 0x99a7, 0x9a1a, 0x9a15, 0x9a25, 0x9a1d, 0x9a24, 0x9a1b,
01723 0x9a22, 0x9a20, 0x9a27, 0x9a23, 0x9a1e, 0x9a1c, 0x9a14, 0x9ac2,
01724 0x9b0b, 0x9b0a, 0x9b0e, 0x9b0c, 0x9b37, 0x9bea, 0x9beb, 0x9be0,
01725 0x9bde, 0x9be4, 0x9be6, 0x9be2, 0x9bf0, 0x9bd4, 0x9bd7, 0x9bec,
01726 0x9bdc, 0x9bd9, 0x9be5, 0x9bd5, 0x9be1, 0x9bda, 0x9d77, 0x9d81,
01727 0x9d8a, 0x9d84, 0x9d88, 0x9d71, 0x9d80, 0x9d78, 0x9d86, 0x9d8b,
01728 0x9d8c, 0x9d7d, 0x9d6b, 0x9d74, 0x9d75, 0x9d70, 0x9d69, 0x9d85,
01729 0x9d73, 0x9d7b, 0x9d82, 0x9d6f, 0x9d79, 0x9d7f, 0x9d87, 0x9d68,
01730 0x9e94, 0x9e91, 0x9ec0, 0x9efc, 0x9f2d, 0x9f40, 0x9f41, 0x9f4d,
01731 0x9f56, 0x9f57, 0x9f58, 0x5337, 0x56b2,
01732 /* 0xf4 */
01733 0x56b5, 0x56b3, 0x58e3, 0x5b45, 0x5dc6, 0x5dc7, 0x5eee, 0x5eef,
01734 0x5fc0, 0x5fc1, 0x61f9, 0x6517, 0x6516, 0x6515, 0x6513, 0x65df,
01735 0x66e8, 0x66e3, 0x66e4, 0x6af3, 0x6af0, 0x6aea, 0x6ae8, 0x6af9,
01736 0x6af1, 0x6aee, 0x6aef, 0x703c, 0x7035, 0x702f, 0x7037, 0x7034,
01737 0x7031, 0x7042, 0x7038, 0x703f, 0x703a, 0x7039, 0x7040, 0x703b,
01738 0x7033, 0x7041, 0x7213, 0x7214, 0x72a8, 0x737d, 0x737c, 0x74ba,
01739 0x76ab, 0x76aa, 0x76be, 0x77cc, 0x77ce, 0x77cf, 0x77cd, 0x77cd,
01740 0x77f2, 0x7925, 0x7923, 0x7927, 0x7928, 0x7924, 0x7929, 0x79b2,
01741 0x7a6e, 0x7a6c, 0x7a6d, 0x7af7, 0x7c49, 0x7c48, 0x7c4a, 0x7c47,
01742 0x7c45, 0x7cee, 0x7e7b, 0x7e7e, 0x7e81, 0x7e80, 0x7fba, 0x7fff,
01743 0x8079, 0x81db, 0x81d9, 0x820b, 0x8268, 0x8269, 0x8622, 0x85ff,
01744 0x8601, 0x85fe, 0x861b, 0x8600, 0x85f6, 0x8604, 0x8609, 0x8605,
01745 0x860c, 0x85fd, 0x8819, 0x8810, 0x8811, 0x8817, 0x8813, 0x8816,
01746 0x8963, 0x8966, 0x89b9, 0x89f7, 0x8b60, 0x8b6a, 0x8b5d, 0x8b68,
01747 0x8b63, 0x8b65, 0x8b67, 0x8b6d, 0x8dae, 0x8e86, 0x8e88, 0x8e84,
01748 0x8f59, 0x8f56, 0x8f57, 0x8f55, 0x8f58, 0x8f5a, 0x908d, 0x9143,
01749 0x9141, 0x91b7, 0x91b5, 0x91b2, 0x91b3, 0x940b, 0x9413, 0x93fb,
01750 0x9420, 0x940f, 0x9414, 0x93fe, 0x9415, 0x9410, 0x9428, 0x9419,
01751 0x940d, 0x93f5, 0x9400, 0x93f7, 0x9407, 0x940e, 0x9416, 0x9412,
01752 0x93fa, 0x9409, 0x93f8, 0x940a, 0x93ff,
01753 /* 0xf5 */
01754 0x93fc, 0x940c, 0x93f6, 0x9411, 0x9406, 0x95de, 0x95e0, 0x95df,
01755 0x972e, 0x972f, 0x97b9, 0x97bb, 0x97fd, 0x97fe, 0x9860, 0x9862,
01756 0x9863, 0x985f, 0x98c1, 0x98c2, 0x9950, 0x994e, 0x9959, 0x994c,
01757 0x994b, 0x9953, 0x9a32, 0x9a34, 0x9a31, 0x9a2c, 0x9a2a, 0x9a36,
01758 0x9a29, 0x9a2e, 0x9a38, 0x9a2d, 0x9ac7, 0x9ac4, 0x9ac6, 0x9b10,
01759 0x9b12, 0x9b11, 0x9c0b, 0x9c08, 0x9bf7, 0x9c05, 0x9c12, 0x9bf8,
01760 0x9c40, 0x9c07, 0x9c0e, 0x9c06, 0x9c17, 0x9c14, 0x9c09, 0x9d9f,
01761 0x9d99, 0x9da4, 0x9d9d, 0x9d92, 0x9d98, 0x9d90, 0x9d9b, 0x9da0,
01762 0x9d94, 0x9d9c, 0x9daa, 0x9d97, 0x9da1, 0x9d9a, 0x9da2, 0x9da8,
01763 0x9d9e, 0x9da3, 0x9dbf, 0x9da9, 0x9d96, 0x9da6, 0x9da7, 0x9e99,
01764 0x9e9b, 0x9e9a, 0x9ee5, 0x9ee4, 0x9ee7, 0x9ee6, 0x9f30, 0x9f2e,
01765 0x9f5b, 0x9f60, 0x9f5e, 0x9f5d, 0x9f59, 0x9f91, 0x513a, 0x5139,
01766 0x5298, 0x5297, 0x56c3, 0x56bd, 0x56be, 0x5b48, 0x5b47, 0x5dcb,

```

```
01767 0x5dcf, 0x5ef1, 0x61fd, 0x651b, 0x6b02, 0x6afc, 0x6b03, 0x6af8,
01768 0x6b00, 0x7043, 0x7044, 0x704a, 0x7048, 0x7049, 0x7045, 0x7046,
01769 0x721d, 0x721a, 0x7219, 0x737e, 0x7517, 0x766a, 0x77d0, 0x792d,
01770 0x7931, 0x792f, 0x7c54, 0x7c53, 0x7cf2, 0x7e8a, 0x7e87, 0x7e88,
01771 0x7e8b, 0x7e86, 0x7e8d, 0x7f4d, 0x7fbb, 0x8030, 0x81dd, 0x8618,
01772 0x862a, 0x8626, 0x861f, 0x8623, 0x861c, 0x8619, 0x8627, 0x862e,
01773 0x8621, 0x8620, 0x8629, 0x861e, 0x8625,
01774 /* 0xf6 */
01775 0x8829, 0x881d, 0x881b, 0x8820, 0x8824, 0x881c, 0x882b, 0x884a,
01776 0x896d, 0x8969, 0x896e, 0x896b, 0x89fa, 0x8b79, 0x8b78, 0x8b45,
01777 0x8b7a, 0x8b7b, 0x8d10, 0x8d14, 0x8daf, 0x8e8e, 0x8e8c, 0x8f5e,
01778 0x8f5b, 0x8f5d, 0x9146, 0x9144, 0x9145, 0x91b9, 0x943f, 0x943b,
01779 0x9436, 0x9429, 0x943d, 0x943c, 0x9430, 0x9439, 0x942a, 0x9437,
01780 0x942c, 0x9440, 0x9431, 0x95e5, 0x95e4, 0x95e3, 0x9735, 0x973a,
01781 0x97bf, 0x97e1, 0x9864, 0x98c9, 0x98c6, 0x98c0, 0x9958, 0x9956,
01782 0x9a39, 0x9a3d, 0x9a46, 0x9a44, 0x9a42, 0x9a41, 0x9a3a, 0x9a3f,
01783 0x9acd, 0x9b15, 0x9b17, 0x9b18, 0x9b16, 0x9b3a, 0x9b52, 0x9c2b,
01784 0x9c1d, 0x9c1c, 0x9c1e, 0x9c2c, 0x9c23, 0x9c28, 0x9c29, 0x9c24, 0x9c21,
01785 0x9db7, 0x9db6, 0x9dbc, 0x9dc1, 0x9dc7, 0x9dca, 0x9dcf, 0x9dbe,
01786 0x9dc5, 0x9dc3, 0x9dbb, 0x9db5, 0x9dce, 0x9db9, 0x9dba, 0x9dac,
01787 0x9dc8, 0x9db1, 0x9dad, 0x9dcc, 0x9db3, 0x9dcd, 0x9db2, 0x9e7a,
01788 0x9e9c, 0x9eeb, 0x9eee, 0x9eed, 0x9f1b, 0x9f18, 0x9f1a, 0x9f31,
01789 0x9f4e, 0x9f65, 0x9f64, 0x9f92, 0x4eb9, 0x56c6, 0x56c5, 0x56cb,
01790 0x5971, 0x5b4b, 0x5b4c, 0x5d4c, 0x5dd1, 0x5ef2, 0x5521, 0x6520,
01791 0x6526, 0x6522, 0x6b0b, 0x6b08, 0x6b09, 0x6c0d, 0x7055, 0x7056,
01792 0x7057, 0x7052, 0x721e, 0x721f, 0x72a9, 0x737f, 0x74d8, 0x74d5,
01793 0x74d9, 0x74d7, 0x766d, 0x76ad, 0x7935, 0x79b4, 0x7a70, 0x7a71,
01794 0x7c57, 0x7c5c, 0x7c59, 0x7c5b, 0x7c5a,
01795 /* 0xf7 */
01796 0x7cf4, 0x7cf1, 0x7e91, 0x7f4f, 0x7f87, 0x81de, 0x826b, 0x8634,
01797 0x8635, 0x8633, 0x862c, 0x8632, 0x8636, 0x882c, 0x8828, 0x8826,
01798 0x882a, 0x8825, 0x8971, 0x89bf, 0x89be, 0x89fb, 0x8b7e, 0x8b84,
01799 0x8b82, 0x8b86, 0x8b85, 0x8b7f, 0x8d15, 0x8e95, 0x8e94, 0x8e9a,
01800 0x8e92, 0x8e90, 0x8e96, 0x8e97, 0x8f60, 0x8f62, 0x9147, 0x944c,
01801 0x9450, 0x944a, 0x944b, 0x944f, 0x9447, 0x9445, 0x9448, 0x9449,
01802 0x9446, 0x973f, 0x97e3, 0x986a, 0x9869, 0x98cb, 0x9954, 0x995b,
01803 0x9a4e, 0x9a53, 0x9a54, 0x9a4c, 0x9a4f, 0x9a48, 0x9a4a, 0x9a49,
01804 0x9a52, 0x9a50, 0x9ad0, 0x9b19, 0x9b2b, 0x9b3b, 0x9b56, 0x9b55,
01805 0x9c46, 0x9c45, 0x9c3f, 0x9c3d, 0x9c39, 0x9c33, 0x9c41, 0x9c3c,
01806 0x9c37, 0x9c34, 0x9c32, 0x9c3d, 0x9c36, 0x9ddb, 0x9dd2, 0x9dde,
01807 0x9dda, 0x9dcb, 0x9dd0, 0x9ddc, 0x9dd1, 0x9ddf, 0x9de9, 0x9dd9,
01808 0x9dd8, 0x9dd6, 0x9ddf5, 0x9ddd5, 0x9ddd, 0x9eb6, 0x9ef0, 0x9f35,
01809 0x9f33, 0x9f32, 0x9f42, 0x9f6b, 0x9f95, 0x9fa2, 0x513d, 0x5299,
01810 0x58e8, 0x58e7, 0x5972, 0x5b4d, 0x5dd8, 0x882f, 0x5f4f, 0x6201,
01811 0x6203, 0x6204, 0x6529, 0x6525, 0x6596, 0x66eb, 0x6b11, 0x6b12,
01812 0x6b0f, 0x6bca, 0x705b, 0x705a, 0x7222, 0x7382, 0x7381, 0x7383,
01813 0x7670, 0x77d4, 0x7c67, 0x7c66, 0x7e95, 0x826c, 0x863a, 0x8640,
01814 0x8639, 0x863c, 0x8631, 0x863b, 0x863e, 0x8830, 0x8832, 0x882e,
01815 0x8833, 0x8976, 0x8974, 0x8973, 0x89fe,
01816 /* 0xf8 */
01817 0x8b8c, 0x8b8e, 0x8b8b, 0x8b88, 0x8c45, 0x8d19, 0x8e98, 0x8f64,
01818 0x8f63, 0x91bc, 0x9462, 0x9455, 0x945d, 0x9457, 0x945e, 0x97c4,
01819 0x97c5, 0x9800, 0x9a56, 0x9a59, 0x9b1e, 0x9b1f, 0x9b20, 0x9c52,
01820 0x9c58, 0x9c50, 0x9c4a, 0x9c4d, 0x9c4b, 0x9c55, 0x9c59, 0x9c4c,
01821 0x9c4e, 0x9dfb, 0x9df7, 0x9def, 0x9de3, 0x9deb, 0x9df8, 0x9de4,
01822 0x9df6, 0x9del, 0x9dee, 0x9de6, 0x9df2, 0x9df0, 0x9de2, 0x9dec,
01823 0x9df4, 0x9dff, 0x9de8, 0x9ded, 0x9ec2, 0x9ed0, 0x9ef2, 0x9ef3,
01824 0x9f06, 0x9f1c, 0x9f38, 0x9f37, 0x9f36, 0x9f43, 0x9f4f, 0x9f71,
01825 0x9f70, 0x9f6e, 0x9f6f, 0x56d3, 0x56cd, 0x5b4e, 0x5c6d, 0x652d,
01826 0x66ed, 0x66ee, 0x6b13, 0x705f, 0x7061, 0x705d, 0x7060, 0x7223,
01827 0x74db, 0x74e5, 0x77d5, 0x7938, 0x79b7, 0x79b6, 0x7c6a, 0x7e97,
01828 0x7f89, 0x826d, 0x8643, 0x8838, 0x8837, 0x8835, 0x884b, 0x8b94,
01829 0x8b95, 0x8e9e, 0x8e9f, 0x8ea0, 0x8e9d, 0x91be, 0x91bd, 0x91c2,
01830 0x946b, 0x9468, 0x9469, 0x96e5, 0x9746, 0x9743, 0x9747, 0x97c7,
01831 0x97e5, 0x9a5e, 0x9ad5, 0x9b59, 0x9c63, 0x9c67, 0x9c66, 0x9c62,
01832 0x9c5e, 0x9c60, 0x9e02, 0x9dfe, 0x9e07, 0x9e03, 0x9e06, 0x9e05,
01833 0x9e00, 0x9e01, 0x9e09, 0x9dff, 0x9dfd, 0x9e04, 0x9ea0, 0x9f1e,
01834 0x9f46, 0x9f74, 0x9f75, 0x9f76, 0x56d4, 0x652e, 0x65b8, 0x6b18,
01835 0x6b19, 0x6b17, 0x6b1a, 0x7062, 0x7226, 0x72aa, 0x77d8, 0x77d9,
01836 0x7939, 0x7c69, 0x7c6b, 0x7cf6, 0x7e9a,
01837 /* 0xf9 */
01838 0x7e98, 0x7e9b, 0x7e99, 0x81e0, 0x81e1, 0x8646, 0x8647, 0x8648,
01839 0x8979, 0x897a, 0x897c, 0x897b, 0x89ff, 0x8b98, 0x8b99, 0x8ea5,
01840 0x8ea4, 0x8ea3, 0x946e, 0x946d, 0x946f, 0x9471, 0x9473, 0x9749,
01841 0x9872, 0x995f, 0x9c68, 0x9c6e, 0x9c6d, 0x9e0b, 0x9e0d, 0x9e10,
01842 0x9e0f, 0x9e12, 0x9e11, 0x9ea1, 0x9ef5, 0x9f09, 0x9f47, 0x9f78,
01843 0x9f7b, 0x9f7a, 0x9f79, 0x571e, 0x7066, 0x7c6f, 0x883c, 0x8db2,
01844 0x8ea6, 0x91c3, 0x9474, 0x9478, 0x9476, 0x9475, 0x9a60, 0x9c74,
01845 0x9c73, 0x9c71, 0x9c75, 0x9e14, 0x9e13, 0x9ef6, 0x9f0a, 0x9fa4,
01846 0x7068, 0x7065, 0x7cf7, 0x866a, 0x883e, 0x883d, 0x883f, 0x8b9e,
01847 0x8c9c, 0x8ea9, 0x8ec9, 0x974b, 0x9873, 0x9874, 0x98cc, 0x9961,
01848 0x99ab, 0x9a64, 0x9a66, 0x9a67, 0x9b24, 0x9e15, 0x9e17, 0x9f48,
01849 0x6207, 0x6b1e, 0x7227, 0x864c, 0x8ea8, 0x9482, 0x9480, 0x9481,
01850 0x9a69, 0x9a68, 0x9b2e, 0x9e19, 0x7229, 0x864b, 0x8b9f, 0x9483,
01851 0x9c79, 0x9eb7, 0x7675, 0x9a6b, 0x9c7a, 0x9e1d, 0x7069, 0x706a,
01852 0x9ea4, 0x9f7e, 0x9f49, 0x9f98,
01853 ;
```

```
01854
01855 static int
01856 big5_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
01857 {
01858     unsigned char c1 = s[0];
01859     if ((c1 >= 0xa1 && c1 <= 0xc7) || (c1 >= 0xc9 && c1 <= 0xf9)) {
01860         if (n >= 2) {
01861             unsigned char c2 = s[1];
01862             if ((c2 >= 0x40 && c2 < 0x7f) || (c2 >= 0xa1 && c2 < 0xff)) {
01863                 unsigned int i = 157 * (c1 - 0xa1) + (c2 - (c2 >= 0xa1 ? 0x62 : 0x40));
01864                 unsigned short wc = 0xffffd;
01865                 if (i < 6280) {
01866                     if (i < 6121)
01867                         wc = big5_2uni_pagea1[i];
01868                     } else {
01869                         if (i < 13932)
01870                             wc = big5_2uni_pagec9[i-6280];
01871                     }
01872                     if (wc != 0xffffd) {
01873                         *pwc = (ucs4_t) wc;
01874                         return 2;
01875                     }
01876                 }
01877                 return RET_ILSEQ;
01878             }
01879             return RET_TOOFEW(0);
01880         }
01881         return RET_ILSEQ;
01882     }
01883     #endif /* NEED_TOWC */
01884
01885     #ifdef NEED_TOMB
01886     static const unsigned short big5_2charset[13703] = {
01887         0xa246, 0xa247, 0xa244, 0xa1b1, 0xa258, 0xa1d3, 0xa150, 0xa1d1,
01888         0xa1d2, 0xa3be, 0xa3bc, 0xa3bd, 0xa3bf, 0xa3bb, 0xa344, 0xa345,
01889         0xa346, 0xa347, 0xa348, 0xa349, 0xa34a, 0xa34b, 0xa34c, 0xa34d,
01890         0xa34e, 0xa34f, 0xa350, 0xa351, 0xa352, 0xa353, 0xa354, 0xa355,
01891         0xa356, 0xa357, 0xa358, 0xa359, 0xa35a, 0xa35b, 0xa35c, 0xa35d,
01892         0xa35e, 0xa35f, 0xa360, 0xa361, 0xa362, 0xa363, 0xa364, 0xa365,
01893         0xa366, 0xa367, 0xa368, 0xa369, 0xa36a, 0xa36b, 0xa36c, 0xa36d,
01894         0xa36e, 0xa36f, 0xa370, 0xa371, 0xa372, 0xa373, 0xc7b3, 0xc7b1,
01895         0xc7b2, 0xc7b4, 0xc7b5, 0xc7b6, 0xc7b7, 0xc7b8, 0xc7b9, 0xc7ba,
01896         0xc7bb, 0xc7bc, 0xc7bd, 0xc7be, 0xc7bf, 0xc7c0, 0xc7c1, 0xc7c2,
01897         0xc7c3, 0xc7c4, 0xc7c5, 0xc7c6, 0xc7c7, 0xc7c8, 0xc7c9, 0xc7ca,
01898         0xc7cb, 0xc7cc, 0xc7cd, 0xc7ce, 0xc7cf, 0xc7d0, 0xc7d1, 0xc7d2, 0xc7d3,
01899         0xc7d4, 0xc7d5, 0xc7d6, 0xc7d7, 0xc7d8, 0xc7d9, 0xc7da, 0xc7db,
01900         0xc7dc, 0xc7dd, 0xc7de, 0xc7df, 0xc7e0, 0xc7e1, 0xc7e2, 0xc7e3,
01901         0xc7e4, 0xc7e5, 0xc7e6, 0xc7e7, 0xc7e8, 0xc7e9, 0xa156, 0xa158,
01902         0xa1a5, 0xa1a6, 0xa1a7, 0xa1a8, 0xa1a9, 0xa145, 0xa14c, 0xa14b, 0xa1ac,
01903         0xa1ab, 0xa1b0, 0xa1c2, 0xa24a, 0xa1c1, 0xa24b, 0xa2b9, 0xa2ba,
01904         0xa2bb, 0xa2bc, 0xa2bd, 0xa2be, 0xa2bf, 0xa2c0, 0xa2c1, 0xa2c2,
01905         0xa1f6, 0xa1f4, 0xa1f7, 0xa1f5, 0xa1f8, 0xa1f9, 0xa1fb, 0xa1fa,
01906         0xa1d4, 0xa1db, 0xa1e8, 0xa1e7, 0xa1fd, 0xa1fc, 0xa1e4, 0xa1e5,
01907         0xa1ec, 0xa1ed, 0xa1ef, 0xa1ee, 0xa1e3, 0xa1dc, 0xa1da, 0xa1dd,
01908         0xa1d8, 0xa1d9, 0xa1e6, 0xa1e9, 0xc7e9, 0xc7ea, 0xc7eb, 0xc7ec,
01909         0xc7ed, 0xc7ee, 0xc7ef, 0xc7f0, 0xc7f1, 0xc7f2, 0xc7f3, 0xc7f4,
01910         0xc7f5, 0xc7f6, 0xc7f7, 0xc7f8, 0xc7f9, 0xc7fa, 0xc7fb, 0xc7fc,
01911         0xa277, 0xa278, 0xa27a, 0xa27b, 0xa27c, 0xa27d, 0xa275, 0xa274,
01912         0xa273, 0xa272, 0xa271, 0xa2a4, 0xa2a5, 0xa2a7, 0xa2a6, 0xa27e,
01913         0xa2a1, 0xa2a3, 0xa2a2, 0xa2ac, 0xa2ad, 0xa2ae, 0xa262, 0xa263,
01914         0xa264, 0xa265, 0xa266, 0xa267, 0xa268, 0xa269, 0xa270, 0xa26f,
01915         0xa26e, 0xa26d, 0xa26c, 0xa26b, 0xa26a, 0xa276, 0xa279, 0xa1bd,
01916         0xa1bc, 0xa1b6, 0xa1b5, 0xa1bf, 0xa1be, 0xa1bb, 0xa1ba, 0xa1b3,
01917         0xa1b7, 0xa1b4, 0xa2a8, 0xa2a9, 0xa2ab, 0xa2aa, 0xa1b9, 0xa1b8,
01918         0xa1f3, 0xa1f0, 0xa1f2, 0xa1f1, 0xa140, 0xa142, 0xa143, 0xa1b2,
01919         0xc6a4, 0xa171, 0xa172, 0xa16d, 0xa16e, 0xa175, 0xa176, 0xa179,
01920         0xa17a, 0xa169, 0xa16a, 0xa245, 0xa165, 0xa166, 0xa1a9, 0xa1aa,
01921         0xa2c3, 0xa2c4, 0xa2c5, 0xa2c6, 0xa2c7, 0xa2c8, 0xa2c9, 0xa2ca,
01922         0xa2cb, 0xc6a5, 0xc6a6, 0xc6a7, 0xc6a8, 0xc6a9, 0xc6aa, 0xc6ab,
01923         0xc6ac, 0xc6ad, 0xc6ae, 0xc6af, 0xc6b0, 0xc6b1, 0xc6b2, 0xc6b3,
01924         0xc6ba, 0xc6b5, 0xc6b6, 0xc6b7, 0xc6b8, 0xc6b9, 0xc6ba, 0xc6bb,
01925         0xc6bc, 0xc6bd, 0xc6be, 0xc6bf, 0xc6c0, 0xc6c1, 0xc6c2, 0xc6c3,
01926         0xc6c4, 0xc6c5, 0xc6c6, 0xc6c7, 0xc6c8, 0xc6c9, 0xc6ca, 0xc6cb,
01927         0xc6cc, 0xc6cd, 0xc6ce, 0xc6cf, 0xc6d0, 0xc6d1, 0xc6d2, 0xc6d3,
01928         0xc6d4, 0xc6d5, 0xc6d6, 0xc6d7, 0xc6d8, 0xc6d9, 0xc6da, 0xc6db,
01929         0xc6dc, 0xc6dd, 0xc6de, 0xc6df, 0xc6e0, 0xc6e1, 0xc6e2, 0xc6e3,
01930         0xc6e4, 0xc6e5, 0xc6e6, 0xc6e7, 0xc6e8, 0xc6e9, 0xc6ea, 0xc6eb,
01931         0xc6ec, 0xc6ed, 0xc6ee, 0xc6ef, 0xc6f0, 0xc6f1, 0xc6f2, 0xc6f3,
01932         0xc6f4, 0xc6f5, 0xc6f6, 0xc6f7, 0xc6a2, 0xc6a3, 0xc6f8, 0xc6f9,
01933         0xc6fa, 0xc6fb, 0xc6fc, 0xc6fd, 0xc6fe, 0xc740, 0xc741, 0xc742,
01934         0xc743, 0xc744, 0xc745, 0xc746, 0xc747, 0xc748, 0xc749, 0xc74a,
01935         0xc74b, 0xc74c, 0xc74d, 0xc74e, 0xc74f, 0xc750, 0xc751, 0xc752,
01936         0xc753, 0xc754, 0xc755, 0xc756, 0xc757, 0xc758, 0xc759, 0xc75a,
01937         0xc75b, 0xc75c, 0xc75d, 0xc75e, 0xc75f, 0xc760, 0xc761, 0xc762,
01938         0xc763, 0xc764, 0xc765, 0xc766, 0xc767, 0xc768, 0xc769, 0xc76a,
01939         0xc76b, 0xc76c, 0xc76d, 0xc76e, 0xc76f, 0xc770, 0xc771, 0xc772,
01940         0xc773, 0xc774, 0xc775, 0xc776, 0xc777, 0xc778, 0xc779, 0xc77a,
```

01941 0xc77b, 0xc77c, 0xc77d, 0xc77e, 0xc7a1, 0xc7a2, 0xc7a3, 0xc7a4,
01942 0xc7a5, 0xc7a6, 0xc7a7, 0xc7a8, 0xc7a9, 0xc7aa, 0xc7ab, 0xc7ac,
01943 0xc7ad, 0xc7ae, 0xc7af, 0xc7b0, 0xc6a1, 0xa374, 0xa375, 0xa376,
01944 0xa377, 0xa378, 0xa379, 0xa37a, 0xa37b, 0xa37c, 0xa37d, 0xa37e,
01945 0xa3a1, 0xa3a2, 0xa3a3, 0xa3a4, 0xa3a5, 0xa3a6, 0xa3a7, 0xa3a8,
01946 0xa3a9, 0xa3aa, 0xa3ab, 0xa3ac, 0xa3ad, 0xa3ae, 0xa3af, 0xa3b0,
01947 0xa3b1, 0xa3b2, 0xa3b3, 0xa3b4, 0xa3b5, 0xa3b6, 0xa3b7, 0xa3b8,
01948 0xa3b9, 0xa3ba, 0xa1c0, 0xa255, 0xa256, 0xa250, 0xa251, 0xa252,
01949 0xa254, 0xa257, 0xa253, 0xa1eb, 0xa1ea, 0xa24f, 0xa440, 0xa442,
01950 0xa443, 0xc945, 0xa456, 0xa454, 0xa457, 0xa455, 0xc946, 0xa4a3,
01951 0xc94f, 0xc94d, 0xa4a2, 0xa4a1, 0xa542, 0xa541, 0xa540, 0xa543,
01952 0xa4fe, 0xa5e0, 0xa5e1, 0xa8c3, 0xa458, 0xa4a4, 0xc950, 0xa4a5,
01953 0xc963, 0xa6ea, 0xcbb1, 0xa459, 0xa4a6, 0xa544, 0xc964, 0xc940,
01954 0xa444, 0xa45b, 0xc947, 0xa45c, 0xa4a7, 0xa545, 0xa547, 0xa546,
01955 0xa5e2, 0xa5e3, 0xa5e4, 0xa8c4, 0xadbc, 0xa441, 0xc941, 0xa445, 0xa45e,
01956 0xa45d, 0xa5e4, 0xa8c5, 0xb0ae, 0xd44b, 0xb6c3, 0xdcbl, 0xdcbl2,
01957 0xa446, 0xa4a9, 0xa8c6, 0xa447, 0xc948, 0xa45f, 0xa4aa, 0xa4ac,
01958 0xc951, 0xa4ad, 0xa4ab, 0xa5e5, 0xa8c7, 0xa8c8, 0xab45, 0xa460,
01959 0xa4ae, 0xa5e6, 0xa5e8, 0xa5e7, 0xa6eb, 0xa8c9, 0xa8ca, 0xab46,
01960 0xab47, 0xadbb, 0xdcbl3, 0xf6d6, 0xa448, 0xa4b0, 0xa4af, 0xc952,
01961 0xa4b1, 0xa4b7, 0xa4b2, 0xa4b3, 0xc954, 0xc953, 0xa4b5, 0xa4b6,
01962 0xa4b4, 0xa54a, 0xa54b, 0xa54c, 0xa54d, 0xa549, 0xa550, 0xc96a,
01963 0xc966, 0xc969, 0xa551, 0xa561, 0xc968, 0xa54e, 0xa54f, 0xa548,
01964 0xc965, 0xc967, 0xa5f5, 0xc9b0, 0xa5f2, 0xa5f6, 0xc9ba, 0xc9ae,
01965 0xa5f3, 0xc9b2, 0xa5f4, 0xa5f7, 0xa5e9, 0xc9b1, 0xa5f8, 0xc9b5,
01966 0xc9b9, 0xc9b6, 0xc9b3, 0xa5ea, 0xa5ec, 0xa5f9, 0xa5ee, 0xc9ab,
01967 0xa5f1, 0xa5ef, 0xa5f0, 0xc9bb, 0xc9b8, 0xc9af, 0xa5ed, 0xc9ac,
01968 0xa5eb, 0xc9b4, 0xc9b7, 0xc9ad, 0xca66, 0xa742, 0xa6f4, 0xca67,
01969 0xa6f1, 0xa744, 0xa6f9, 0xa6f8, 0xca5b, 0xa6fc, 0xa6f7, 0xca60,
01970 0xca68, 0xca64, 0xca6a, 0xa6fa, 0xa6fd, 0xa6ee, 0xa747, 0xca5d, 0xcbbd,
01971 0xa6ec, 0xa743, 0xa6ed, 0xa6f5, 0xa6f6, 0xca62, 0xca5e, 0xa6fb,
01972 0xa6f3, 0xca5a, 0xa6ef, 0xca65, 0xa745, 0xa748, 0xa6f2, 0xa740,
01973 0xa746, 0xa6f0, 0xca63, 0xa741, 0xca69, 0xca5c, 0xa6fe, 0xca5f,
01974 0xca61, 0xa8d8, 0xcbbf, 0xcbbcb, 0xa8d0, 0xcbbcc, 0xa8cb, 0xa8d5,
01975 0xa8ce, 0xcbb9, 0xa8d6, 0xcbb8, 0xcbbc, 0xcbbc3, 0xcbbc1, 0xa8de,
01976 0xa8d9, 0xcbb3, 0xcbb5, 0xa8db, 0xa8cf, 0xcbb6, 0xcbbc2, 0xcbbc9,
01977 0xa8d4, 0xcbbb, 0xcbb4, 0xa8d3, 0xcbb7, 0xa8d7, 0xcbbba, 0xa8d2,
01978 0xa8cd, 0xa8dc, 0xcbbc4, 0xa8dd, 0xcbbc8, 0xcbbc6, 0xcbbca, 0xa8da,
01979 0xcbbbe, 0xcbb2, 0xcbbc0, 0xa8d1, 0xcbbc5, 0xa8cc, 0xcbbc7, 0xab56,
01980 0xab4a, 0xcde0, 0xcde8, 0xab49, 0xab51, 0xab5d, 0xcdee, 0xcdec,
01981 0xcde7, 0xab4b, 0xcded, 0xcde3, 0xab59, 0xab50, 0xab58, 0xcdde,
01982 0xcdea, 0xcde1, 0xab54, 0xcde2, 0xcddd, 0xab5b, 0xab4e, 0xab57,
01983 0xab4d, 0xcddf, 0xcde4, 0xcdeb, 0xab55, 0xab52, 0xcde6, 0xab5a,
01984 0xcde9, 0xcde5, 0xab4f, 0xab5c, 0xab53, 0xab4c, 0xab48, 0xcdef,
01985 0xadd7, 0xadd1, 0xadd6, 0xd0d0, 0xd0cf, 0xd0d4, 0xd0d5,
01986 0xadc4, 0xadc, 0xadda, 0xadce, 0xd0c9, 0xad7, 0xd0ca, 0xaddc,
01987 0xadd3, 0xadbe, 0xadbf, 0xd0dd, 0xb0bf, 0xadcc, 0xadcb, 0xd0cb,
01988 0xadcf, 0xad5b, 0xad6, 0xd0d6, 0xadd5, 0xadd4, 0xadca, 0xd0ce,
01989 0xd0d7, 0xd0c8, 0xadc9, 0xd0d8, 0xadd2, 0xd0cc, 0xadc0, 0xadc3,
01990 0xadc2, 0xd0d9, 0xadd0, 0xadc5, 0xadd9, 0xaddb, 0xd0d3, 0xadd8,
01991 0xd0db, 0xd0cd, 0xd0dc, 0xd0d1, 0xd0da, 0xd0d2, 0xadcc8, 0xd463,
01992 0xd457, 0xb0b3, 0xd45c, 0xd462, 0xb0b2, 0xd455, 0xb0b6, 0xd459,
01993 0xd452, 0xb0b4, 0xd456, 0xb0b9, 0xb0be, 0xd467, 0xd451, 0xb0ba,
01994 0xd466, 0xb0b5, 0xd458, 0xb0b1, 0xd453, 0xd44f, 0xd45d, 0xd450,
01995 0xd44e, 0xd45a, 0xd460, 0xd461, 0xb0b7, 0xd85b, 0xd45e, 0xd44d,
01996 0xd45f, 0xb0c1, 0xd464, 0xb0c0, 0xd44c, 0xd454, 0xd465, 0xb0bc,
01997 0xb0bb, 0xb0b8, 0xb0bd, 0xb0af, 0xb0b0, 0xb3c8, 0xd85e, 0xd857,
01998 0xb3c5, 0xd85f, 0xd855, 0xd858, 0xb3c4, 0xd859, 0xb3c7, 0xd85d,
01999 0xd853, 0xd852, 0xb3c9, 0xb3ca, 0xb3c6, 0xb3cb, 0xd851, 0xd85c,
02000 0xd85a, 0xd854, 0xb3c3, 0xd856, 0xb6ca, 0xb6c4, 0xdcbl7, 0xb6cd,
02001 0xdcbl, 0xdcc0, 0xb6c6, 0xb6c7, 0xdcba, 0xb6c5, 0xdcc3, 0xb6cb,
02002 0xdcc4, 0xdcblf, 0xb6cc, 0xdcbl4, 0xb6c9, 0xdcbl5, 0xdcbl6, 0xdcblc,
02003 0xdcbl8, 0xb6c8, 0xdcbl6, 0xb6ce, 0xdcblb, 0xdcc2, 0xdcbl9, 0xdcc1,
02004 0xb9b6, 0xb9b3, 0xb9b4, 0xe0f9, 0xe0f1, 0xb9b2, 0xb9af, 0xe0f2,
02005 0xb9b1, 0xe0f5, 0xe0fe, 0xe0fd, 0xe0f8, 0xb9ae, 0xe0f0,
02006 0xb9ac, 0xe0f3, 0xb9b7, 0xe0f6, 0xe0fa, 0xb9b0, 0xb9ad, 0xe0fc,
02007 0xe0fb, 0xb9b5, 0xe0f4, 0xbbf8, 0xe4ec, 0xe4e9, 0xbbf9, 0xbbf7,
02008 0xe4f0, 0xe4ed, 0xe4e6, 0xbbf6, 0xbbf8, 0xe4e7, 0xbbf5, 0xbbf4,
02009 0xe4ea, 0xe4eb, 0xbbf9, 0xbbf7, 0xe4f1, 0xe4ee, 0xe4ef, 0xbeaa,
02010 0xe8f8, 0xbea7, 0xe8f5, 0xbea9, 0xbeab, 0xe8f6, 0xbea8, 0xe8f7,
02011 0xe8f4, 0xc076, 0xecbd, 0xc077, 0xecbb, 0xecbc, 0xecba, 0xecb9,
02012 0xecbe, 0xc075, 0xefb8, 0xefb9, 0xe4e8, 0xefb7, 0xc078, 0xc35f,
02013 0xf1eb, 0xf1ec, 0xc4d7, 0xc4d8, 0xf5c1, 0xf5c0, 0xc56c, 0xc56b,
02014 0xf7d0, 0xa449, 0xa461, 0xa4b9, 0xa4b8, 0xa553, 0xa552, 0xa5fc,
02015 0xa5fb, 0xa5fd, 0xa5fa, 0xa74a, 0xa749, 0xa74b, 0xa8e0, 0xa8df,
02016 0xa8e1, 0xab5e, 0xa259, 0xd0de, 0xa25a, 0xb0c2, 0xa25c, 0xa25b,
02017 0xd860, 0xa25d, 0xb9b8, 0xa25e, 0xa44a, 0xa4ba, 0xa5fe, 0xa8e2,
02018 0xa44b, 0xa4bd, 0xa4bb, 0xa4bc, 0xa640, 0xa74c, 0xa8e4, 0xa8e3,
02019 0xa8e5, 0xaddd, 0xbeac, 0xc94e, 0xa554, 0xa555, 0xa641, 0xca6a,
02020 0xab60, 0xab5f, 0xd0e0, 0xd0df, 0xb0c3, 0xa4be, 0xc955, 0xc9cd,
02021 0xab61, 0xade0, 0xadde, 0xaddf, 0xbead, 0xa556, 0xa642, 0xc9bc,
02022 0xa74d, 0xa74e, 0xca6b, 0xcbbce, 0xa8e6, 0xcbbcf, 0xd0e2, 0xd0e3,
02023 0xade3, 0xd0e4, 0xd0e1, 0xade4, 0xade2, 0xade1, 0xd0e5, 0xd468,
02024 0xd861, 0xdcc5, 0xe140, 0xbbf6, 0xbeae, 0xe8f9, 0xa44c, 0xa45a,
02025 0xb0c4, 0xb3cd, 0xb9b9, 0xc942, 0xa4bf, 0xa559, 0xa557, 0xa558,
02026 0xa8e7, 0xa44d, 0xa44e, 0xa462, 0xa4c0, 0xa4c1, 0xa4c2, 0xc9be,
02027 0xa55a, 0xc96b, 0xa646, 0xc9bf, 0xa644, 0xa645, 0xc9bd, 0xa647,

```

02028 0xa643, 0xca6c, 0xaaec, 0xca6d, 0xca6e, 0xa750, 0xa74f, 0xa753,
02029 0xa751, 0xa752, 0xa8ed, 0xa8ec, 0xcbd4, 0xcbd1, 0xcbd2, 0xcbd0,
02030 0xa8ee, 0xa8ea, 0xa8e9, 0xa8eb, 0xa8e8, 0xa8ef, 0xab63, 0xcdf0,
02031 0xcbd3, 0xab68, 0xcdf1, 0xab64, 0xab67, 0xab66, 0xab65, 0xab62,
02032 0xd0e8, 0xade7, 0xd0eb, 0xade5, 0xd0e7, 0xade8, 0xade6, 0xade9,
02033 0xd0e9, 0xd0ea, 0xd0e6, 0xd0ec, 0xb3d1, 0xb0c5, 0xd469, 0xd46b,
02034 0xd46a, 0xd46c, 0xb0c6, 0xb3ce, 0xb3cf, 0xb3d0, 0xb6d0, 0xdcc7,
02035 0xdcc6, 0xdcc8, 0xdcc9, 0xb6d1, 0xb6cf, 0xe141, 0xe142, 0xb9bb,
02036 0xb9ba, 0xe35a, 0xbc40, 0xbc41, 0xbc42, 0xbc44, 0xe4f2, 0xe4f3,
02037 0xbc43, 0xbeaf, 0xbeb0, 0xf1ed, 0xf5c3, 0xf5c2, 0xf7d1, 0xa44f,
02038 0xa55c, 0xa55b, 0xa648, 0xc9c0, 0xa755, 0xa756, 0xa754, 0xa757,
02039 0xca6f, 0xca70, 0xa8f1, 0xcbd5, 0xa8f0, 0xcdf2, 0xab6c, 0xcdf3,
02040 0xab6b, 0xab69, 0xab6a, 0xd0ed, 0xb0c7, 0xd46e, 0xb0ca, 0xd46d,
02041 0xb1e5, 0xb0c9, 0xb0c8, 0xb3d4, 0xb3d3, 0xb3d2, 0xb6d2, 0xb6d5,
02042 0xb6d6, 0xb6d4, 0xb6d3, 0xe143, 0xe144, 0xe4f5, 0xbc45, 0xe4f4,
02043 0xbeb1, 0xecbf, 0xc079, 0xflee, 0xc455, 0xa463, 0xa4c3, 0xc956,
02044 0xa4c4, 0xa4c5, 0xa55d, 0xa55e, 0xa649, 0xca71, 0xcbd6, 0xcbd7,
02045 0xab6d, 0xa0ee, 0xb0cc, 0xb0cb, 0xd863, 0xd862, 0xa450, 0xa4c6,
02046 0xa55f, 0xb0cd, 0xc943, 0xc96c, 0xa560, 0xc9c2, 0xa64b, 0xa64a,
02047 0xc9c1, 0xa758, 0xadea, 0xd46f, 0xb6d7, 0xe145, 0xb9bc, 0xe8fa,
02048 0xf3fd, 0xa4c7, 0xcbd8, 0xcdf4, 0xb0d0, 0xb0ce, 0xb0cf, 0xa451,
02049 0xa464, 0xa2cd, 0xa4ca, 0xa4c9, 0xa4c8, 0xa563, 0xa562, 0xc96d,
02050 0xc9c3, 0xa8f5, 0xa8f2, 0xa8f4, 0xa8f3, 0xab6e, 0xb3d5, 0xa452,
02051 0xa4cb, 0xa565, 0xa564, 0xca72, 0xa8f6, 0xc957, 0xa567, 0xa566,
02052 0xa64c, 0xa64d, 0xca73, 0xa759, 0xa75a, 0xa8f7, 0xa8f8, 0xa8f9,
02053 0xab6f, 0xcdf5, 0xadeb, 0xc944, 0xa4cc, 0xc9c4, 0xca74, 0xca75,
02054 0xcbd9, 0xcbda, 0xcdf7, 0xcdf6, 0xcdf9, 0xcdf8, 0xab70, 0xd470,
02055 0xaded, 0xd0ef, 0xadec, 0xd864, 0xb3d6, 0xd865, 0xe146, 0xb9bd,
02056 0xbc46, 0xf1ef, 0xc958, 0xa568, 0xb0d1, 0xa453, 0xa465, 0xa4ce,
02057 0xa4cd, 0xa4cf, 0xa8fb, 0xa8fa, 0xa8fc, 0xab71, 0xadee, 0xe8fb,
02058 0xc24f, 0xa466, 0xa56a, 0xa579, 0xa574, 0xa56f, 0xa56e, 0xa575,
02059 0xa573, 0xa56c, 0xa57a, 0xa56d, 0xa569, 0xa578, 0xa577, 0xa576,
02060 0xa56b, 0xa572, 0xa571, 0xa57b, 0xa570, 0xa653, 0xa659, 0xa655,
02061 0xa65b, 0xc9c5, 0xa658, 0xa64e, 0xa651, 0xa654, 0xa650, 0xa657,
02062 0xa65a, 0xa64f, 0xa652, 0xa656, 0xa65c, 0xca7e, 0xca7b, 0xa767,
02063 0xca7c, 0xa75b, 0xa75d, 0xa775, 0xa770, 0xcaa5, 0xca7d, 0xa75f,
02064 0xa761, 0xcaa4, 0xa768, 0xca78, 0xa774, 0xa776, 0xa75c, 0xa76d,
02065 0xca76, 0xa773, 0xa764, 0xa76e, 0xa76f, 0xca77, 0xa76c, 0xa76a,
02066 0xa76b, 0xa771, 0xcaa1, 0xa75e, 0xa772, 0xcaa3, 0xa766, 0xa763,
02067 0xca7a, 0xa762, 0xcaa6, 0xa765, 0xa769, 0xa760, 0xcaa2, 0xca79,
02068 0xcbeb, 0xcbea, 0xa94f, 0xcbed, 0xcbeef, 0xcbe4, 0xcbe7, 0xcbee,
02069 0xa950, 0xcbe1, 0xcbe5, 0xcbe9, 0xa94b, 0xa94a, 0xa94d, 0xa94e,
02070 0xcbe6, 0xa8fe, 0xa94c, 0xa945, 0xa941, 0xcbe2, 0xa944, 0xa949,
02071 0xa952, 0xcbe3, 0xcbed, 0xa943, 0xcbed, 0xcbedf, 0xa946, 0xa948,
02072 0xcdbd, 0xcbe0, 0xa951, 0xa94d, 0xcbe8, 0xa953, 0xa94a, 0xcbe9,
02073 0xa947, 0xa942, 0xa940, 0xcbec, 0xa94e, 0xc4e8, 0xcdfb, 0xc4eb,
02074 0xcdfd, 0xab78, 0xaba8, 0xab74, 0xaba7, 0xab7d, 0xaba4, 0xab72,
02075 0xcdfc, 0xc43b, 0xaba3, 0xc44f, 0xaba5, 0xab79, 0xc445, 0xc442,
02076 0xab77, 0xcdfa, 0xaba6, 0xc44a, 0xab7c, 0xc44c, 0xaba9, 0xab73,
02077 0xab7e, 0xab7b, 0xc440, 0xaba1, 0xc446, 0xc447, 0xab7a, 0xaba2,
02078 0xab76, 0xab75, 0xcdfe, 0xc444, 0xc44e, 0xd144, 0xadfb, 0xd0f1,
02079 0xd0f6, 0xadfa, 0xae40, 0xd0f4, 0xadee, 0xadf9, 0xadfe, 0xd0fb,
02080 0xadfa, 0xadfd, 0xd0fe, 0xadf5, 0xd0f5, 0xd142, 0xd143, 0xadf7,
02081 0xd141, 0xadf3, 0xae43, 0xd0f8, 0xadf1, 0xd146, 0xd0f9, 0xd0fd,
02082 0xadf6, 0xae42, 0xd0fa, 0xadfc, 0xd140, 0xd147, 0xd4a1, 0xd145,
02083 0xae44, 0xadf0, 0xd0fc, 0xd0f3, 0xadf8, 0xd0f2, 0xd0f7, 0xd0f0,
02084 0xae41, 0xd477, 0xb0e4, 0xd4a7, 0xb0e2, 0xb0df, 0xd47c, 0xb0db,
02085 0xd4a2, 0xb0e6, 0xd476, 0xd47b, 0xd47a, 0xadf2, 0xb0e1, 0xd4a5,
02086 0xd4a8, 0xd473, 0xb3e8, 0xd4a9, 0xb0e7, 0xb0d9, 0xb0d6, 0xd47e,
02087 0xb0d3, 0xd4a6, 0xb0da, 0xd4aa, 0xd474, 0xd4a4, 0xb0dd, 0xd475,
02088 0xd478, 0xd47d, 0xb0de, 0xb0dc, 0xb0e8, 0xb0e3, 0xb0d7, 0xb1d2,
02089 0xb0d8, 0xd479, 0xb0e5, 0xb0e0, 0xd4a3, 0xb0d5, 0xb0d4, 0xd471,
02090 0xd472, 0xd86a, 0xb3d7, 0xb3da, 0xd875, 0xb3ee, 0xd878, 0xb3d8,
02091 0xd871, 0xb3de, 0xb3e4, 0xb5bd, 0xb3e2, 0xd86e, 0xb3ef, 0xb3db,
02092 0xb3e3, 0xd876, 0xdcd7, 0xd87b, 0xd86f, 0xd866, 0xd873, 0xd86d,
02093 0xb3e1, 0xd879, 0xb3dd, 0xb3f1, 0xb3ea, 0xb3df, 0xb3dc, 0xb3e7,
02094 0xd87a, 0xd86c, 0xd872, 0xd874, 0xd868, 0xd877, 0xb3d9, 0xd867,
02095 0xb3e0, 0xb3f0, 0xb3ec, 0xd869, 0xb3e6, 0xb3ed, 0xb3e9, 0xb3e5,
02096 0xd870, 0xb3eb, 0xdcd5, 0xdcd1, 0xdce0, 0xdcca, 0xdcd3, 0xb6e5,
02097 0xb6e6, 0xb6de, 0xdcdc, 0xb6e8, 0xdccc, 0xdccc, 0xdcdc, 0xdcdc,
02098 0xb6dc, 0xdcdc, 0xdcdc, 0xb6df, 0xdcdc, 0xb6da, 0xdcd2, 0xdcd9,
02099 0xdcdb, 0xdcdf, 0xb6e3, 0xdccb, 0xb6dd, 0xdcd0, 0xb6d8, 0xb6e4,
02100 0xdcd4, 0xb6e0, 0xb6e1, 0xb6e7, 0xb6bd, 0xa25f, 0xb6d9, 0xdcd4,
02101 0xb6e2, 0xdcd8, 0xb9cd, 0xb9c8, 0xe155, 0xe151, 0xe14b, 0xb9c2,
02102 0xb9be, 0xe154, 0xb9bf, 0xe14e, 0xe150, 0xe153, 0xb9c4, 0xb9cb,
02103 0xb9c5, 0xe149, 0xb9c6, 0xb9c7, 0xe14c, 0xb9cc, 0xe14a, 0xe14f,
02104 0xb9c3, 0xe148, 0xb9c9, 0xb9c1, 0xb9c0, 0xe14d, 0xe152, 0xb9ca,
02105 0xe147, 0xb9c4, 0xe547, 0xe544, 0xb9c7, 0xb9c5, 0xb9c4, 0xb9c4a,
02106 0xe542, 0xb9c4, 0xe4f9, 0xb9c5, 0xe546, 0xb9c4, 0xe548, 0xb9c4,
02107 0xe543, 0xe545, 0xb9c4, 0xe541, 0xe4fa, 0xe4f7, 0xd86b, 0xe4fd,
02108 0xe4f6, 0xe4fc, 0xe4fb, 0xe4f8, 0xb9c4, 0xb9c4, 0xb9c5, 0xe4fe,
02109 0xb9b2, 0xe540, 0xe945, 0xe8fd, 0xb9be, 0xe942, 0xb9be, 0xb9ba,
02110 0xe941, 0xb9b9, 0xb9b5, 0xb9b8, 0xb9b3, 0xb9bd, 0xe943, 0xe8fe,
02111 0xb9bc, 0xe8fc, 0xb9bb, 0xe944, 0xe940, 0xb9c5, 0xb9bf, 0xe946,
02112 0xb9b7, 0xb9ba, 0xeccc, 0xeccc, 0xc07b, 0xeccc, 0xeccc, 0xeccc,
02113 0xeccc, 0xc07d, 0xeccc, 0xc07e, 0xeccc, 0xeccc, 0xc07a, 0xc0a1,
02114 0xc07c, 0xeccc, 0xc250, 0xefbc, 0xefba, 0xefbf, 0xefbd, 0xefbb,

```


02115 0xefbe, 0xc360, 0xf1f2, 0xf1f3, 0xc456, 0xf1f4, 0xf1f0, 0xf1f5,
02116 0xf1f1, 0xc251, 0xf3fe, 0xf441, 0xc459, 0xf440, 0xc458, 0xc457,
02117 0xc45a, 0xf5c5, 0xf5c6, 0xc4da, 0xc4d9, 0xc4db, 0xf5c4, 0xf6d8,
02118 0xf6d7, 0xc56d, 0xc56f, 0xc56e, 0xf6d9, 0xc5c8, 0xf8a6, 0xc5f1,
02119 0xf8a5, 0xf8ee, 0xc949, 0xa57d, 0xa57c, 0xa65f, 0xa65e, 0xc9c7,
02120 0xa65d, 0xc9c6, 0xa779, 0xc9c7, 0xc9c8, 0xa777, 0xc9c9, 0xc9ca,
02121 0xa778, 0xc9cb, 0xc9cc, 0xa954, 0xabaa, 0xd148, 0xd149, 0xae45,
02122 0xae46, 0xd14a, 0xb0e9, 0xb0eb, 0xd14b, 0xb0ea, 0xd87c, 0xb3f2,
02123 0xb6e9, 0xb6ea, 0xdce1, 0xb9cf, 0xb9ce, 0xe549, 0xe948, 0xe947,
02124 0xf96b, 0xa467, 0xc959, 0xc96e, 0xc96f, 0xa662, 0xa666, 0xc9c9,
02125 0xa664, 0xa663, 0xc9c8, 0xa665, 0xa661, 0xa660, 0xc9ca, 0xa7a6,
02126 0xa7a3, 0xa77d, 0xc9ca, 0xc9ab, 0xa7a1, 0xc9ad, 0xa77b, 0xc9ae,
02127 0xc9ac, 0xa77e, 0xa7a2, 0xa7a5, 0xa7a4, 0xa77c, 0xc9af, 0xa959,
02128 0xc9bf, 0xa95b, 0xa95a, 0xcc40, 0xa958, 0xa957, 0xc9bf, 0xc9bf,
02129 0xc9bf, 0xc9bf, 0xc9bf, 0xc9bf, 0xc9bf, 0xc9bf, 0xc9bf, 0xc9bf,
02130 0xa956, 0xc9bf, 0xa95c, 0xcc41, 0xc9bf, 0xabab, 0xa955, 0xabac,
02131 0xc954, 0xc95a, 0xab2, 0xc958, 0xc95e, 0xc955, 0xc959, 0xc95b,
02132 0xc95d, 0xc957, 0xc956, 0xc951, 0xc952, 0xabad, 0xabaf, 0xabae,
02133 0xc953, 0xc95c, 0xab1, 0xc950, 0xd153, 0xd152, 0xd157, 0xd14e,
02134 0xd151, 0xd150, 0xd154, 0xd158, 0xae47, 0xae4a, 0xd14f, 0xd155,
02135 0xae49, 0xd14a, 0xab0, 0xd14b, 0xd156, 0xd14d, 0xae48, 0xd14c,
02136 0xd4b1, 0xb0ec, 0xb0f0, 0xd4c1, 0xd4af, 0xd4bd, 0xb0f1, 0xd4bf,
02137 0xd4c5, 0xd4c9, 0xd4c0, 0xd4b4, 0xd4bc, 0xd4ca, 0xd4cb, 0xd4be,
02138 0xd4b9, 0xd4b2, 0xd8a6, 0xd4b0, 0xb0f5, 0xd4b7, 0xb0f6, 0xb0f2,
02139 0xd4ad, 0xd4c3, 0xd4b5, 0xd4b3, 0xd4c6, 0xb0f3, 0xd4cc, 0xb0ed,
02140 0xb0ef, 0xd4bb, 0xd4b6, 0xae4b, 0xb0ee, 0xd4b8, 0xd4c7, 0xd4cb,
02141 0xd4c2, 0xd4c4, 0xd4ae, 0xd8a1, 0xd8aa, 0xd8a9, 0xb3fa, 0xd8a2,
02142 0xb3fb, 0xb3f9, 0xd8a4, 0xb3f6, 0xd8a8, 0xd8a3, 0xd8a5, 0xd87d,
02143 0xb3f4, 0xd8b2, 0xd8b1, 0xd8ae, 0xb3f3, 0xb3f7, 0xb3f8, 0xd14b,
02144 0xd8ab, 0xb3f5, 0xb0f4, 0xd8ad, 0xd87e, 0xd8b0, 0xd8af, 0xd8b3,
02145 0xdcef, 0xd8ac, 0xd8a7, 0xdce7, 0xb6f4, 0xb6f7, 0xb6f2, 0xdce6,
02146 0xdcea, 0xdce5, 0xb6ec, 0xb6f6, 0xdce2, 0xb6f0, 0xdce9, 0xb6ee,
02147 0xb6ed, 0xdcec, 0xb6ef, 0xdcee, 0xdceb, 0xb6eb, 0xb6f5, 0xdcf0,
02148 0xdce4, 0xdced, 0xdce3, 0xb6f1, 0xb6f3, 0xdce8, 0xdcf1, 0xe15d,
02149 0xb9d0, 0xe163, 0xb9d5, 0xe15f, 0xe166, 0xe157, 0xb9d7, 0xb9d1,
02150 0xe15c, 0xb9d6, 0xe15b, 0xe164, 0xb9d2, 0xb9d6, 0xe15a, 0xe160,
02151 0xe165, 0xe156, 0xb9d4, 0xe15e, 0xe162, 0xe168, 0xe158, 0xe161,
02152 0xb9d3, 0xe167, 0xe159, 0xb9d5, 0xe54b, 0xb9d7, 0xb9d6, 0xe54d,
02153 0xe552, 0xe54e, 0xe551, 0xb9d5, 0xb9d6, 0xe54b, 0xe54a, 0xe550,
02154 0xb9d5, 0xe54f, 0xe54c, 0xb9d8, 0xe94d, 0xe94f, 0xe94a, 0xbec1,
02155 0xe94c, 0xbec0, 0xe94e, 0xbec3, 0xe950, 0xbec2, 0xe949, 0xe94b,
02156 0xc0a5, 0xeccc, 0xc0a4, 0xeccc, 0xc0a3, 0xeccc, 0xc0a2, 0xecca,
02157 0xc253, 0xc252, 0xf1f6, 0xf1f8, 0xf1f7, 0xc361, 0xc362, 0xc363,
02158 0xf442, 0xc45b, 0xf7d3, 0xf7d2, 0xc5f2, 0xa468, 0xa4d0, 0xa7a7,
02159 0xc5f5, 0xb3fc, 0xdcf2, 0xb9d8, 0xe169, 0xe553, 0xc95a,
02160 0xcab0, 0xcc42, 0xce60, 0xd159, 0xae4c, 0xf1f9, 0xc4dc, 0xa469,
02161 0xa57e, 0xc970, 0xa667, 0xa668, 0xa95d, 0xb0f7, 0xb9da, 0xb9db,
02162 0xb9d9, 0xa46a, 0xa4d1, 0xa4d3, 0xa4d2, 0xc95b, 0xa4d4, 0xa5a1,
02163 0xc971, 0xa5a2, 0xa669, 0xa66a, 0xc9cb, 0xa7a8, 0xcab1, 0xa961,
02164 0xcc43, 0xa95f, 0xa960, 0xa95e, 0xd15a, 0xab6, 0xab6, 0xab7,
02165 0xab4, 0xcc61, 0xa962, 0xab3, 0xae4d, 0xae4e, 0xae4f, 0xd4cd,
02166 0xb3fe, 0xd8b4, 0xb0f8, 0xb6f8, 0xb9dd, 0xb9dc, 0xe16a, 0xb9d5,
02167 0xbec4, 0xfefc, 0xf6da, 0xf7d4, 0xa46b, 0xa5a3, 0xa5a4, 0xc9d1,
02168 0xa66c, 0xa66f, 0xc9cf, 0xc9cd, 0xa66e, 0xc9d0, 0xc9d2, 0xc9cc,
02169 0xa671, 0xa670, 0xa66d, 0xa66b, 0xc9ce, 0xa7b3, 0xa7b0, 0xcab6,
02170 0xcab9, 0xcab8, 0xa7aa, 0xa7b2, 0xa7af, 0xcab5, 0xcab3, 0xa7ae,
02171 0xa7a9, 0xa7ac, 0xcab4, 0xcabb, 0xcab7, 0xa7ad, 0xa7b1, 0xa7b4,
02172 0xcab2, 0xcaba, 0xa7ab, 0xa967, 0xa96f, 0xcc4f, 0xcc48, 0xa970,
02173 0xcc53, 0xcc44, 0xcc4b, 0xa966, 0xcc45, 0xa964, 0xcc4c, 0xcc50,
02174 0xa963, 0xcc51, 0xcc4a, 0xcc4d, 0xa972, 0xa969, 0xcc54, 0xcc52,
02175 0xa96e, 0xa96c, 0xcc49, 0xa96b, 0xcc47, 0xcc46, 0xa96a, 0xa968,
02176 0xa971, 0xa96d, 0xa965, 0xcc4e, 0xab9, 0xabc0, 0xce6f, 0xab8,
02177 0xce67, 0xce63, 0xce73, 0xce62, 0xabbb, 0xce6c, 0xabbe, 0xabc1,
02178 0xabbc, 0xce70, 0xabbf, 0xae56, 0xce76, 0xce64, 0xce66, 0xce6d,
02179 0xce71, 0xce75, 0xce72, 0xce6b, 0xce6e, 0xabc3, 0xce6a,
02180 0xce69, 0xce74, 0xabba, 0xce65, 0xabc2, 0xabbd, 0xae5c, 0xd162,
02181 0xae5b, 0xd160, 0xae50, 0xae55, 0xd15f, 0xd15c, 0xd161, 0xae51,
02182 0xd15b, 0xae54, 0xae52, 0xd163, 0xae53, 0xae57, 0xae58, 0xae5a,
02183 0xae59, 0xd15d, 0xd15e, 0xd164, 0xd4d4, 0xb0f9, 0xd8c2, 0xd4d3,
02184 0xd4e6, 0xb140, 0xd4e4, 0xb0fe, 0xb0fa, 0xd4ed, 0xd4dd, 0xd4e0,
02185 0xb143, 0xd4ea, 0xd4e2, 0xb0fb, 0xb144, 0xd4e7, 0xd4e5, 0xd4d6,
02186 0xd4eb, 0xd4df, 0xd4da, 0xd4d0, 0xd4ec, 0xd4dc, 0xd4cf, 0xb142,
02187 0xd4e1, 0xd4ee, 0xd4de, 0xd4d2, 0xd4d7, 0xd4ce, 0xb141, 0xd4db,
02188 0xd4d8, 0xb0fc, 0xd4d1, 0xd4e9, 0xb0fd, 0xd4d9, 0xd4d5, 0xd4e8,
02189 0xb440, 0xd8bb, 0xd8b8, 0xd8c9, 0xd8bd, 0xd8ca, 0xb442, 0xd8c6,
02190 0xd8c3, 0xd8c4, 0xd8c7, 0xd8cb, 0xd4e3, 0xd8cd, 0xdd47, 0xb443,
02191 0xd8ce, 0xd8b6, 0xd8c0, 0xd8c5, 0xb441, 0xb444, 0xd8cc, 0xd8cf,
02192 0xd8ba, 0xd8b7, 0xd8b9, 0xd8be, 0xd8bc, 0xb445, 0xd8c8, 0xd8bf,
02193 0xd8c1, 0xd8b5, 0xdcf8, 0xdcf8, 0xb742, 0xb740, 0xdd44, 0xdcf9,
02194 0xdd44, 0xdd40, 0xdcf7, 0xdd46, 0xdcf6, 0xdcf6, 0xb6fe, 0xb6fd,
02195 0xb6fc, 0xdcfb, 0xdd41, 0xb6f9, 0xb741, 0xdcf4, 0xdcf6, 0xdcf3,
02196 0xdcf8, 0xb6fa, 0xdd42, 0xdcf5, 0xb6fb, 0xdd45, 0xe16e, 0xb9e2,
02197 0xb9e1, 0xb9e3, 0xe17a, 0xe170, 0xe176, 0xe16b, 0xe179, 0xe178,
02198 0xe17c, 0xe175, 0xb9de, 0xe174, 0xb9e4, 0xe16d, 0xb9df, 0xe17b,
02199 0xb9e0, 0xe16f, 0xe172, 0xe177, 0xe171, 0xe16c, 0xe173, 0xe555,
02200 0xb9e1, 0xe558, 0xe557, 0xe55a, 0xe55c, 0xb9e5, 0xe556, 0xe554,
02201 0xe55d, 0xe55b, 0xe559, 0xe55f, 0xe55e, 0xb9e6, 0xb9e5, 0xb9e6,

```

02202 0xbc62, 0xe560, 0xe957, 0xe956, 0xe955, 0xe958, 0xe951, 0xe952,
02203 0xe95a, 0xe953, 0xbec5, 0xe95c, 0xe95b, 0xe954, 0xecd1, 0xc0a8,
02204 0xeccf, 0xecd4, 0xecd3, 0xe959, 0xc0a7, 0xecd2, 0xeccc, 0xecd6,
02205 0xecd5, 0xc0a6, 0xecd0, 0xbec6, 0xc254, 0xefc1, 0xf1fa, 0xf1fb,
02206 0xf1fc, 0xc45c, 0xc45d, 0xf443, 0xf5c8, 0xf5c7, 0xf6db, 0xf6dc,
02207 0xf7d5, 0xf8a7, 0xa46c, 0xa46d, 0xa46e, 0xa4d5, 0xa5a5, 0xc9d3,
02208 0xa672, 0xa673, 0xa7b7, 0xa7b8, 0xa7b6, 0xa7b5, 0xa973, 0xcc55,
02209 0xa975, 0xa974, 0xcc56, 0xabc4, 0xae5d, 0xd165, 0xd4f0, 0xb145,
02210 0xb447, 0xd4ef, 0xb446, 0xb9e5, 0xe17d, 0xbec7, 0xc0a9, 0xecd7,
02211 0xc45e, 0xc570, 0xc972, 0xa5a6, 0xc973, 0xa676, 0xa674, 0xa675,
02212 0xa677, 0xa7ba, 0xa7b9, 0xcabc, 0xa7bb, 0xcabd, 0xcc57, 0xcc58,
02213 0xa976, 0xa978, 0xa97a, 0xa977, 0xa97b, 0xa979, 0xabc8, 0xabc5,
02214 0xabc7, 0xabc9, 0xabc6, 0xd166, 0xce77, 0xd168, 0xd167, 0xae63,
02215 0xae5f, 0xae60, 0xae62, 0xae64, 0xae61, 0xae66, 0xae65, 0xb14a,
02216 0xd4f2, 0xd4f1, 0xb149, 0xb148, 0xb147, 0xb14b, 0xb146, 0xd8d5,
02217 0xd8d2, 0xb449, 0xd8d1, 0xd8d6, 0xb44b, 0xd8d4, 0xb448, 0xb44a,
02218 0xd8d3, 0xdd48, 0xdd49, 0xdd4a, 0xb9e6, 0xb9ee, 0xe17e, 0xb9e8,
02219 0xb9ec, 0xe1a1, 0xb9ed, 0xb9e9, 0xb9ea, 0xb9ef, 0xb9eb, 0xb9ec,
02220 0xd8d0, 0xbc67, 0xbc65, 0xbc64, 0xe95d, 0xbec8, 0xecd8, 0xecd9,
02221 0xc364, 0xc45f, 0xa46f, 0xa678, 0xabca, 0xd169, 0xae67, 0xb14e,
02222 0xb14d, 0xb14c, 0xb14c, 0xb44c, 0xb44d, 0xd8d7, 0xb9ef, 0xbec9, 0xa470,
02223 0xc95c, 0xa4d6, 0xc974, 0xc9d4, 0xa679, 0xa97c, 0xdd4b, 0xa471,
02224 0xa4d7, 0xc9d5, 0xcabe, 0xcabf, 0xa7bc, 0xd8d8, 0xb44e, 0xdd4c,
02225 0xc0aa, 0xa472, 0xa4a8, 0xa4d8, 0xc975, 0xa5a7, 0xa7c0, 0xa7bf,
02226 0xa7bd, 0xa7be, 0xcc59, 0xa97e, 0xa9a1, 0xcc5a, 0xa97d, 0xabcce,
02227 0xce78, 0xabc0, 0xabc1, 0xabc2, 0xae6a, 0xae68, 0xd16b, 0xae69,
02228 0xd16a, 0xae5e, 0xd4f3, 0xb150, 0xb151, 0xb14f, 0xb9f0, 0xe1a2,
02229 0xbc68, 0xbc69, 0xe561, 0xc0ab, 0xefc2, 0xefc3, 0xc4dd, 0xf8a8,
02230 0xc94b, 0xa4d9, 0xa473, 0xc977, 0xc976, 0xa67a, 0xc9d7, 0xc9d8,
02231 0xc9d6, 0xc9d9, 0xcac7, 0xcac2, 0xcac4, 0xcac6, 0xcac3, 0xa7c4,
02232 0xcac0, 0xcac1, 0xa7c1, 0xa7c2, 0xcac5, 0xcac8, 0xa7c3, 0xcac9,
02233 0xcc68, 0xcc62, 0xcc5d, 0xa9a3, 0xcc65, 0xcc63, 0xcc5c, 0xcc69,
02234 0xcc6c, 0xcc67, 0xcc60, 0xa9a5, 0xcc66, 0xa9a6, 0xcc61, 0xcc64,
02235 0xcc5b, 0xcc5f, 0xcc6b, 0xa9a7, 0xa9a8, 0xcc5e, 0xcc6a, 0xa9a2,
02236 0xa9a4, 0xcea6, 0xcea4, 0xceaa, 0xcea3, 0xcea5, 0xce7d, 0xce7b,
02237 0xceac, 0xcea9, 0xce79, 0xabd0, 0xcea7, 0xcea8, 0xcea6, 0xce7c,
02238 0xce7a, 0xabc7, 0xcea2, 0xce7e, 0xcea1, 0xcead, 0xae6f, 0xae6e,
02239 0xd16c, 0xae6b, 0xd16e, 0xae70, 0xd16f, 0xae73, 0xae71, 0xd170,
02240 0xceae, 0xae6d, 0xae6f, 0xae6c, 0xd16d, 0xd171, 0xae72, 0xb153,
02241 0xb152, 0xd4f5, 0xd4f9, 0xd4fb, 0xb154, 0xd4fe, 0xb158, 0xd541,
02242 0xb15a, 0xb156, 0xb15e, 0xb15b, 0xd4f7, 0xb155, 0xd4f6, 0xd4f4,
02243 0xd543, 0xd4f8, 0xb157, 0xd542, 0xb15c, 0xd4fd, 0xd4fc, 0xb15d,
02244 0xd4fa, 0xb159, 0xd544, 0xd540, 0xd8e7, 0xd8ee, 0xd8e3, 0xb451,
02245 0xd8df, 0xd8ef, 0xd8d9, 0xd8ec, 0xd8ea, 0xd8e4, 0xd8ed, 0xd8e6,
02246 0xd8de, 0xd8f0, 0xd8dc, 0xd8e9, 0xd8da, 0xd8f1, 0xb452, 0xd8eb,
02247 0xdd4f, 0xd8dd, 0xb44f, 0xd8e1, 0xb450, 0xd8e0, 0xd8e5, 0xd8e2,
02248 0xd8e8, 0xdd53, 0xdd56, 0xdd4e, 0xdd50, 0xdd55, 0xdd54, 0xb743,
02249 0xd8db, 0xdd52, 0xb744, 0xdd4d, 0xdd51, 0xe1a9, 0xe1b0, 0xe1a7,
02250 0xe1ae, 0xe1a5, 0xe1ad, 0xe1b1, 0xe1a4, 0xe1a8, 0xe1a3, 0xb9f1,
02251 0xe1a6, 0xb9f2, 0xe1ac, 0xe1ab, 0xe1aa, 0xe1af, 0xe565, 0xe567,
02252 0xbcb6, 0xbcb7, 0xe563, 0xe562, 0xe56c, 0xe56a, 0xbcb6, 0xe56d,
02253 0xe564, 0xe569, 0xe56b, 0xe566, 0xe961, 0xe966, 0xe960, 0xe965,
02254 0xe95e, 0xe968, 0xe964, 0xe969, 0xe963, 0xe95f, 0xe967, 0xe96a,
02255 0xe962, 0xecd4, 0xc0af, 0xc0ad, 0xc0ac, 0xc0ae, 0xefc4, 0xf172,
02256 0xf1fd, 0xf444, 0xf445, 0xc460, 0xf5c9, 0xc4de, 0xf5ca, 0xf6de,
02257 0xc572, 0xc571, 0xf6dd, 0xc5c9, 0xf7d6, 0xa474, 0xa67b, 0xc9da,
02258 0xcaca, 0xa8b5, 0xb15f, 0xa475, 0xa5aa, 0xa5a9, 0xa5a8, 0xa7c5,
02259 0xae74, 0xdd57, 0xa476, 0xa477, 0xa478, 0xa4da, 0xabd1, 0xceaf,
02260 0xb453, 0xa479, 0xc95d, 0xa5ab, 0xa5ac, 0xc978, 0xa67c, 0xcacb,
02261 0xa7c6, 0xcacc, 0xa9ae, 0xcc6e, 0xa9ac, 0xa9ab, 0xcc6d, 0xa9a9,
02262 0xcc6f, 0xa9aa, 0xa9ad, 0xabd2, 0xabd4, 0xc6b3, 0xc6b0, 0xc6b1,
02263 0xc6b2, 0xc6b4, 0xabd3, 0xd174, 0xd173, 0xae76, 0xae75, 0xb162,
02264 0xd546, 0xb161, 0xb163, 0xb160, 0xb455, 0xd545, 0xb456, 0xd8f3,
02265 0xb457, 0xd8f2, 0xb454, 0xdd5a, 0xdd5c, 0xb745, 0xdd5b, 0xdd59,
02266 0xdd58, 0xe1b4, 0xb9f7, 0xb9f5, 0xb9f6, 0xe1b2, 0xe1b3, 0xb9f3,
02267 0xe571, 0xe56f, 0xbcb6, 0xe570, 0xbcb6, 0xbcb6, 0xb9f4, 0xe96d,
02268 0xe96b, 0xe96c, 0xe56e, 0xecd4, 0xc0b0, 0xecd4, 0xefc5, 0xefc6,
02269 0xe96e, 0xf1fe, 0xa47a, 0xa5ad, 0xa67e, 0xc9db, 0xa67d, 0xa9af,
02270 0xb746, 0xa4db, 0xa5ae, 0xabd5, 0xb458, 0xc979, 0xc97a, 0xc9dc,
02271 0xa7c8, 0xcad0, 0xcace, 0xa7c9, 0xcacd, 0xcacf, 0xcad1, 0xa7c7,
02272 0xa9b3, 0xa9b4, 0xa9b1, 0xa9b0, 0xc6b8, 0xa9b2, 0xabd6, 0xc6b7,
02273 0xc6b9, 0xc6be, 0xc6ba, 0xabd7, 0xae79, 0xd175, 0xd177, 0xae77,
02274 0xd178, 0xae78, 0xd176, 0xc6b5, 0xd547, 0xd54a, 0xd54b, 0xd548,
02275 0xb167, 0xb166, 0xb164, 0xb165, 0xd549, 0xb168, 0xb45a, 0xb45b,
02276 0xb45c, 0xdd5d, 0xdd5f, 0xdd61, 0xb748, 0xb747, 0xb459, 0xdd60,
02277 0xdd5e, 0xe1b8, 0xe1b6, 0xe1bc, 0xb9f8, 0xe1bd, 0xe1ba, 0xb9f9,
02278 0xe1b7, 0xe1b5, 0xe1bb, 0xb9f7, 0xe573, 0xe1b9, 0xb9f2, 0xe574,
02279 0xb9f1, 0xb9f4, 0xe575, 0xb9f6, 0xb9f3, 0xe973, 0xe971, 0xe970,
02280 0xe972, 0xe96f, 0xc366, 0xf446, 0xf447, 0xf5cb, 0xf6df, 0xc655,
02281 0xa9b5, 0xa7ca, 0xabd8, 0xa47b, 0xa4dc, 0xa5af, 0xc9dd, 0xa7cb,
02282 0xcad2, 0xc6bb, 0xabd9, 0xb9fa, 0xa47c, 0xa6a1, 0xb749, 0xa47d,
02283 0xa4dd, 0xa4de, 0xa5b1, 0xa5b0, 0xc9de, 0xa6a2, 0xcad3, 0xa7cc,
02284 0xcc71, 0xcc72, 0xcc73, 0xa9b6, 0xa9b7, 0xcc70, 0xa9b8, 0xabda,
02285 0xc6bc, 0xd17a, 0xae7a, 0xd179, 0xb169, 0xd54c, 0xb16a, 0xd54d,
02286 0xb45d, 0xdd62, 0xe1bf, 0xe1be, 0xb9fb, 0xb9f5, 0xe576, 0xbeca,
02287 0xe974, 0xc0b1, 0xc573, 0xf7d8, 0xcc74, 0xc6bd, 0xb16b, 0xd8f4,
02288 0xb74a, 0xc255, 0xa7ce, 0xa7cd, 0xabdb, 0xd17b, 0xb16d, 0xb343,

```

02289 0xb16e, 0xb16c, 0xb45e, 0xelc0, 0xb9fc, 0xbc76, 0xc94c, 0xc9df,
02290 0xcd5, 0xa7cf, 0xcd4, 0xa7d0, 0xa9bc, 0xcc77, 0xcc76, 0xa9bb,
02291 0xa9b9, 0xa9ba, 0xcc75, 0xabdd, 0xcbe, 0xab0, 0xabdc, 0xab2,
02292 0xabde, 0xabdf, 0xabe1, 0xae7d, 0xae7c, 0xae7b, 0xd54f, 0xb16f,
02293 0xb172, 0xb170, 0xd54e, 0xb175, 0xb171, 0xd550, 0xb174, 0xb173,
02294 0xd8f6, 0xd8f5, 0xb461, 0xb45f, 0xb460, 0xd8f7, 0xb74b, 0xdd64,
02295 0xb74c, 0xdd63, 0xe577, 0xbc78, 0xelc1, 0xbc77, 0xb9fd, 0xecde,
02296 0xe975, 0xc0b2, 0xecdd, 0xf240, 0xf448, 0xf449, 0xa4df, 0xa5b2,
02297 0xc97b, 0xa7d2, 0xa7d4, 0xc9e2, 0xcd8, 0xcd7, 0xcd6, 0xc9e1,
02298 0xc9e0, 0xa6a4, 0xa7d3, 0xa7d1, 0xa6a3, 0xa9bd, 0xcc78, 0xa9be,
02299 0xcadd, 0xcadf, 0xcade, 0xcc79, 0xcada, 0xa7d8, 0xa7d6, 0xcd9,
02300 0xcadb, 0xcae1, 0xa7d5, 0xcadc, 0xcae5, 0xa9c0, 0xcae2, 0xa7d7,
02301 0xcae0, 0xcae3, 0xa9bf, 0xa9c1, 0xcae4, 0xccaf, 0xcac2, 0xcc7e,
02302 0xcacae, 0xcac9, 0xabe7, 0xa9c2, 0xcacae, 0xcacae, 0xcacae3, 0xcacac,
02303 0xa9c3, 0xa9c8, 0xa9c8, 0xa9c6, 0xcac3, 0xcc7c, 0xcac5, 0xa9cd, 0xcdb0,
02304 0xabe4, 0xcac6, 0xabe5, 0xa9c9, 0xcac8, 0xcacd, 0xabe6, 0xcc7b,
02305 0xa9ca, 0xabe8, 0xa9cb, 0xa9c7, 0xa9cc, 0xcac7, 0xcc7a, 0xcacab,
02306 0xa9c4, 0xcc7d, 0xcac4, 0xcac1, 0xa9c5, 0xcdbf, 0xcdbf, 0xcdbf,
02307 0xd1a1, 0xcdbf, 0xabee, 0xcdbf, 0xcdbf, 0xcdbf, 0xcdbf, 0xcdbf,
02308 0xcdbf, 0xabe9, 0xaea3, 0xcdbf, 0xcdbf, 0xaea4, 0xcdbf, 0xae7e,
02309 0xd17d, 0xcdbf, 0xcdbf, 0xd17c, 0xcdbf, 0xcdbf, 0xaea1, 0xabf2,
02310 0xaea2, 0xcd0, 0xd17e, 0xabeb, 0xaea6, 0xabf1, 0xabf0, 0xabef,
02311 0xaea5, 0xcd1, 0xaea7, 0xabea, 0xcdbf, 0xb176, 0xd1a6, 0xd1a6,
02312 0xd1a8, 0xaea8, 0xaea8, 0xd553, 0xd1ac, 0xd1a3, 0xb178, 0xd551,
02313 0xaea9, 0xaeab, 0xd1ae, 0xd552, 0xd1a5, 0xaeac, 0xd1a9, 0xaeaf,
02314 0xd1ab, 0xaeaa, 0xd1aa, 0xd1ad, 0xd1a7, 0xaea9, 0xb179, 0xd1a2,
02315 0xb177, 0xb17a, 0xd555, 0xd55e, 0xb464, 0xb17c, 0xb1a3, 0xb465,
02316 0xd560, 0xb1aa, 0xd8f9, 0xd556, 0xb1a2, 0xb1a5, 0xb17e, 0xd554,
02317 0xd562, 0xd565, 0xd949, 0xd563, 0xd8fd, 0xb1a1, 0xb1a8, 0xb1ac,
02318 0xd55d, 0xd8f8, 0xd561, 0xb17b, 0xd8fa, 0xd564, 0xd8fc, 0xd559,
02319 0xb462, 0xd557, 0xd558, 0xb1a7, 0xb1a6, 0xd55b, 0xb1ab, 0xd55f,
02320 0xb1a4, 0xd55c, 0xb1a9, 0xb466, 0xb463, 0xd8fb, 0xd55a, 0xb17d,
02321 0xb46b, 0xb46f, 0xd940, 0xb751, 0xb46d, 0xd944, 0xb471, 0xdd65,
02322 0xd946, 0xb753, 0xb469, 0xb46c, 0xd947, 0xd948, 0xd94e, 0xb473,
02323 0xb754, 0xd94a, 0xd94f, 0xd943, 0xb75e, 0xb755, 0xb472, 0xd941,
02324 0xd950, 0xb75d, 0xb470, 0xb74e, 0xd94d, 0xb474, 0xd945, 0xd8fe,
02325 0xb46a, 0xd942, 0xd94b, 0xb74d, 0xb752, 0xb467, 0xd94c, 0xb750,
02326 0xb468, 0xb75c, 0xelc3, 0xdd70, 0xdd68, 0xelc2, 0xdd6c, 0xdd6e,
02327 0xdd6b, 0xb75b, 0xdd6a, 0xb75f, 0xelc2, 0xb75a, 0xba40, 0xdd71,
02328 0xelc4, 0xb758, 0xdd69, 0xdd6d, 0xb9fe, 0xb74f, 0xdd66, 0xdd67,
02329 0xba41, 0xb757, 0xb759, 0xb756, 0xdd6f, 0xelc8, 0xelc9, 0xelce,
02330 0xb75d, 0xelc5, 0xb75d, 0xb46e, 0xelc0, 0xb75c, 0xelc5, 0xba45,
02331 0xelc4, 0xba43, 0xba44, 0xelc1, 0xe5aa, 0xb75a, 0xb46e, 0xelc3,
02332 0xbca3, 0xelc7, 0xb75b, 0xbca2, 0xelc6, 0xelc1, 0xelc7, 0xelcd,
02333 0xbca4, 0xb75d, 0xb75d, 0xbca2, 0xe57a, 0xelcf, 0xbca1, 0xbca4, 0xelcc,
02334 0xb75e, 0xe579, 0xe57e, 0xbce, 0xe578, 0xe9a3, 0xe5a9, 0xbca8,
02335 0xbca6, 0xbce, 0xe5a6, 0xe5a2, 0xbcac, 0xe978, 0xbcaa, 0xe5a1,
02336 0xe976, 0xe5a5, 0xe5a8, 0xe57d, 0xbcab, 0xbca5, 0xe977, 0xbecd,
02337 0xe5a7, 0xbca7, 0xbca9, 0xe5a4, 0xbcad, 0xe5a3, 0xe57c, 0xe57b,
02338 0xbecb, 0xe5ab, 0xe97a, 0xece0, 0xbcd0, 0xe9a2, 0xe97e, 0xece1,
02339 0xbcd1, 0xe9a1, 0xe97c, 0xc0b4, 0xcdbf, 0xe979, 0xe97b, 0xc0b5,
02340 0xbcd3, 0xc0b3, 0xbcd2, 0xc0b7, 0xe97d, 0xbecf, 0xefcf, 0xefc7,
02341 0xece7, 0xefc8, 0xece3, 0xc256, 0xece5, 0xece4, 0xc0b6, 0xece2,
02342 0xece6, 0xefd0, 0xefcc, 0xefce, 0xefc9, 0xefca, 0xefcd, 0xefcb,
02343 0xc367, 0xc36a, 0xc369, 0xc368, 0xc461, 0xf44a, 0xc462, 0xf241,
02344 0xc4df, 0xf5cc, 0xc4e0, 0xc574, 0xc5ca, 0xf7d9, 0xf7da, 0xf7db,
02345 0xf9ba, 0xa4e0, 0xc97c, 0xa5b3, 0xa6a6, 0xa6a7, 0xa6a5, 0xa6a8,
02346 0xa7da, 0xa7d9, 0xcdb1, 0xa9cf, 0xa9ce, 0xd1af, 0xb1ad, 0xb1ae,
02347 0xb475, 0xdd72, 0xb760, 0xb761, 0xdd74, 0xdd76, 0xdd75, 0xelc7,
02348 0xelc6, 0xb47d, 0xelc8, 0xbcae, 0xbcd4, 0xc0b8, 0xc257, 0xc257,
02349 0xc0b9, 0xa4e1, 0xcae6, 0xcdb2, 0xa9d1, 0xa9d0, 0xa9d2, 0xabf3,
02350 0xcded, 0xcded, 0xd1b0, 0xaeb0, 0xb1af, 0xb476, 0xd951, 0xa4e2,
02351 0xa47e, 0xa4e3, 0xc97d, 0xa5b7, 0xa5b6, 0xa5b5, 0xa6ab,
02352 0xc9e9, 0xc9eb, 0xa6aa, 0xc9e3, 0xc9e4, 0xc9ea, 0xc9e6, 0xc9e8,
02353 0xa6a9, 0xc9ec, 0xc9ec, 0xc9e5, 0xa7e1, 0xa7ea, 0xa7e8, 0xcacf,
02354 0xcacae, 0xcacf5, 0xa7e6, 0xcacf6, 0xcacf7, 0xcacf3, 0xa7e5, 0xcacf,
02355 0xcacae, 0xa7e3, 0xcacf4, 0xa7e4, 0xa9d3, 0xa7de, 0xcacf1, 0xcac7,
02356 0xa7db, 0xa7ee, 0xcacae, 0xcacf2, 0xa7e0, 0xa7e2, 0xcacae, 0xcac9,
02357 0xcacae, 0xa7ed, 0xa7e7, 0xa7ec, 0xcacae, 0xa7eb, 0xa7dd, 0xa7dc,
02358 0xa7e9, 0xa9e1, 0xcdbf, 0xcdbf, 0xa9dc, 0xa9ef, 0xcdbf, 0xcdbf,
02359 0xcdbf, 0xcdbf, 0xa9ea, 0xcdbf, 0xcdbf, 0xa9e8, 0xcdbf, 0xcdbf,
02360 0xa9d9, 0xcdbf, 0xa9e3, 0xa9e2, 0xcdbf, 0xa9d7, 0xa9d8, 0xa9d6,
02361 0xa9ee, 0xa9e6, 0xa9e0, 0xa9d4, 0xcdbf, 0xa9df, 0xa9de, 0xa9e7,
02362 0xa9f0, 0xcded, 0xa9e4, 0xcdbf, 0xa9da, 0xa9dd, 0xa9de, 0xa9ec,
02363 0xa9ed, 0xa9eb, 0xa9e5, 0xa9e9, 0xa9db, 0xabf4, 0xcda, 0xc41,
02364 0xabf8, 0xabf8, 0xc40, 0xcdbf, 0xabf8, 0xd1b1, 0xaeb1, 0xc43,
02365 0xcded, 0xcded, 0xabf8, 0xcdbf, 0xcdbf, 0xcdbf, 0xcdbf, 0xabf7,
02366 0xabfb, 0xc42, 0xaeb3, 0xcdbf, 0xabf9, 0xc45, 0xcded, 0xabfc,
02367 0xaeb2, 0xabf6, 0xcded, 0xcded, 0xcded, 0xcded, 0xcded, 0xd1b2,
02368 0xc44, 0xcdbf, 0xcdbf, 0xcdbf, 0xcdbf, 0xcdbf, 0xd1b1, 0xabf8,
02369 0xaecc, 0xd1b4, 0xd1c4, 0xaeb6, 0xd566, 0xd1c6, 0xd1c0, 0xd1b7,
02370 0xd1c9, 0xd1ba, 0xaebc, 0xd57d, 0xd1bd, 0xaebe, 0xaeb5, 0xd1cb,
02371 0xd1bf, 0xaeb8, 0xd1b8, 0xd1b5, 0xd1b6, 0xaeb9, 0xd1c5, 0xd1cc,
02372 0xaebb, 0xd1bc, 0xd1bb, 0xaecc, 0xaecc, 0xaecc, 0xaecc, 0xaecc,
02373 0xd1c8, 0xd1c2, 0xaeb7, 0xd1b3, 0xd1ca, 0xd1c1, 0xd1c3, 0xd1c7,
02374 0xd567, 0xb1b7, 0xb1cb, 0xb1ca, 0xb1bf, 0xd579, 0xd575, 0xd572,
02375 0xd5a6, 0xb1ba, 0xb1b2, 0xd577, 0xb4a8, 0xb1b6, 0xd5a1, 0xb1cc,

```

02376 0xb1c9, 0xd57b, 0xd56a, 0xb1c8, 0xd5a3, 0xd569, 0xb1bd, 0xb1c1,
02377 0xd5a2, 0xd573, 0xb1c2, 0xb1bc, 0xd568, 0xb478, 0xd5a5, 0xd571,
02378 0xb1c7, 0xd574, 0xd5a4, 0xb1c6, 0xd952, 0xb1b3, 0xd56f, 0xb1b8,
02379 0xb1c3, 0xb1be, 0xd578, 0xd56e, 0xd56c, 0xd57e, 0xb1b0, 0xb1c4,
02380 0xb1b4, 0xb477, 0xd57c, 0xb1b5, 0xb1b1, 0xb1c0, 0xb1bb, 0xb1b9,
02381 0xd570, 0xb1c5, 0xd56d, 0xd57a, 0xd576, 0xd954, 0xd953, 0xd56b,
02382 0xd964, 0xb47a, 0xd96a, 0xd959, 0xd967, 0xdd77, 0xb47d, 0xd96b,
02383 0xd96e, 0xb47c, 0xd95c, 0xd96d, 0xd96c, 0xb47e, 0xd955, 0xb479,
02384 0xb4a3, 0xb4a1, 0xd969, 0xd95f, 0xb4a5, 0xd970, 0xd968, 0xd971,
02385 0xb4ad, 0xb4ab, 0xd966, 0xd965, 0xd963, 0xd95d, 0xb4a4, 0xb4a2,
02386 0xd1b9, 0xd956, 0xddb7, 0xd957, 0xb47b, 0xb4aa, 0xdd79, 0xb4a6,
02387 0xb4a7, 0xd958, 0xd96f, 0xdd78, 0xd960, 0xd95b, 0xb4a9, 0xd961,
02388 0xd95e, 0xb4ae, 0xb770, 0xdd7c, 0xddb1, 0xddb6, 0xddaa, 0xb76c,
02389 0xddbb, 0xb769, 0xdd7a, 0xdd7b, 0xb762, 0xb76b, 0xdda4, 0xb76e,
02390 0xb76f, 0xdda5, 0xddb2, 0xddb8, 0xb76a, 0xb764, 0xdda3, 0xdd7d,
02391 0xddba, 0xdda8, 0xdda9, 0xdd7e, 0xddb4, 0xddab, 0xddb5, 0xddad,
02392 0xb765, 0xeld9, 0xb768, 0xb766, 0xddb9, 0xddb0, 0xddac, 0xdda1,
02393 0xba53, 0xbdaf, 0xb76d, 0xdda7, 0xdda6, 0xb767, 0xb763, 0xeee,
02394 0xddb3, 0xddae, 0xdda2, 0xeee9, 0xelda, 0xeee5, 0xeee, 0xba51,
02395 0xb4ac, 0xeee, 0xba4c, 0xba4b, 0xelf1, 0xeldb, 0xeee8, 0xeldc,
02396 0xeee7, 0xba4f, 0xeee, 0xd962, 0xelf2, 0xeee3, 0xba52, 0xeee,
02397 0xbcaf, 0xelf0, 0xeee, 0xba54, 0xe5ad, 0xbcb0, 0xe5ae, 0xelfd,
02398 0xeee0, 0xeldd, 0xeee2, 0xelde, 0xelf3, 0xba4e, 0xbcb1, 0xba50,
02399 0xba55, 0xeee1, 0xeee, 0xeee6, 0xe5b1, 0xba4a, 0xbcb4, 0xe9aa,
02400 0xe5b6, 0xe5b5, 0xe5b7, 0xe5b4, 0xbcb5, 0xbcb, 0xbcb8, 0xbcb9,
02401 0xe5af, 0xe5b2, 0xe5bc, 0xbcc1, 0xbcbf, 0xe5b3, 0xd95a, 0xbcb2,
02402 0xe5b9, 0xe5b0, 0xbcc2, 0xe5b8, 0xba4d, 0xbcb7, 0xeee4, 0xbcb,
02403 0xbcb, 0xbcc0, 0xbcbd, 0xbcbc, 0xbcb6, 0xe5bb, 0xbcb3, 0xbcc3,
02404 0xbcd8, 0xbcd9, 0xe9a9, 0xbce2, 0xbcd, 0xbcd6, 0xbcd, 0xe9ab,
02405 0xbcd, 0xbcd5, 0xbcd, 0xe9a8, 0xc0bb, 0xbcd7, 0xbcd, 0xc0ba,
02406 0xe9a7, 0xe9a6, 0xbce0, 0xbce1, 0xe9a5, 0xe9a4, 0xc0bc, 0xe9ae,
02407 0xbcd, 0xe9ac, 0xc0bd, 0xc0c2, 0xeca, 0xccc, 0xc0bf, 0xccc,
02408 0xccc9, 0xccc, 0xc0c0, 0xc0c3, 0xccc8, 0xc0be, 0xc0c1, 0xc259,
02409 0xe9ad, 0xc258, 0xc25e, 0xefd4, 0xc25c, 0xc25d, 0xefd7, 0xefd3,
02410 0xc25a, 0xefd1, 0xc36b, 0xefd5, 0xefd6, 0xefd2, 0xc25b, 0xf242,
02411 0xf245, 0xf246, 0xf244, 0xf247, 0xc36c, 0xf243, 0xf44e, 0xc464,
02412 0xf44d, 0xf44c, 0xf44b, 0xc463, 0xc465, 0xf5cd, 0xc4e2, 0xc4e1,
02413 0xf6e1, 0xf6e0, 0xf6e3, 0xc5cb, 0xc575, 0xf7dd, 0xf6e2, 0xf7dc,
02414 0xc5cd, 0xc5cc, 0xc5f3, 0xf8a9, 0xf8ef, 0xa4e4, 0xd972, 0xe9af,
02415 0xa6ac, 0xcacf, 0xa7f1, 0xa7ef, 0xa7f0, 0xcc1, 0xa9f1, 0xac46,
02416 0xccc7, 0xccc8, 0xac47, 0xd1ce, 0xaec4, 0xaec5, 0xd1cd, 0xb1d3,
02417 0xb1cf, 0xd5a7, 0xb1d6, 0xb1d5, 0xb1ce, 0xb1d1, 0xb1d4, 0xb1d0,
02418 0xd976, 0xb1cd, 0xb4af, 0xb4b1, 0xb4b2, 0xd975, 0xd978, 0xb4b0,
02419 0xd973, 0xd977, 0xd974, 0xb771, 0xddbc, 0xba56, 0xelf4, 0xbce3,
02420 0xbcc4, 0xbcd7, 0xbcd5, 0xbcc5, 0xbcc6, 0xe5bf, 0xe5be, 0xe5c0, 0xe9b1,
02421 0xe9b0, 0xccc, 0xccc, 0xc0c4, 0xc0c5, 0xf248, 0xa4e5, 0xd979,
02422 0xb4b4, 0xb4b3, 0xddbd, 0xefd8, 0xc4e3, 0xf7de, 0xa4e6, 0xaec6,
02423 0xb1d8, 0xb1d7, 0xd97a, 0xd97b, 0xb772, 0xelf5, 0xba57, 0xe9b2,
02424 0xa4e7, 0xa5b8, 0xa9f2, 0xcc2, 0xccc9, 0xac48, 0xb1d9, 0xd97c,
02425 0xb4b5, 0xb773, 0xe5c1, 0xe5c2, 0xccc0, 0xc25f, 0xf8f0, 0xa4e8,
02426 0xcc3, 0xa9f3, 0xac49, 0xccc, 0xaec7, 0xd1d2, 0xd1d0, 0xd1d1,
02427 0xaec8, 0xd1cf, 0xb1db, 0xb1dc, 0xd5a8, 0xb1dd, 0xb1da, 0xd97d,
02428 0xd97e, 0xddbe, 0xba59, 0xba58, 0xccc1, 0xefd9, 0xf24a, 0xf249,
02429 0xf44f, 0xc95e, 0xac4a, 0xa4e9, 0xa5b9, 0xa6ae, 0xa6ad, 0xa6af,
02430 0xa6b0, 0xc9ee, 0xc9ed, 0xcacf, 0xa7f2, 0xcacf, 0xcafa, 0xcacf,
02431 0xcacf, 0xa9f4, 0xccc9, 0xccc5, 0xccc, 0xa9fb, 0xa9fb, 0xccc,
02432 0xccc6, 0xccc, 0xa9f8, 0xccc8, 0xccc4, 0xccc, 0xa9fe, 0xccc,
02433 0xa9f7, 0xccc, 0xa9fa, 0xa9fc, 0xccc0, 0xccc, 0xccc7, 0xa9f6,
02434 0xa9f5, 0xa9fd, 0xccc, 0xccc5, 0xac50, 0xac4d, 0xccc, 0xccc,
02435 0xac53, 0xac4b, 0xccc0, 0xac4e, 0xac51, 0xccc3, 0xac4c, 0xccc8,
02436 0xac4f, 0xac52, 0xccc, 0xccc2, 0xccc6, 0xccc, 0xccc, 0xccc7,
02437 0xccc4, 0xaed0, 0xaec9, 0xaec, 0xaecf, 0xd1d5, 0xaeca, 0xd1d3,
02438 0xaec, 0xaecb, 0xd1d6, 0xaecd, 0xd5ac, 0xb1df, 0xd5ab, 0xd5ad,
02439 0xb1de, 0xb1e3, 0xd1d4, 0xd5aa, 0xd5ae, 0xb1e0, 0xd5a9, 0xb1e2,
02440 0xb1e1, 0xd9a7, 0xd9a2, 0xb4b6, 0xb4ba, 0xb4b7, 0xd9a5, 0xd9a8,
02441 0xb4b8, 0xb4b9, 0xb4be, 0xddc7, 0xd9a6, 0xb4bc, 0xd9a3, 0xd9a1,
02442 0xb4bd, 0xd9a4, 0xb779, 0xddbf, 0xb776, 0xb777, 0xb775, 0xddc4,
02443 0xddc3, 0xddc0, 0xb77b, 0xddc2, 0xb4bb, 0xddc6, 0xddc1, 0xb778,
02444 0xb774, 0xb77a, 0xddc5, 0xba5c, 0xelf8, 0xelf7, 0xelf6, 0xba5a,
02445 0xba5b, 0xe5c5, 0xe5c8, 0xbcc8, 0xbcc7, 0xe5c9, 0xe5c4, 0xbcca,
02446 0xe5c6, 0xbcc9, 0xe5c3, 0xe5c7, 0xbce9, 0xbce6, 0xe9bb, 0xe9ba,
02447 0xe9b9, 0xe9b4, 0xe9b5, 0xbce7, 0xbce4, 0xbce8, 0xe9b3, 0xbce5,
02448 0xe9b6, 0xe9b7, 0xe9bc, 0xe9b8, 0xccc2, 0xc0c7, 0xefdc, 0xc0c6,
02449 0xefda, 0xefdb, 0xc260, 0xc36e, 0xf24b, 0xc36d, 0xf451, 0xf452,
02450 0xc466, 0xf450, 0xc4e4, 0xf7df, 0xc5ce, 0xf8aa, 0xf8ab, 0xa4ea,
02451 0xa6b1, 0xa6b2, 0xa7f3, 0xccc1, 0xac54, 0xaed1, 0xb1e4, 0xb0d2,
02452 0xb4bf, 0xb4c0, 0xb3cc, 0xd9a9, 0xb77c, 0xelfa, 0xelf9, 0xa4eb,
02453 0xa6b3, 0xccc2, 0xaa42, 0xaa41, 0xccc9, 0xcffa, 0xd1d7, 0xd1d8,
02454 0xaed2, 0xaed3, 0xaed4, 0xd5af, 0xb1e6, 0xb4c2, 0xb4c1, 0xddc8,
02455 0xdf7a, 0xelfb, 0xe9bd, 0xc261, 0xc467, 0xa4ec, 0xa5bc, 0xa5bd,
02456 0xa5bb, 0xa5be, 0xa5ba, 0xa6b6, 0xc9f6, 0xa6b5, 0xa6b7, 0xc9f1,
02457 0xc9f0, 0xc9f3, 0xc9f2, 0xc9f5, 0xa6b4, 0xc9ef, 0xc9f4, 0xcafd,
02458 0xa7fd, 0xcacf, 0xc443, 0xa7fc, 0xc447, 0xc442, 0xc445, 0xa7f5,
02459 0xa7f6, 0xa7f7, 0xa7f8, 0xa840, 0xc441, 0xa7fa, 0xa841, 0xc440,
02460 0xc446, 0xa7f9, 0xc444, 0xa7fb, 0xa7fa, 0xa7fe, 0xaa57, 0xccc4,
02461 0xaa43, 0xaa4d, 0xaa4e, 0xaa46, 0xaa58, 0xaa4c, 0xccc, 0xaa53,
02462 0xccc7, 0xaa49, 0xccc6, 0xccc7, 0xccc, 0xccc8, 0xaa56, 0xccc4,

```

02463 0xaa51, 0xaa4f, 0xcce5, 0xcce3, 0xccdb, 0xcd3, 0ccda, 0xaa4a,
02464 0xaa50, 0xaa44, 0xccde, 0ccdd, 0ccd5, 0xaa52, 0ccel, 0ccd6,
02465 0xaa55, 0xcce8, 0xaa45, 0xaa4c, 0ccd9, 0cce2, 0xaa54, 0xaa47,
02466 0xaa4b, 0xcce0, 0xcf5b, 0xac5c, 0xac69, 0xcf56, 0xcf4c, 0xac62,
02467 0xcf4a, 0xac5b, 0xcf45, 0xac65, 0xcf52, 0xcfe, 0xcf41, 0xcf44,
02468 0xcfeb, 0xcf51, 0xcf61, 0xac60, 0xcf46, 0xcf58, 0xcfe, 0xcf5f,
02469 0xcf60, 0xcf63, 0xcf5a, 0xcf4b, 0xcf53, 0xac66, 0xac59, 0xac61,
02470 0xac6d, 0xac56, 0xac58, 0xcf43, 0xac6a, 0xac63, 0xcf5d, 0xcf40,
02471 0xac6c, 0xac67, 0xcf49, 0xac6b, 0xcf50, 0xcf48, 0xac64, 0xcf5c,
02472 0xcf54, 0xac5e, 0xcf62, 0xcf47, 0xac5a, 0xcf59, 0xcf4f, 0xac5f,
02473 0xcf55, 0xac57, 0xcfc, 0xac68, 0xae3, 0xac5d, 0xcf4e, 0xcf4d,
02474 0xcf42, 0xcf5e, 0xcf57, 0xac55, 0xd1ec, 0xaea, 0xd1ed, 0xd1e1,
02475 0xaedf, 0xaeeb, 0xd1da, 0xd1e3, 0xd1eb, 0xd1d9, 0xd1f4, 0xaed5,
02476 0xd1f3, 0xd1ee, 0xd1ef, 0xaedd, 0xae8, 0xd1e5, 0xd1e6, 0xd1f0,
02477 0xd1e7, 0xd1e2, 0xd1dc, 0xd1dd, 0xd1ea, 0xd1e4, 0xaed6, 0xaeda,
02478 0xd1f2, 0xd1de, 0xae6, 0xae2, 0xae5, 0xaeec, 0xaedb, 0xae7,
02479 0xd1e9, 0xae9, 0xaed8, 0xaed7, 0xd1db, 0xd1df, 0xae0, 0xd1f1,
02480 0xd1e8, 0xd1e0, 0xae4, 0xae1, 0xaed9, 0xaedc, 0xd5c4, 0xd5b4,
02481 0xd5b5, 0xd5b9, 0xd5c8, 0xd5c5, 0xd5be, 0xd5bd, 0xb1ed, 0xd5c1,
02482 0xd5d0, 0xd5b0, 0xd5d1, 0xd5c3, 0xd5d5, 0xd5c9, 0xb1ec, 0xd5c7,
02483 0xb1e7, 0xb1f, 0xb1f2, 0xb1f6, 0xb1f5, 0xd5b1, 0xd5ce, 0xd5d4,
02484 0xd5cc, 0xd5d3, 0xd5c0, 0xd5b2, 0xd5d2, 0xd5c2, 0xb1ea, 0xb1f7,
02485 0xd5cb, 0xb1f0, 0xd5ca, 0xd5b3, 0xb1f8, 0xb1fa, 0xd5cd, 0xb1fb,
02486 0xb1e9, 0xd5ba, 0xd5cf, 0xb1ef, 0xb1f9, 0xd5bc, 0xd5c6, 0xd5b7,
02487 0xd5bb, 0xb1f4, 0xd5b6, 0xb1e8, 0xb1f1, 0xb1ee, 0xd5bf, 0xaede,
02488 0xd9c0, 0xb1eb, 0xb1f3, 0xd9c3, 0xd9d9, 0xd9ce, 0xb4d6, 0xb4d1,
02489 0xd9bd, 0xb4d2, 0xd9cd, 0xd9c6, 0xd9d3, 0xb4ce, 0xd9ab, 0xd9d5,
02490 0xb4c4, 0xd9b3, 0xb4c7, 0xb4c6, 0xb4d7, 0xd9ad, 0xd9cf, 0xd9d0,
02491 0xb4c9, 0xb4c5, 0xd9bb, 0xb4d0, 0xd9b6, 0xd9d1, 0xb4cc, 0xd9c9,
02492 0xd9bd, 0xd9b0, 0xd9b5, 0xd9af, 0xb4cb, 0xd9c2, 0xdd, 0xd9b1,
02493 0xb4cf, 0xd9ba, 0xd9d2, 0xb4ca, 0xd9b7, 0xd9b4, 0xd9c5, 0xb4cd,
02494 0xb4c3, 0xb4d9, 0xd9c8, 0xd9c7, 0xd9ac, 0xb4c8, 0xd9d4, 0xd9bc,
02495 0xd9be, 0xd9cb, 0xd9ca, 0xd9aa, 0xb4d3, 0xb4d5, 0xd9b2, 0xd9b9,
02496 0xd9c1, 0xb4d4, 0xd9b8, 0xd9c4, 0xd9d7, 0xd9cc, 0xd9d8, 0xd9ae,
02497 0xddf2, 0xb7a6, 0xddf0, 0xdddb, 0xdde0, 0xddd9, 0xddec, 0xddcb,
02498 0xddd2, 0xddea, 0xddf4, 0xdddc, 0xddcf, 0xdde2, 0xdde7, 0xddd3,
02499 0xdde4, 0xddd0, 0xddd7, 0xddd8, 0xb7a8, 0xddeb, 0xdde9, 0xddcc,
02500 0xddee, 0xddef, 0xddf1, 0xb7ac, 0xb7a4, 0xd5b8, 0xddd4, 0xdde6,
02501 0xddd5, 0xb7a1, 0xb7b1, 0xdded, 0xb7af, 0xb7ab, 0xddca, 0xb7a3,
02502 0xddcd, 0xb7b0, 0xdddd, 0ddc9, 0xb7a9, 0xdde1, 0xddd1, 0xb7aa,
02503 0xddda, 0xb77e, 0xb4d8, 0xdde3, 0xd9bf, 0xddce, 0xdde8, 0xb7a5,
02504 0xdde5, 0xb7a2, 0xdddf, 0xb7ad, 0xddd6, 0xddf3, 0xb7a7, 0xddec6,
02505 0xb7ae, 0xe24a, 0xe248, 0xe25e, 0xe246, 0xe258, 0xb77d, 0xba5f,
02506 0xe242, 0xe25d, 0xe247, 0xe255, 0xba64, 0xba5d, 0xe25b, 0xe240,
02507 0xe25a, 0xba6f, 0xe251, 0xe261, 0xba6d, 0xe249, 0xba5e, 0xe24b,
02508 0xe259, 0xba67, 0xe244, 0xba6b, 0xba61, 0xe24d, 0xe243, 0xe1fc,
02509 0xe257, 0xba68, 0xe260, 0xe1fd, 0xba65, 0xe253, 0xba66, 0xe245,
02510 0xe250, 0xe24c, 0xe24e, 0xba60, 0xe25f, 0xba6e, 0xe24f, 0xe262,
02511 0xe1fe, 0xe254, 0xba63, 0xba6c, 0xba6a, 0xe241, 0xe256, 0xba69,
02512 0xba62, 0xe252, 0xe25c, 0xe5d5, 0xe5d1, 0xe5cd, 0xe5e1, 0xe5de,
02513 0xbccd, 0xe5e5, 0xe5d4, 0xbcd8, 0xe5db, 0xe5d0, 0xe5da, 0xbcd5,
02514 0xe5ee, 0xe5eb, 0xe5dd, 0xe5ce, 0xe5e2, 0xe5e4, 0xbcd1, 0xe5d8,
02515 0xe5d3, 0xe5ca, 0xbcce, 0xbcd6, 0xe5e7, 0xbcd7, 0xe5cb, 0xe5ed,
02516 0xe5e0, 0xe5e6, 0xbcd4, 0xe5e3, 0xe5ea, 0xbcd9, 0xbcd3, 0xe5dc,
02517 0xe5cf, 0xe5ef, 0xe5cc, 0xe5e8, 0xbcd0, 0xe5d6, 0xe5d7, 0xbccf,
02518 0xbccc, 0xe5d2, 0xbcd2, 0xbccb, 0xe5e9, 0xe5ec, 0xe5d9, 0xe9ca,
02519 0xe9c2, 0xe9be, 0xbef6, 0xbbeb, 0xbef0, 0xbbec, 0xe9cc, 0xe9d7,
02520 0xbeea, 0xe9c4, 0xe9cd, 0xe5df, 0xe9ce, 0xbef1, 0xe9dd, 0xbef5,
02521 0xbef8, 0xe9c0, 0xbef4, 0xe9db, 0xe9dc, 0xe9d2, 0xe9d1, 0xe9c9,
02522 0xe9d3, 0xe9da, 0xbef9, 0xbef, 0xbef, 0xe9cb, 0xe9c8, 0xe9c5,
02523 0xe9d8, 0xbef7, 0xe9d6, 0xbef3, 0xbef2, 0xe9d0, 0xe9bf, 0xe9c1,
02524 0xe9c3, 0xe9d5, 0xe9cf, 0xbbee, 0xe9c6, 0xe9d4, 0xe9c7, 0xc0cf,
02525 0xed45, 0xc0c8, 0xecf5, 0xed41, 0xc0ca, 0xed48, 0xecfc, 0xecf7,
02526 0xed49, 0xecf3, 0xecfe, 0xc0d1, 0xed44, 0xed4a, 0xecfd, 0xc0c9,
02527 0xed40, 0xecf4, 0xc0d0, 0xed47, 0xecf9, 0xc0cc, 0xecfb, 0xecf8,
02528 0xc0d2, 0xecfa, 0xc0cb, 0xc0ce, 0xed43, 0xecf6, 0xed46, 0xed42,
02529 0xc263, 0xefe7, 0xc268, 0xc269, 0xc262, 0xefe6, 0xefe3, 0xefe4,
02530 0xc266, 0xefde, 0xefe2, 0xc265, 0xefdf, 0xc267, 0xc264, 0xefdd,
02531 0xefe1, 0xefe5, 0xf251, 0xf24e, 0xf257, 0xf256, 0xf254, 0xf24f,
02532 0xc372, 0xf250, 0xc371, 0xc0cd, 0xf253, 0xc370, 0xf258, 0xf252,
02533 0xf24d, 0xefe0, 0xc36f, 0xf24c, 0xf456, 0xf455, 0xf255, 0xc468,
02534 0xf459, 0xf45a, 0xf454, 0xf458, 0xf453, 0xf5d1, 0xf457, 0xc4e7,
02535 0xc4e5, 0xf5cf, 0xf5d2, 0xf5ce, 0xf5d0, 0xc4e6, 0xf6e5, 0xf6e6,
02536 0xc576, 0xf6e4, 0xf7e2, 0xc5cf, 0xf7e0, 0xf7e1, 0xf8ac, 0xc656,
02537 0xf8f3, 0xf8f1, 0xf8f2, 0xf8f4, 0xf9bb, 0xa4ed, 0xa6b8, 0xaa59,
02538 0xcce9, 0xcf64, 0xd1f5, 0xd1f7, 0xd1f6, 0xd1f8, 0xb1fd, 0xd5d7,
02539 0xd1f9, 0xd5d6, 0xd5d8, 0xd5d9, 0xd9da, 0xb4db, 0xd9db, 0xd9dd,
02540 0xb4dc, 0xb4da, 0xd9dc, 0xddfa, 0xddf8, 0xddf7, 0xddf6, 0xddf5,
02541 0xb7b2, 0xddf9, 0xba70, 0xe263, 0xe265, 0xba71, 0xe264, 0xbcd3,
02542 0xbcd4, 0xe5f0, 0xe9df, 0xe9de, 0xe9e0, 0xbef9, 0xed4b, 0xc0d3,
02543 0xefe8, 0xc26a, 0xf259, 0xc577, 0xa4ee, 0xa5bf, 0xa6b9, 0xa842,
02544 0xaa5a, 0xaa5b, 0xac6e, 0xd1fa, 0xb7b3, 0xe6d1, 0xbefa, 0xc26b,
02545 0xa4ef, 0xa6ba, 0xcceb, 0xaa5c, 0xccea, 0xcf65, 0xac6f, 0xcf66,
02546 0xac70, 0xd1fc, 0xaeee, 0xaeeed, 0xd5de, 0xd5dc, 0xd5dd, 0xd5db,
02547 0xd5da, 0xd9de, 0xd9e1, 0xb4de, 0xd9df, 0xb4dd, 0xd9e0, 0xddfb,
02548 0xe266, 0xe267, 0xe268, 0xe5f3, 0xe5f2, 0xbcdc, 0xe5f1, 0xe5f4,
02549 0xe9e1, 0xe9e2, 0xe9e3, 0xed4c, 0xc0d4, 0xc26c, 0xf25a, 0xc4e8,

```

02550 0xc95f, 0xac71, 0xcf67, 0xaeef, 0xb1fe, 0xb4df, 0xd9e2, 0xb7b5,
02551 0xb7b4, 0xe269, 0xe26a, 0xbccd, 0xbcd, 0xe9e5, 0xe9e4, 0xefe9,
02552 0xf7e3, 0xa4f0, 0xa960, 0xa5c0, 0xa843, 0xcb48, 0xac72, 0xb7b6,
02553 0xa4f1, 0xcf68, 0xac73, 0xcf69, 0xc0d5, 0xa4f2, 0xccec, 0xcf6a,
02554 0xd242, 0xd241, 0xd1fe, 0xd1fd, 0xd243, 0xd240, 0xb240, 0xb241,
02555 0xb4e0, 0xd9e3, 0xd9e4, 0xd9e5, 0xde41, 0xde42, 0xde40, 0xddfd,
02556 0xddfe, 0xb7b7, 0xe26b, 0xe5f7, 0xe5f6, 0xe5f5, 0xe5f8, 0xe9e7,
02557 0xe9e6, 0xbefb, 0xe9e8, 0xc0d6, 0xed4d, 0xefea, 0xf25b, 0xf6e7,
02558 0xa4f3, 0xa5c2, 0xa5c1, 0xaa5d, 0xc961, 0xc97e, 0xa6bb, 0xc9f7,
02559 0xcb49, 0xcb4a, 0xaa5e, 0xcced, 0xac74, 0xcf6b, 0xcf6c, 0xae0,
02560 0xae4, 0xd244, 0xae3, 0xae1, 0xae2, 0xd5df, 0xb242, 0xb4e3,
02561 0xb4e1, 0xb4e2, 0xd9e6, 0xba72, 0xa4f4, 0xc9a1, 0xa5c3, 0xc9a4,
02562 0xa5c6, 0xc9a3, 0xa5c5, 0xa5c4, 0xa844, 0xc9a2, 0xc9f8, 0xc9fc,
02563 0xc9fe, 0xca40, 0xa6c5, 0xa6c6, 0xc9fb, 0xa6c1, 0xc9f9, 0xc9fd,
02564 0xa6c2, 0xa6b8, 0xa6bd, 0xa6be, 0xa6c4, 0xc9fa, 0xa6bc, 0xa845, 0xa6bf,
02565 0xa6c0, 0xa6c3, 0xcb5b, 0xcb59, 0xcb4c, 0xa851, 0xcb53, 0xa84c,
02566 0xcb4d, 0xcb55, 0xcb52, 0xa84f, 0xcb51, 0xa856, 0xcb5a, 0xa858,
02567 0xa85a, 0xcb4b, 0xcb4d, 0xa84d, 0xcb5c, 0xa854, 0xa857, 0xcd45, 0xa847,
02568 0xa85e, 0xa855, 0xcb4e, 0xa84a, 0xa859, 0xcb56, 0xa848, 0xa849,
02569 0xcd43, 0xcb4f, 0xa850, 0xa85b, 0xcb5d, 0xcb50, 0xa84e, 0xa853,
02570 0xccee, 0xcdee, 0xcde5, 0xcde7, 0xa852, 0xa85d, 0xa846, 0xcb54, 0xa84b,
02571 0xcb58, 0xcd44, 0xaa6a, 0xaa7a, 0xccf5, 0xaa71, 0xcd4b, 0xaa62,
02572 0xaa65, 0xcd42, 0xccf3, 0xccf7, 0xaa6d, 0xaa6f, 0xccfa, 0xaa76,
02573 0xaa68, 0xaa66, 0xaa67, 0xaa75, 0xcd47, 0xaa70, 0xccf9, 0xccfb,
02574 0xaa6e, 0xaa73, 0xccfc, 0xcd4a, 0xac75, 0xaa79, 0xaa63, 0xcd49,
02575 0xcd4d, 0xccf8, 0xcd4f, 0xcd40, 0xaa6c, 0xccf4, 0xaa6b, 0xaa7d,
02576 0xaa72, 0xccf2, 0xccf5, 0xf75, 0xaa78, 0xaa7c, 0xcd41, 0xcd4d, 0xaa7e,
02577 0xaa77, 0xaa69, 0xaa5f, 0xaa64, 0xccf6, 0xaa60, 0xcd4e, 0xccf0,
02578 0xccef, 0xccfd, 0xccf1, 0xaa7b, 0xae5, 0xaa74, 0xccfe, 0xaa61,
02579 0xaca6, 0xaca6, 0xcfd7c, 0xcfa1, 0xcfa4, 0xc77, 0xcfa7, 0xcfaa,
02580 0xcfac, 0xcf74, 0xac76, 0xac7b, 0xd249, 0xacad, 0xcfa5, 0xcfad,
02581 0xcf7b, 0xcf73, 0xd264, 0xac7e, 0xcfa2, 0xcf78, 0xcf7a, 0xaca5,
02582 0xcf7d, 0xac7d, 0xcf70, 0xcfa8, 0xcfab, 0xac7a, 0xaca8, 0xcfd6d,
02583 0xacaa, 0xac78, 0xaca, 0xcfa9, 0xcfd6f, 0xacab, 0xd25e, 0xcd48,
02584 0xac7c, 0xac77, 0xcf76, 0xcfd6e, 0xacad, 0xaca4, 0xcfa3, 0xaca9,
02585 0xaca7, 0xcfd79, 0xcfa1, 0xcfd71, 0xaca2, 0xaca3, 0xcfd72, 0xcfa6,
02586 0xac79, 0xcfd7e, 0xd24c, 0xaefd, 0xaf43, 0xd255, 0xd25b, 0xd257,
02587 0xd24a, 0xd24d, 0xd246, 0xd247, 0xaf4a, 0xae4a, 0xd256, 0xd25f,
02588 0xaf45, 0xae4f, 0xaf40, 0xd24e, 0xaf42, 0xd24f, 0xd259, 0xaf44,
02589 0xd268, 0xd248, 0xae4c, 0xae4b, 0xaf48, 0xd245, 0xd266, 0xd25a,
02590 0xd267, 0xd261, 0xd253, 0xd262, 0xd25c, 0xd265, 0xd263, 0xaf49,
02591 0xd254, 0xae4f, 0xae48, 0xaf41, 0xaf47, 0xd260, 0xaf46, 0xd251,
02592 0xb243, 0xd269, 0xd250, 0xd24b, 0xae4e, 0xaf4b, 0xae4f, 0xd258,
02593 0xd25d, 0xb265, 0xd5e1, 0xd5e5, 0xb252, 0xb250, 0xb247, 0xd5e3,
02594 0xd5e2, 0xb25b, 0xd5e8, 0xb255, 0xd5fa, 0xd647, 0xb244, 0xd5f7,
02595 0xd5f0, 0xb267, 0xd5e0, 0xd5fc, 0xb264, 0xb258, 0xb263, 0xb24e,
02596 0xd5ec, 0xd5fe, 0xd5f6, 0xb24f, 0xb249, 0xd645, 0xd5fd, 0xd640,
02597 0xb251, 0xb259, 0xd642, 0xd5ea, 0xd5fb, 0xd5ef, 0xd644, 0xb25e,
02598 0xb246, 0xb25c, 0xd5f4, 0xd5f2, 0xd5f3, 0xb253, 0xd5ee, 0xd5ed,
02599 0xb248, 0xd5e7, 0xd646, 0xb24a, 0xd5f1, 0xb268, 0xb262, 0xd5e6,
02600 0xb25f, 0xb25d, 0xb266, 0xd5f8, 0xb261, 0xd252, 0xd5f9, 0xb260,
02601 0xd641, 0xb245, 0xd5f5, 0xb257, 0xd5e9, 0xb256, 0xb254, 0xb24c,
02602 0xb24b, 0xd9e7, 0xd643, 0xd5eb, 0xd9fc, 0xb24d, 0xb541, 0xb25a,
02603 0xb4ee, 0xd9f6, 0xb4fc, 0xd9ea, 0xb4eb, 0xb4e7, 0xda49, 0xb4ed,
02604 0xb4f1, 0xb4ec, 0xb4f5, 0xda4d, 0xda44, 0xd9f1, 0xb4fa, 0xb4f4,
02605 0xd9fd, 0xb4ea, 0xda4a, 0xda43, 0xb4e8, 0xd9f7, 0xb4f7, 0xda55,
02606 0xda56, 0xb4e5, 0xda48, 0xb4f9, 0xd9fb, 0xd9ed, 0xd9ee, 0xb4fd,
02607 0xd9f2, 0xd9f9, 0xd9f3, 0xb4fb, 0xb544, 0xd9ef, 0xd9e8, 0xd9e9,
02608 0xd9eb, 0xb4ea, 0xd9f8, 0xb4f8, 0xb542, 0xd9fa, 0xda53, 0xda4b,
02609 0xb4e6, 0xb4f2, 0xb4f0, 0xda57, 0xb4ef, 0xda41, 0xd9f4, 0xd9f4,
02610 0xd9fe, 0xb547, 0xda45, 0xda42, 0xd9f0, 0xb543, 0xda4f, 0xda4c,
02611 0xda54, 0xb4e9, 0xda40, 0xb546, 0xda47, 0xb4f3, 0xb4f6, 0xda46,
02612 0xb545, 0xd9f5, 0xd5e4, 0xda50, 0xda4e, 0xda52, 0xd9ec, 0xb540,
02613 0xde61, 0xde60, 0xde46, 0xb7bd, 0xde5f, 0xde49, 0xde4a, 0xb7c7,
02614 0xde68, 0xb7c2, 0xde5e, 0xde43, 0xb7c8, 0xb7be, 0xde52, 0xde48,
02615 0xde4b, 0xde63, 0xb7b8, 0xde6a, 0xde62, 0xb7c1, 0xde57, 0xb7cc,
02616 0xb7cb, 0xb7c5, 0xde69, 0xb7b9, 0xde55, 0xde4c, 0xde59, 0xde65,
02617 0xb7cd, 0xb7bb, 0xde54, 0xde4d, 0xb7c4, 0xb7c3, 0xde50, 0xde5a,
02618 0xde64, 0xde47, 0xde51, 0xb7bc, 0xde5b, 0xb7c9, 0xb7c0, 0xde4e,
02619 0xb7bf, 0xde45, 0xde53, 0xde67, 0xb4fe, 0xbab0, 0xde56, 0xe26c,
02620 0xde58, 0xde66, 0xb7c6, 0xde4f, 0xb7ba, 0xb7ca, 0xbcf0, 0xde44,
02621 0xde5d, 0xde5c, 0xe2aa, 0xbaad, 0xe27d, 0xe2a4, 0xbaa2, 0xe26e,
02622 0xbaaaf, 0xba77, 0xe26d, 0xe2b0, 0xbab1, 0xe271, 0xe2a3, 0xe273,
02623 0xe2b3, 0xe2af, 0xba75, 0xbaa1, 0xe653, 0xbaae, 0xba7d, 0xe26f,
02624 0xe2ae, 0xbaa3, 0xe2ab, 0xe2b8, 0xe275, 0xe27e, 0xe2b6, 0xe2ac,
02625 0xba7c, 0xe27c, 0xba76, 0xba74, 0xbaa8, 0xe27a, 0xe277, 0xe278,
02626 0xe2b2, 0xe2b7, 0xe2b5, 0xba7a, 0xe2b9, 0xba7e, 0xbaa7, 0xe270,
02627 0xe5fa, 0xe279, 0xba78, 0xbaa, 0xbaa9, 0xba7b, 0xe2a5, 0xe274,
02628 0xbaaa, 0xe2a7, 0xbaa4, 0xbaa6, 0xba73, 0xe2a9, 0xe2a1, 0xe272,
02629 0xbaa5, 0xe2b1, 0xe2b4, 0xe27b, 0xe2a8, 0xba79, 0xbcdf, 0xe2a6,
02630 0xe5f9, 0xe2ad, 0xe276, 0xe644, 0xe64e, 0xbce2, 0xe64d, 0xe659,
02631 0xbce4, 0xe64b, 0xe64f, 0xbcef, 0xe646, 0xbce7, 0xe652, 0xe9f0,
02632 0xbcf3, 0xbcf2, 0xe654, 0xe643, 0xe65e, 0xbced, 0xbce3, 0xe657,
02633 0xe65b, 0xe660, 0xe655, 0xe649, 0xbce6, 0xbce9, 0xbcf1, 0xbcec,
02634 0xe64c, 0xe2a2, 0xe648, 0xe65f, 0xbce8, 0xbceb, 0xe661, 0xbce0,
02635 0xe656, 0xe5fb, 0xe65c, 0xc0df, 0xe64a, 0xbce1, 0xe645, 0xbce5,
02636 0xe5fc, 0xbaab, 0xe641, 0xe65a, 0xe642, 0xe640, 0xbcea, 0xe658,

```

02637 0xe5fe, 0xe651, 0xe650, 0xe65d, 0xe647, 0xbcee, 0xe9f3, 0xbf49,
02638 0xbefe, 0xea40, 0xe9eb, 0xbf41, 0xe9f7, 0xbf48, 0xbf43, 0xe9f5,
02639 0xed4f, 0xe9fb, 0xea42, 0xe9fa, 0xe9e9, 0xe9f8, 0xea44, 0xea46,
02640 0xbefd, 0xea45, 0xbf44, 0xbf4a, 0xbf47, 0xe9fe, 0xbf46, 0xe9f9,
02641 0xe9ed, 0xe9f2, 0xe9fd, 0xbf45, 0xbf42, 0xbefc, 0xbf40, 0xe9f1,
02642 0xe5fd, 0xe9ec, 0xe9ef, 0xea41, 0xe9f4, 0xe9ea, 0xed4e, 0xea43,
02643 0xe9ee, 0xe9fc, 0xed51, 0xc0e3, 0xc0d7, 0xc0db, 0xed53, 0xed59,
02644 0xed57, 0xc0d9, 0xc0da, 0xc0e1, 0xed5a, 0xed52, 0xc0dc, 0xed56,
02645 0xed55, 0xed5b, 0xc0e2, 0xc0dd, 0xc0e0, 0xed54, 0xc0e4, 0xc0de,
02646 0xc0e5, 0xc0d8, 0xed58, 0xed50, 0xe9ff, 0xc271, 0xe9ff4, 0xe9ff6,
02647 0xc26f, 0xe9ff2, 0xe9ff3, 0xe9fee, 0xe9f6, 0xe9ef, 0xc270, 0xe9feb,
02648 0xc26d, 0xe9ff8, 0xc26e, 0xe9fec, 0xe9fed, 0xe9ff1, 0xc273, 0xc272,
02649 0xe9ff0, 0xc378, 0xf25f, 0xf265, 0xc379, 0xf25c, 0xc376, 0xc373,
02650 0xf267, 0xc377, 0xc374, 0xf25e, 0xf261, 0xf262, 0xf263, 0xf266,
02651 0xe9ff5, 0xf25d, 0xc375, 0xf264, 0xf268, 0xf260, 0xf45d, 0xc46a,
02652 0xf460, 0xc46b, 0xf468, 0xf45f, 0xf45c, 0xf45e, 0xf462, 0xf465,
02653 0xf464, 0xf467, 0xf45b, 0xc469, 0xf463, 0xf466, 0xf469, 0xf461,
02654 0xf5d3, 0xf5d4, 0xf5d8, 0xf5d9, 0xf5d6, 0xf5d7, 0xf5d5, 0xc4e9,
02655 0xc578, 0xf6eb, 0xf6e8, 0xf6e9, 0xf6ea, 0xc579, 0xf7e5, 0xf7e4,
02656 0xf8af, 0xc5f4, 0xf8ad, 0xf8b0, 0xf8ae, 0xf8f5, 0xc657, 0xc665,
02657 0xf9a3, 0xf96c, 0xf96e, 0xf9a2, 0xf9d0, 0xf9d1, 0xa4f5, 0xa6c7, 0xca41,
02658 0xcb5e, 0xa85f, 0xa862, 0xcb5f, 0xa860, 0xa861, 0xcd58, 0xcd5a,
02659 0xcd55, 0xcd52, 0xcd54, 0xaaa4, 0xaa2, 0xcd56, 0xaaa3, 0xcd53,
02660 0xcd50, 0xaaa1, 0xcd57, 0xcd51, 0xaaa5, 0xcd59, 0xcfaf, 0xcfb3,
02661 0xacb7, 0xcfb6, 0xacaf, 0xacb2, 0xacb4, 0xacb6, 0xacb3, 0xcfb2,
02662 0xcfb1, 0xacb1, 0xcfb4, 0xcfb5, 0xcfae, 0xacb5, 0xacb0, 0xcfb0,
02663 0xd277, 0xd278, 0xd279, 0xaf50, 0xaf4c, 0xd26e, 0xd276, 0xd27b,
02664 0xaf51, 0xd26c, 0xd272, 0xd26b, 0xd275, 0xd271, 0xaf4d, 0xaf4f,
02665 0xd27a, 0xd26a, 0xd26d, 0xd273, 0xd274, 0xd27c, 0xd270, 0xaf4e,
02666 0xb26d, 0xd64e, 0xd650, 0xd64c, 0xd658, 0xd64a, 0xd657, 0xb269,
02667 0xd648, 0xda5b, 0xd652, 0xb26c, 0xd653, 0xd656, 0xd65a, 0xd64f,
02668 0xd654, 0xb26a, 0xb26b, 0xd659, 0xd64d, 0xd649, 0xd65b, 0xd651,
02669 0xd655, 0xd64b, 0xb548, 0xb549, 0xda65, 0xb54f, 0xda59, 0xda62,
02670 0xda58, 0xb54c, 0xda60, 0xda5e, 0xda5f, 0xb54a, 0xda63, 0xda5c,
02671 0xda5a, 0xb54b, 0xda5d, 0xda61, 0xb54d, 0xda64, 0xde70, 0xde77,
02672 0xde79, 0xdea1, 0xb7da, 0xde6b, 0xb7d2, 0xde7a, 0xb7d7, 0xdea2,
02673 0xb7ce, 0xde7d, 0xde6d, 0xde7e, 0xde6c, 0xb7dc, 0xde78, 0xb7cf,
02674 0xdea3, 0xb7d4, 0xde71, 0xb7d9, 0xde7c, 0xde6f, 0xde76, 0xde72,
02675 0xde6e, 0xb7d1, 0xb7d8, 0xb7d6, 0xb7d3, 0xb7db, 0xb7d0, 0xde75,
02676 0xb7d5, 0xb54e, 0xde7b, 0xde73, 0xde74, 0xe2c1, 0xbab4, 0xe2bd,
02677 0xe2c3, 0xe2bf, 0xbab6, 0xe2be, 0xe2c2, 0xe2ba, 0xe2bc, 0xbab5,
02678 0xe2c0, 0xbab7, 0xbab2, 0xe2c4, 0xbab3, 0xe667, 0xe664,
02679 0xe670, 0xe66a, 0xe66c, 0xbcf4, 0xe666, 0xe66e, 0xe66d, 0xe66b,
02680 0xe671, 0xbcf7, 0xe668, 0xe66f, 0xbcf5, 0xe663, 0xe665, 0xbcf6,
02681 0xe662, 0xe672, 0xe669, 0xea4a, 0xbf51, 0xea55, 0xea53, 0xbf4b,
02682 0xea49, 0xea4c, 0xea4d, 0xea48, 0xbf55, 0xbf56, 0xea47, 0xea56,
02683 0xea51, 0xbf4f, 0xbf4c, 0xea50, 0xea4e, 0xbf52, 0xea52, 0xbf4d,
02684 0xbf4e, 0xea4f, 0xbf50, 0xea4b, 0xea54, 0xbf53, 0xea57, 0xea58,
02685 0xbf54, 0xc0e7, 0xc0ee, 0xed5c, 0xed62, 0xed60, 0xc0ea, 0xc0e9,
02686 0xc0e6, 0xed5e, 0xc0ec, 0xc0eb, 0xc0e8, 0xed61, 0xed5d, 0xed5f,
02687 0xc0ed, 0xc277, 0xe9ff, 0xc274, 0xc275, 0xe9ff, 0xc276, 0xe9ff,
02688 0xe9ff9, 0xf26c, 0xe9ff, 0xf26d, 0xc37a, 0xf26b, 0xf26a, 0xf269,
02689 0xc37b, 0xc46c, 0xf46a, 0xf46b, 0xf5dc, 0xf5db, 0xc4ea, 0xf5da,
02690 0xf6ec, 0xf6ed, 0xf7e6, 0xf8b1, 0xf8f6, 0xf9bc, 0xc679, 0xf9c6,
02691 0xa4f6, 0xaaa6, 0xaaa7, 0xacb8, 0xc0ef, 0xa4f7, 0xaaa8, 0xaf52,
02692 0xb7dd, 0xa4f8, 0xb26e, 0xbab8, 0xc962, 0xcfb7, 0xd27d, 0xe2c5,
02693 0xc0f0, 0xa4f9, 0xaaa9, 0xcfb8, 0xcfb9, 0xda66, 0xb550, 0xdea4,
02694 0xb7de, 0xe2c6, 0xbcf8, 0xc37c, 0xa4fa, 0xda67, 0xa4fb, 0xa6c9,
02695 0xca42, 0xa6c8, 0xa865, 0xa864, 0xa863, 0xcb60, 0xaaaa, 0xaaaab,
02696 0xcd5b, 0xcfb8, 0xcfb9, 0xcfb9, 0xcfb9, 0xcfb9, 0xcfb9, 0xcfb9,
02697 0xd2a2, 0xd2a1, 0xd27e, 0xaf53, 0xd65d, 0xd65e, 0xb26f, 0xd65c,
02698 0xd65f, 0xb552, 0xb270, 0xb551, 0xda6b, 0xda6a, 0xda68, 0xda69,
02699 0xda6c, 0xdea6, 0xdea5, 0xdea9, 0xdea8, 0xdea7, 0xbab9, 0xe2c9,
02700 0xe2c8, 0xbaba, 0xe2c7, 0xe673, 0xe674, 0xbcf9, 0xea59, 0xea5a,
02701 0xf272, 0xc37d, 0xf271, 0xf270, 0xf26e, 0xf26f, 0xc4eb, 0xf46c,
02702 0xf6ee, 0xf8f7, 0xa4fc, 0xc9a5, 0xa5c7, 0xc9a6, 0xca43, 0xca44,
02703 0xcb66, 0xcb62, 0xcb61, 0xaaaac, 0xcb65, 0xa867, 0xcb63, 0xa866,
02704 0xcb67, 0xcb64, 0xcd5f, 0xcfbe, 0xcd5d, 0xcd64, 0xaaaad, 0xaab0,
02705 0xcd65, 0xcd61, 0xcd62, 0xcd5c, 0xaaaaf, 0xcd5e, 0xaaaae, 0xcd63,
02706 0xcd60, 0xcfc2, 0xcabd, 0xcabe, 0xcfc5, 0xcfbf, 0xcfc4, 0xcfc0,
02707 0xcabc, 0xcfc3, 0xcfc1, 0xd2a8, 0xd2a5, 0xd2a7, 0xaf58, 0xaf57,
02708 0xaf55, 0xd2a4, 0xd2a9, 0xaf54, 0xaf56, 0xd2a6, 0xd667, 0xd2a3,
02709 0xd2aa, 0xd662, 0xd666, 0xd665, 0xda6e, 0xda79, 0xd668, 0xd663,
02710 0xda6d, 0xb274, 0xb273, 0xd661, 0xd664, 0xb275, 0xb272, 0xb271,
02711 0xd660, 0xd669, 0xda70, 0xda77, 0xb554, 0xda76, 0xda73, 0xb556,
02712 0xda75, 0xda6f, 0xda71, 0xda74, 0xda72, 0xb555, 0xda78, 0xb553,
02713 0xb7df, 0xdead, 0xdead, 0xb7e2, 0xb7e1, 0xdea, 0xdea,
02714 0xe2ca, 0xbabb, 0xb7e0, 0xdeb0, 0xdea, 0xe2cd, 0xe2cb, 0xbcf,
02715 0xbabc, 0xe2cc, 0xe676, 0xbcfb, 0xe675, 0xe67e, 0xe67d, 0xe67b,
02716 0xe67a, 0xe677, 0xe678, 0xe679, 0xe67c, 0xe6a1, 0xea5f, 0xea5c,
02717 0xea5d, 0xbf57, 0xea5b, 0xea61, 0xea60, 0xea5e, 0xed64, 0xed65,
02718 0xc0f1, 0xc0f2, 0xed63, 0xc279, 0xe9ff, 0xc278, 0xc37e, 0xc3a1,
02719 0xc46d, 0xf46e, 0xf46d, 0xf5dd, 0xf6ef, 0xc57a, 0xf7e8, 0xf7e7,
02720 0xf7e9, 0xa5c8, 0xcfc6, 0xaf59, 0xb276, 0xd66a, 0xa5c9, 0xc9a7,
02721 0xa4fd, 0xca45, 0xcb6c, 0xcb6a, 0xcb6b, 0xcb68, 0xa868, 0xcb69,
02722 0xcd6d, 0xaab3, 0xcd6b, 0xcd67, 0xcd6a, 0xcd66, 0xaab5, 0xcd69,
02723 0xaab2, 0xaab1, 0xaab4, 0xcd6c, 0xcd68, 0xacc2, 0xacc5, 0xcfce,

```

02724 0xcfcfd, 0xcfcfc, 0xacbf, 0xcfd5, 0xcfcfb, 0xacc1, 0xd2af, 0xcfd2,
02725 0xcfd0, 0xacc4, 0xcfc8, 0xcfd3, 0xcfcfa, 0xcfd4, 0xcfd1, 0xcfc9,
02726 0xacc0, 0xcfd6, 0xcfc7, 0xacc3, 0xd2b4, 0xd2ab, 0xd2b6, 0xd2ae,
02727 0xd2b9, 0xd2ba, 0xd2ac, 0xd2b8, 0xd2b5, 0xd2b3, 0xd2b7, 0xaf5f,
02728 0xaf5d, 0xd2b1, 0xd2ad, 0xd2b0, 0xd2bb, 0xd2b2, 0xaf5e, 0xcfcf,
02729 0xaf5a, 0xaf5c, 0xd678, 0xd66d, 0xd66b, 0xd66c, 0xd673, 0xd674,
02730 0xd670, 0xb27b, 0xd675, 0xd672, 0xd66f, 0xb279, 0xd66e, 0xb277,
02731 0xb27a, 0xd671, 0xd679, 0xaf5b, 0xb278, 0xd677, 0xd676, 0xb27c,
02732 0xda7e, 0xdaa1, 0xb560, 0xdaa7, 0xdaa9, 0xdaa2, 0xb55a, 0xdaa6,
02733 0xdaa5, 0xb55b, 0xb561, 0xb562, 0xdaa8, 0xb558, 0xda7d, 0xda7b,
02734 0xdaa3, 0xda7a, 0xb55f, 0xda7c, 0xdaa4, 0xdaaa, 0xb559, 0xb55e,
02735 0xb55c, 0xb55d, 0xb557, 0xb7e9, 0xdeb7, 0xb7e8, 0xdeb, 0xdeb1,
02736 0xdeb, 0xdeb2, 0xdeb3, 0xdeb, 0xdeb, 0xdeb8, 0xdeb9, 0xdeb5,
02737 0xdeb4, 0xdeb, 0xb7e5, 0xdeb6, 0xb7ea, 0xb7e4, 0xb7eb, 0xb7ec,
02738 0xb7e7, 0xb7e6, 0xe2ce, 0xbabe, 0xbabd, 0xe2d3, 0xbcf, 0xbabf,
02739 0xbac1, 0xe2d4, 0xb7e3, 0xbac0, 0xe2d0, 0xe2d2, 0xe2cf, 0xe2d1,
02740 0xe6ab, 0xe6aa, 0xe6a7, 0xbd40, 0xea62, 0xbd41, 0xe6a6, 0xbcf,
02741 0xe6a8, 0xe6a5, 0xe6a2, 0xe6a9, 0xe6a3, 0xe6a4, 0xbcf, 0xed69,
02742 0xea66, 0xea65, 0xea67, 0xed66, 0xbf5a, 0xea63, 0xbf58, 0xbf5c,
02743 0xbf5b, 0xea64, 0xea68, 0xbf59, 0xed6d, 0xc0f5, 0xc27a, 0xc0f6,
02744 0xc0f3, 0xc0f4, 0xed6a, 0xed68, 0xed6b, 0xed6e, 0xc0f4, 0xed6c, 0xed67,
02745 0xf042, 0xf045, 0xf275, 0xf040, 0xf46f, 0xf046, 0xc3a2, 0xf044,
02746 0xc27b, 0xf041, 0xf043, 0xf047, 0xf276, 0xf274, 0xc3a3, 0xf273,
02747 0xc46e, 0xc4ed, 0xf6f1, 0xc4ec, 0xf6f3, 0xf6f0, 0xf6f2, 0xc5d0,
02748 0xf8b2, 0xa5ca, 0xcd6e, 0xd2bc, 0xd2bd, 0xb27d, 0xdeb, 0xbf5d,
02749 0xc3a4, 0xc57b, 0xf8b3, 0xa5cb, 0xcd6f, 0xa260, 0xcfd7, 0xcfd8,
02750 0xd2be, 0xd2bf, 0xb27e, 0xb2a1, 0xdaab, 0xdec, 0xdec1, 0xdec0,
02751 0xe2d5, 0xe2d6, 0xe2d7, 0xbac2, 0xe6ad, 0xe6ac, 0xea69, 0xbf5e,
02752 0xbf5f, 0xed72, 0xed6f, 0xed70, 0xed71, 0xf049, 0xf048, 0xc27c,
02753 0xf277, 0xf5de, 0xa5cc, 0xacc6, 0xb2a2, 0xdec3, 0xa5cd, 0xd2c0,
02754 0xb2a3, 0xb563, 0xb564, 0xa5ce, 0xa5cf, 0xca46, 0xa86a, 0xa869,
02755 0xacc7, 0xcfd9, 0xdaac, 0xa5d0, 0xa5d1, 0xa5d2, 0xa5d3, 0xa86b,
02756 0xa86c, 0xc6e, 0xc6e, 0xaab6, 0xcd72, 0xcd70, 0xcd71, 0xcfd,
02757 0xcfdb, 0xaccb, 0xacc9, 0xacca, 0xacc8, 0xaf60, 0xaf64, 0xaf63,
02758 0xd2c1, 0xaf62, 0xaf61, 0xd2c2, 0xb2a6, 0xd67b, 0xd67a, 0xb2a4,
02759 0xb2a5, 0xb565, 0xdaae, 0xdaad, 0xb2a7, 0xb7ed, 0xdec5,
02760 0xb7ee, 0xdec4, 0xe2d8, 0xe6ae, 0xbd42, 0xea6a, 0xed73, 0xc3a6,
02761 0xc3a5, 0xc57c, 0xa5d4, 0xcd73, 0xb2a8, 0xe2d9, 0xbac3, 0xc6f,
02762 0xc70, 0xc70, 0xaab8, 0xaab9, 0xaab7, 0xaccf, 0xacd0, 0xacc,
02763 0xacce, 0xcfdc, 0xcfd, 0xacc, 0xd2c3, 0xaf68, 0xaf69, 0xb2ab,
02764 0xd2c9, 0xaf6e, 0xaf6c, 0xd2ca, 0xd2c5, 0xaf6b, 0xaf6a, 0xaf65,
02765 0xd2c8, 0xd2c7, 0xd2c4, 0xaf6d, 0xd2c6, 0xaf66, 0xaf67, 0xb2ac,
02766 0xd6a1, 0xd6a2, 0xb2ad, 0xd67c, 0xd67e, 0xd6a4, 0xd6a3, 0xd67d,
02767 0xb2a9, 0xb2aa, 0xdab6, 0xb56b, 0xb56a, 0xdab0, 0xb568, 0xdab3,
02768 0xb56c, 0xdab4, 0xb56d, 0xdab1, 0xb567, 0xb569, 0xdab5, 0xdab2,
02769 0xdaaf, 0xded2, 0xdec7, 0xb7f0, 0xb7f3, 0xb7f2, 0xb7f7, 0xb7f6,
02770 0xded3, 0xded1, 0xdec, 0xdec, 0xdec, 0xb7f4, 0xded0, 0xdec,
02771 0xded4, 0xded, 0xb7f5, 0xb7ef, 0xb7f1, 0xdec9, 0xe2db, 0xbac7,
02772 0xe2df, 0xbac6, 0xe2dc, 0xbac5, 0xdec8, 0xdecf, 0xe2de, 0xbac8,
02773 0xe2e0, 0xe2dd, 0xe2da, 0xe6b1, 0xe6b5, 0xe6b7, 0xe6b3, 0xe6b2,
02774 0xe6b0, 0xbd45, 0xbd43, 0xbd48, 0xe6b4, 0xbd46, 0xe6af,
02775 0xbd47, 0xbac4, 0xe6b6, 0xbd44, 0xea6c, 0xea6b, 0xea73, 0xea6d,
02776 0xea72, 0xea6f, 0xbf60, 0xea71, 0xbf61, 0xbf62, 0xea70, 0xea6e,
02777 0xc0f8, 0xd67, 0xc0f7, 0xed77, 0xed75, 0xed76, 0xc0f9, 0xf04d,
02778 0xc2a1, 0xf04e, 0xc27d, 0xf04f, 0xc27e, 0xf04c, 0xf050, 0xf04a,
02779 0xc3a7, 0xf278, 0xc3a8, 0xc46f, 0xf04b, 0xc470, 0xc4ee, 0xf5df,
02780 0xc57e, 0xf6f4, 0xc57d, 0xf7ea, 0xc5f5, 0xc5f6, 0xf9cc, 0xacd1,
02781 0xcfd, 0xb56e, 0xb56f, 0xa5d5, 0xa6ca, 0xca47, 0xcb71, 0xa86d,
02782 0xaaba, 0xacd2, 0xacd3, 0xacd4, 0xd6a6, 0xd2cb, 0xaf6f, 0xb2ae,
02783 0xd6a5, 0xdab8, 0xb571, 0xdab7, 0xb570, 0xded5, 0xbd4a, 0xe6bb,
02784 0xe6b8, 0xe6b9, 0xe6ba, 0xed78, 0xf051, 0xf471, 0xf470, 0xf6f5,
02785 0xa5d6, 0xcd75, 0xaf70, 0xb572, 0xded6, 0xe2e1, 0xbd4b, 0xea74,
02786 0xf052, 0xf472, 0xa5d7, 0xaabb, 0xacd7, 0xcdf, 0xacd8, 0xacd6,
02787 0xacd5, 0xd2cc, 0xaf71, 0xaf72, 0xaf73, 0xb2b0, 0xd6a7, 0xb2af,
02788 0xdab9, 0xb2b1, 0xb573, 0xded7, 0xb7f8, 0xb7f9, 0xbac9, 0xbaca,
02789 0xbd4c, 0xbf64, 0xea75, 0xbf63, 0xed79, 0xc0fa, 0xf053, 0xf473,
02790 0xa5d8, 0xa86e, 0xcd78, 0xcd77, 0xaabc, 0xcd76, 0xaabd, 0xcd79,
02791 0xcfe5, 0xadb, 0xacda, 0xcfe7, 0xcfe6, 0xacdf, 0xacde, 0xacd9,
02792 0xcfe1, 0xcfe2, 0xcfe3, 0xace0, 0xcfe0, 0xacdc, 0xcfe4, 0xacdd,
02793 0xd2cf, 0xd2d3, 0xd2d1, 0xd2d0, 0xd2d4, 0xd2d5, 0xd2d6, 0xd2ce,
02794 0xd2cd, 0xaf75, 0xaf76, 0xd2d7, 0xd2d2, 0xd6b0, 0xd2d8, 0xaf77,
02795 0xaf74, 0xd6aa, 0xd6a9, 0xd6ab, 0xd6ac, 0xd6ae, 0xd6ad, 0xd6b2,
02796 0xb2b5, 0xb2b2, 0xb2b6, 0xd6a8, 0xb2b7, 0xd6b1, 0xb2b4, 0xd6af,
02797 0xb2b3, 0xdabc, 0xdabe, 0xdaba, 0xdabb, 0xdabf, 0xdac1, 0xdac2,
02798 0xdabd, 0xdac0, 0xb574, 0xdedb, 0xdee0, 0xded8, 0xdedc, 0xdee1,
02799 0xdedd, 0xb7fa, 0xb843, 0xb7fd, 0xded9, 0xdeda, 0xbace, 0xb846,
02800 0xb7fe, 0xb844, 0xb7fc, 0xdedf, 0xb845, 0xdede, 0xb841, 0xb7fb,
02801 0xb842, 0xdee2, 0xe2e6, 0xe2e8, 0xb840, 0xe2e3, 0xbacc, 0xe2e9,
02802 0xbacd, 0xe2e7, 0xe2e2, 0xe2e5, 0xe2ea, 0xbacb, 0xe2e4, 0xbd4e,
02803 0xe6bf, 0xe6be, 0xbd51, 0xbd4f, 0xe6bc, 0xbd4d, 0xe6bd, 0xbd50,
02804 0xea7d, 0xeaa1, 0xea7e, 0xea76, 0xea7a, 0xea79, 0xea77, 0xbf66,
02805 0xbf67, 0xbf65, 0xea78, 0xea7b, 0xea7c, 0xbf68, 0xc140, 0xeda3,
02806 0xc0fc, 0xed7b, 0xc0fe, 0xc141, 0xc0fd, 0xeda2, 0xed7c, 0xc0fb,
02807 0xeda1, 0xed7e, 0xed7e, 0xed7d, 0xf055, 0xc2a4, 0xc2a5, 0xc2a2,
02808 0xc2a3, 0xf054, 0xf27b, 0xc3a9, 0xf279, 0xf27a, 0xf474, 0xf477,
02809 0xf475, 0xf476, 0xf5e0, 0xc4ef, 0xf7eb, 0xf8b4, 0xc5f7, 0xf8f8,
02810 0xf8f9, 0xc666, 0xa5d9, 0xace1, 0xdac3, 0xdee3, 0xa5da, 0xa86f,

```


02811 0xaabe, 0xcfe8, 0xcfe9, 0xaf78, 0xdac4, 0xb575, 0xb847, 0xc142,
02812 0xeda4, 0xf27c, 0xf478, 0xa5db, 0xcd1, 0xcd7a, 0xcd7c, 0xcd7e,
02813 0xcd7d, 0xcd7b, 0xaabf, 0xace2, 0xcff2, 0xcfed, 0xcfea, 0xcff1,
02814 0xace4, 0xace5, 0xcff0, 0xcfef, 0xcfee, 0xcfeb, 0xcfec, 0xcff3,
02815 0xace3, 0xaf7c, 0xafa4, 0xafa3, 0xd2e1, 0xd2db, 0xd2d9, 0xafa1,
02816 0xd6b9, 0xaf7a, 0xaf7a, 0xd2de, 0xd2e2, 0xd2e4, 0xd2e0, 0xd2da,
02817 0xd2df, 0xd2dd, 0xaf79, 0xd2e5, 0xafa5, 0xd2e3, 0xaf7d, 0xd2dc,
02818 0xaf7e, 0xaf7b, 0xb2b9, 0xd6ba, 0xd6b3, 0xd6b5, 0xd6b7, 0xd6b8,
02819 0xd6b6, 0xb2ba, 0xd6bb, 0xd6b4, 0xdac8, 0xb576, 0xdad0, 0xdac5,
02820 0xdad1, 0xdac6, 0xdac7, 0xdacf, 0xdace, 0xdacb, 0xb2b8, 0xb577,
02821 0xdac9, 0xdacc, 0xb578, 0xdacd, 0xdaca, 0xdeee, 0xdef2, 0xb84e,
02822 0xe2f0, 0xb851, 0xdef0, 0xdead, 0xdead, 0xdead, 0xdead, 0xdead,
02823 0xb84d, 0xb84c, 0xb848, 0xdead, 0xb84f, 0xb850, 0xdead, 0xdead,
02824 0xdef1, 0xb84a, 0xb84b, 0xdead, 0xdead, 0xdead, 0xbad0, 0xe2f4,
02825 0xdead, 0xe2f6, 0xbad4, 0xe2f7, 0xe2f3, 0xbad1, 0xe2ef, 0xbad3,
02826 0xe2ec, 0xe2f1, 0xe2f5, 0xe2ee, 0xb849, 0xe2eb, 0xbad2, 0xe2ed,
02827 0xb5d4, 0xe6c1, 0xb5d8, 0xb5d6, 0xbacf, 0xe6c8, 0xe6c9, 0xb5d3,
02828 0xe6c7, 0xe6ca, 0xb5d5, 0xb5d2, 0xe6c3, 0xe6c0, 0xe6c5, 0xe6c2,
02829 0xb5d9, 0xe6c4, 0xe6c6, 0xb5d7, 0xbf6a, 0xeaa8, 0xeaa2, 0xeaa6,
02830 0xeaac, 0xeaa9, 0xeaaa, 0xeaa7, 0xeaa4, 0xbf6c, 0xbf69,
02831 0xeaa3, 0xeaa5, 0xbf6b, 0xeaa6, 0xc146, 0xedaa, 0xeda5, 0xc145,
02832 0xc143, 0xedac, 0xc144, 0xeda8, 0xeda9, 0xeda6, 0xedad, 0xf056,
02833 0xc147, 0xeda7, 0xeda6, 0xedab, 0xf05a, 0xf057, 0xc2a6, 0xf05b,
02834 0xf05d, 0xf05c, 0xf058, 0xf059, 0xf2a3, 0xc3aa, 0xf27e, 0xf2a2,
02835 0xf27d, 0xf2a4, 0xf2a1, 0xf47a, 0xf47d, 0xf479, 0xc471, 0xf47b,
02836 0xf47c, 0xf47e, 0xc472, 0xc474, 0xc473, 0xf5e1, 0xf5e3, 0xf5e2,
02837 0xf6f6, 0xf8b5, 0xf8fa, 0xa5dc, 0xc372, 0xaac0, 0xcda3, 0xaac1,
02838 0xaac2, 0xcda2, 0xcff8, 0xcff7, 0xace6, 0xace9, 0xace8, 0xace7,
02839 0xcff4, 0xcff6, 0xcff5, 0xd2e8, 0xafa7, 0xd2ec, 0xd2eb, 0xd2ea,
02840 0xd2e6, 0xafa6, 0xafa9, 0xafa8, 0xd6c2, 0xd6c0, 0xd6bc, 0xb2bb, 0xd6bd,
02841 0xafab, 0xafa9, 0xafa8, 0xd6c2, 0xd6c0, 0xd6bc, 0xb2bb, 0xd6bd,
02842 0xb2bc, 0xd6be, 0xd6bf, 0xd6c1, 0xb2bd, 0xdad5, 0xdad4, 0xdad3,
02843 0xdad2, 0xdef6, 0xb852, 0xdef3, 0xdef5, 0xb853, 0xb854, 0xdef4,
02844 0xe341, 0xe2f9, 0xe2fa, 0xbad7, 0xbad5, 0xbad6, 0xe343, 0xe342,
02845 0xe2fe, 0xe2fd, 0xe2fc, 0xe2fb, 0xe340, 0xe2f8, 0xe6cb, 0xe6d0,
02846 0xe6ce, 0xe6cd, 0xe6cc, 0xe6cf, 0xeaae, 0xbf6d, 0xc148, 0xedb0,
02847 0xc149, 0xedaf, 0xf05f, 0xf05e, 0xc2a7, 0xf2a5, 0xc3ab, 0xf4a1,
02848 0xc5a1, 0xf6f7, 0xf8b7, 0xf8b6, 0xc9a8, 0xace4, 0xaceb, 0xd6c3,
02849 0xb856, 0xa5dd, 0xa872, 0xa871, 0xa870, 0xcda4, 0xaac4, 0xaac3,
02850 0xacee, 0xcffa, 0xcffd, 0xcffb, 0xacec, 0xaced, 0xcffa, 0xcffc,
02851 0xafb5, 0xd2f3, 0xd2f5, 0xd2f4, 0xafb2, 0xd2ef, 0xafb0, 0xafaf,
02852 0xafb3, 0xafb1, 0xafb4, 0xd2f2, 0xd2ed, 0xd2ee, 0xd2f1, 0xd2f0,
02853 0xd6c6, 0xd6c7, 0xd6c5, 0xd6c4, 0xb2be, 0xb57d, 0xdad6, 0xdad8,
02854 0xdada, 0xb57c, 0xb57a, 0xdad7, 0xb57b, 0xdad9, 0xb579, 0xdf41,
02855 0xdef7, 0xdefa, 0xdefe, 0xb85a, 0xdefc, 0xdefb, 0xdef8, 0xdef9,
02856 0xb858, 0xdf40, 0xb857, 0xb85c, 0xb85b, 0xb859, 0xdefd, 0xe349,
02857 0xe348, 0xe344, 0xbad8, 0xe347, 0xe346, 0xbad9, 0xb5d5, 0xe6d2,
02858 0xb5d6, 0xb5d4, 0xb5d5, 0xb5d5, 0xe6d3, 0xe6d4, 0xe6d5, 0xe6d6,
02859 0xe6d7, 0xe6d8, 0xe6d9, 0xb5d6, 0xb5d7, 0xb5d8, 0xb5d9, 0xe6da,
02860 0xc14a, 0xedb4, 0xedb6, 0xedb2, 0xedb1, 0xf060, 0xc2aa, 0xc2a8,
02861 0xc2a9, 0xf2a6, 0xf2a7, 0xc3ad, 0xc3ac, 0xf4a3, 0xf4a4, 0xf4a2,
02862 0xf6f8, 0xf6f9, 0xa5de, 0xca48, 0xa873, 0xcda5, 0xaac6, 0xaac5,
02863 0xcda6, 0xd040, 0xacef, 0xcffe, 0xacf0, 0xafb6, 0xd2f8, 0xd2f6,
02864 0xd2fc, 0xafb7, 0xd2f7, 0xd2fb, 0xd2f9, 0xd2fa, 0xd6c8, 0xd6ca,
02865 0xb2bf, 0xd6c9, 0xb2c0, 0xb5a2, 0xb5a1, 0xb57e, 0xdadb, 0xdf44,
02866 0xb85d, 0xb85e, 0xdf43, 0xdf42, 0xe34a, 0xbad9, 0xbada, 0xe34b,
02867 0xe34c, 0xbad6, 0xeab5, 0xe6d3, 0xe6d5, 0xe6d4, 0xeab4, 0xeab5,
02868 0xeab2, 0xeab6, 0xeab3, 0xbf73, 0xedb7, 0xc14b, 0xedb8, 0xedb9,
02869 0xc2ab, 0xc2ac, 0xc475, 0xc5d1, 0xa5df, 0xd041, 0xd2fd, 0xafb8,
02870 0xb3ba, 0xb3b9, 0xdadd, 0xb5a3, 0xdadc, 0xdf45, 0xbadc, 0xbadc,
02871 0xe34d, 0xbadd, 0xc476, 0xf4a5, 0xa6cb, 0xaac7, 0xcda7, 0xacf2,
02872 0xacf1, 0xd042, 0xd043, 0xd340, 0xd342, 0xafb9, 0xd344, 0xd347,
02873 0xd345, 0xd346, 0xd343, 0xd2fe, 0xafba, 0xd348, 0xd341, 0xd6d3,
02874 0xb2c6, 0xd6dc, 0xb2c3, 0xd6d5, 0xb2c7, 0xb2c1, 0xd6d0, 0xd6dd,
02875 0xd6d1, 0xd6de, 0xb2c5, 0xb2c2, 0xd6d4, 0xd6d7, 0xb2c4, 0xd6d8,
02876 0xb2c8, 0xd6d9, 0xd6cf, 0xd6d6, 0xd6da, 0xd6d2, 0xd6cd, 0xd6cb,
02877 0xd6db, 0xdadf, 0xdae4, 0xdae0, 0xdae6, 0xb5a7, 0xd6cc, 0xdae1,
02878 0xb5a5, 0xdade, 0xb5ac, 0xdae2, 0xb5ab, 0xdae3, 0xb5ad, 0xb5a8,
02879 0xb5ae, 0xb5a9, 0xb5aa, 0xb5a6, 0xdae5, 0xb861, 0xdf50, 0xdf53,
02880 0xdf47, 0xdf4c, 0xdf46, 0xb863, 0xdf4a, 0xdf48, 0xb862, 0xdf4f,
02881 0xdf4e, 0xdf4b, 0xdf4d, 0xdf49, 0xbae1, 0xdf52, 0xb85f, 0xdf51,
02882 0xe35d, 0xbae8, 0xe358, 0xbae7, 0xe34e, 0xe350, 0xbae0, 0xe355,
02883 0xe354, 0xe357, 0xbae5, 0xe352, 0xe351, 0xbae4, 0xbadf, 0xe353,
02884 0xbae2, 0xe359, 0xe35b, 0xe356, 0xe34f, 0xbae3, 0xbad6, 0xbade,
02885 0xe35c, 0xe6d9, 0xbad6, 0xe6db, 0xbad6, 0xbad6, 0xe6de, 0xe6de,
02886 0xbae6, 0xe6dc, 0xe6d8, 0xb860, 0xbad6, 0xbad6, 0xbad6, 0xbad6,
02887 0xbf76, 0xe6dd, 0xe6d7, 0xbad6, 0xe6da, 0xeac0, 0xeabb, 0xeac5,
02888 0xbf74, 0xeabd, 0xbf78, 0xeac3, 0xeaba, 0xeab7, 0xeac6, 0xc151,
02889 0xbf79, 0xeac2, 0xeab8, 0xbf77, 0xeabc, 0xbf7b, 0xeab9, 0xeabe,
02890 0xbf7a, 0xeac1, 0xeac4, 0xedcb, 0xedcc, 0xedcb, 0xedc3, 0xedc1,
02891 0xc14f, 0xedc8, 0xeabf, 0xedbf, 0xedc9, 0xc14e, 0xedbe, 0xedbd,
02892 0xedc7, 0xedc4, 0xedc6, 0xedba, 0xedca, 0xc14c, 0xedc5, 0xedce,
02893 0xedc2, 0xc150, 0xc14d, 0xedc0, 0xedbb, 0xedcd, 0xbf75, 0xf063,
02894 0xf061, 0xf067, 0xc2b0, 0xf065, 0xf064, 0xc2b2, 0xf06a, 0xc2b1,
02895 0xf06b, 0xf068, 0xc2ae, 0xf069, 0xf062, 0xc2af, 0xc2ad, 0xf2ab,
02896 0xf066, 0xf06c, 0xf2a8, 0xc3b2, 0xc3b0, 0xf2aa, 0xf2ac, 0xf2a9,
02897 0xc3b1, 0xc3ae, 0xc3af, 0xc3b3, 0xc478, 0xf4aa, 0xf4a9, 0xf4a7,

```

02898 0xf4a6, 0xf4a8, 0xc477, 0xc479, 0xc4f0, 0xf5e5, 0xf5e4, 0xf6fa,
02899 0xf6fc, 0xf6fe, 0xf6fd, 0xf6fb, 0xc5a3, 0xc5a2, 0xc5d3, 0xc5d2,
02900 0xc5d4, 0xf7ed, 0xf7ec, 0xf8fb, 0xf8b8, 0xf8fc, 0xc658, 0xc659,
02901 0xf96d, 0xc67e, 0xa6cc, 0xcda8, 0xd045, 0xd046, 0xd044, 0xacf3,
02902 0xd047, 0xd048, 0xd049, 0xd349, 0xd34f, 0xd34d, 0xafbb, 0xd34b,
02903 0xd34c, 0xd34e, 0xd34a, 0xb2c9, 0xd6de, 0xb2cb, 0xd6e0, 0xb2ca,
02904 0xd6df, 0xdae8, 0xb5af, 0xdaea, 0xdae7, 0xd6e1, 0xb5b0, 0xdae9,
02905 0xdf56, 0xb864, 0xdf54, 0xb865, 0xdf55, 0xb866, 0xbae9, 0xe361,
02906 0xe35e, 0xe360, 0xbaea, 0xbaeb, 0xe35f, 0xe6df, 0xe6e0, 0xbd6b,
02907 0xe6e2, 0xe6e1, 0xa261, 0xeaca, 0xeacb, 0xeac7, 0xeac8, 0xbf7c,
02908 0xbf7d, 0xeac9, 0xc157, 0xc153, 0xc158, 0xc154, 0xc156, 0xc152,
02909 0xc155, 0xc2b3, 0xedcf, 0xf2ae, 0xf2ad, 0xf4ab, 0xc47a, 0xc47b,
02910 0xf741, 0xf5e6, 0xf740, 0xf8fd, 0xf9a4, 0xa6cd, 0xa874, 0xcda9,
02911 0xaaac, 0xacf6, 0xd04c, 0xacf4, 0xd04a, 0xacf9, 0xacf5, 0xacfa,
02912 0xacf8, 0xd04b, 0xacf7, 0xafbf, 0xafbe, 0xd35a, 0xafc7, 0xd353,
02913 0xd359, 0xafc3, 0xd352, 0xd358, 0xd356, 0xafc2, 0xafc4, 0xd355,
02914 0xafbd, 0xd354, 0xafc8, 0xafc5, 0xafc9, 0xafc6, 0xd351, 0xd350,
02915 0xd357, 0xafc0, 0xafbc, 0xafc1, 0xd6f0, 0xd6e9, 0xb5b5, 0xd6e8,
02916 0xb2cf, 0xb2d6, 0xb2d3, 0xb2d9, 0xb2d8, 0xb2d4, 0xd6e2, 0xd6e5,
02917 0xd6e4, 0xb2d0, 0xd6e6, 0xd6ef, 0xb2d1, 0xd6e3, 0xd6ec, 0xd6ed,
02918 0xb2d2, 0xd6ea, 0xb2d7, 0xb2cd, 0xb2d5, 0xd6e7, 0xb2cc, 0xd6eb,
02919 0xd6ee, 0xdafb, 0xdaf2, 0xb5b2, 0xdaf9, 0xdaf6, 0xdaee, 0xdaf7,
02920 0xb5b4, 0xdaef, 0xdaeb, 0xb86c, 0xdaf4, 0xb5b1, 0xdafa, 0xb5b8,
02921 0xb5ba, 0xdaed, 0xb5b9, 0xdaf0, 0xb5b3, 0xdaf8, 0xdaf1, 0xdaf5,
02922 0xdaf3, 0xb5b6, 0xdaec, 0xb5bb, 0xb2ce, 0xb5b7, 0xb5bc, 0xb868,
02923 0xdf5d, 0xdf5e, 0xdf61, 0xdf65, 0xdf5b, 0xdf59, 0xb86a, 0xdf60,
02924 0xdf64, 0xdf5c, 0xdf58, 0xdf57, 0xdf62, 0xdf5a, 0xdf5e, 0xb86b,
02925 0xb869, 0xdf66, 0xb867, 0xdf63, 0xe372, 0xbaee, 0xe36a, 0xbd78,
02926 0xe374, 0xbaf1, 0xe378, 0xbaf7, 0xe365, 0xe375, 0xe362, 0xe377,
02927 0xe366, 0xbafe, 0xbafb, 0xbafb, 0xe376, 0xe370, 0xbaed, 0xbaf5,
02928 0xbaf3, 0xbaf9, 0xe363, 0xbafa, 0xe371, 0xbaf6, 0xbaec, 0xe373,
02929 0xbaf5, 0xbaf0, 0xbaf8, 0xe368, 0xe367, 0xe364, 0xe36c, 0xe369,
02930 0xe36d, 0xbaf4, 0xe379, 0xbaf2, 0xe36e, 0xe36f, 0xe36b, 0xbafc,
02931 0xe6e7, 0xbd70, 0xbd79, 0xbd75, 0xe6e4, 0xbd72, 0xbd76, 0xe6f0,
02932 0xbd6c, 0xe6e8, 0xbd74, 0xe6eb, 0xe6e6, 0xbd73, 0xbd77, 0xe6e5,
02933 0xbd71, 0xe6ef, 0xbd6e, 0xe6ee, 0xe6ed, 0xbd7a, 0xe572, 0xbd6d,
02934 0xe6ec, 0xe6e3, 0xbd7b, 0xe6ea, 0xbd6f, 0xe6e9, 0xbfa2, 0xbfa7,
02935 0xbf7e, 0xead8, 0xeacf, 0xeadb, 0xead3, 0xead9, 0xbfa8, 0xbfa1,
02936 0xeacc, 0xead2, 0xeadc, 0xead5, 0xeada, 0xeace, 0xead6, 0xbfa3,
02937 0xead4, 0xbfa6, 0xbfa5, 0xead0, 0xead1, 0xeacd, 0xead7, 0xbfa4,
02938 0xeade, 0xeadd, 0xedda, 0xedd6, 0xc15f, 0xedd0, 0xc159, 0xc169,
02939 0xeddc, 0xc161, 0xc15d, 0xedd3, 0xc164, 0xc167, 0xedde, 0xc15c,
02940 0xedd5, 0xc165, 0xede0, 0xeddd, 0xedd1, 0xc160, 0xc15a, 0xc168,
02941 0xedd8, 0xc163, 0xedd2, 0xc15e, 0xeddf, 0xc162, 0xc15b, 0xedd9,
02942 0xc166, 0xedd7, 0xeddb, 0xf074, 0xc2b9, 0xf077, 0xc2b4,
02943 0xc2b5, 0xf06f, 0xf076, 0xf071, 0xc2ba, 0xc2b7, 0xf06d, 0xc2b6,
02944 0xf073, 0xf075, 0xc2b8, 0xf072, 0xf070, 0xf2b8, 0xc3b7, 0xc3b8,
02945 0xc3b4, 0xc3b5, 0xf2b4, 0xf2b2, 0xf2b6, 0xc3ba, 0xf2b7, 0xf2b0,
02946 0xf2af, 0xf2b3, 0xf2b1, 0xc3b6, 0xf2b5, 0xf4ac, 0xc47e, 0xc47d,
02947 0xf4ad, 0xf4af, 0xf4ae, 0xc4a1, 0xf5eb, 0xf5e8, 0xf5e9, 0xf5e7,
02948 0xf5ea, 0xc4f2, 0xf5ec, 0xc4f1, 0xf742, 0xc5d5, 0xc5d7, 0xf7ee,
02949 0xc5d6, 0xf8b9, 0xf940, 0xf942, 0xf8fe, 0xf941, 0xc66c, 0xa6ce,
02950 0xacfb, 0xd26f, 0xafca, 0xb2da, 0xdafc, 0xdafd, 0xeadf, 0xc16a,
02951 0xede1, 0xc2bb, 0xf2ba, 0xf2b9, 0xc4a2, 0xf5ed, 0xf743, 0xc5f8,
02952 0xca49, 0xaac9, 0xa875, 0xd04d, 0xd360, 0xd35b, 0xd35f, 0xd35d,
02953 0xafcb, 0xd35e, 0xd35c, 0xd6f1, 0xdafe, 0xdb40, 0xdf69, 0xdf6a,
02954 0xb86e, 0xb86f, 0xdf68, 0xdf6b, 0xdf67, 0xb86d, 0xbb40, 0xb870,
02955 0xe37a, 0xbd7c, 0xe6f1, 0xbd7d, 0xbfa9, 0xae2, 0xae0, 0xae1,
02956 0xede4, 0xede3, 0xede2, 0xf2bb, 0xc3b9, 0xf2bc, 0xf744, 0xc5f9,
02957 0xf8ba, 0xa6cf, 0xaacb, 0xaaca, 0xd04f, 0xacfc, 0xd04e, 0xd362,
02958 0xafcc, 0xd6f2, 0xd361, 0xb2dc, 0xd6f5, 0xd6f3, 0xd6f4, 0xb2db,
02959 0xdb42, 0xdb43, 0xdb41, 0xb873, 0xdf6d, 0xdf6c, 0xdf6e, 0xb872,
02960 0xb871, 0xc6f2, 0xe6f4, 0xbd7e, 0xe6f3, 0xae3, 0xbfaa, 0xf079,
02961 0xf078, 0xc3bb, 0xf2bd, 0xc3bd, 0xc3bc, 0xf4b0, 0xf5ee, 0xc4f3,
02962 0xa6d0, 0xd050, 0xacfd, 0xd365, 0xafce, 0xd364, 0xd363, 0xafcd,
02963 0xd6fb, 0xd6fd, 0xd6fe, 0xd6f7, 0xb2dd, 0xd6f8, 0xb2de, 0xd6fc,
02964 0xd6f9, 0xd6fa, 0xb2df, 0xb5be, 0xb5bf, 0xdb44, 0xdf6f, 0xdf70,
02965 0xe37e, 0xbb43, 0xbb41, 0xbb42, 0xe37b, 0xe37c, 0xe37d, 0xe6f9,
02966 0xe6fa, 0xbda1, 0xe6f7, 0xe6f6, 0xe6f8, 0xe6f5, 0xbfad, 0xae4,
02967 0xbfab, 0xbfac, 0xede6, 0xc16b, 0xede5, 0xefa8, 0xf07a, 0xf07b,
02968 0xc2bc, 0xc2bd, 0xc16c, 0xf2be, 0xf2bf, 0xf4b1, 0xc4a3, 0xa6d1,
02969 0xa6d2, 0xacfe, 0xaacc, 0xafcf, 0xd051, 0xb5c0, 0xa6d3, 0xad41,
02970 0xd052, 0xd053, 0xad40, 0xad42, 0xa6d4, 0xd054, 0xafd1, 0xd366,
02971 0xafd3, 0xafd0, 0xafd2, 0xd741, 0xb2e0, 0xd740, 0xd6fe, 0xdf71,
02972 0xe3a1, 0xbda2, 0xbfae, 0xae6, 0xae5, 0xede7, 0xf5ef, 0xa6d5,
02973 0xcb73, 0xcdaa, 0xad43, 0xd055, 0xd368, 0xafd4, 0xd367, 0xafd5,
02974 0xd743, 0xb2e2, 0xd742, 0xd744, 0xb2e1, 0xdb46, 0xdb47, 0xdb45,
02975 0xb5c1, 0xb874, 0xb875, 0xbb45, 0xe3a3, 0xe3a2, 0xbb44, 0xe6fb,
02976 0xe6fc, 0xae7, 0xc170, 0xc16f, 0xc16d, 0xc16e, 0xc171, 0xf07c,
02977 0xc2bf, 0xc2be, 0xf2c0, 0xf4b2, 0xc5a5, 0xc5a4, 0xa6d6, 0xd1fb,
02978 0xb877, 0xb5c2, 0xb876, 0xbb46, 0xa6d7, 0xc9a9, 0xa6d8, 0xa6d9,
02979 0xcdab, 0xcb76, 0xcb77, 0xa877, 0xcb74, 0xa876, 0xa879, 0xcb75,
02980 0xa87b, 0xa87a, 0xcb78, 0xa878, 0xaad1, 0xaacf, 0xcdad, 0xaace,
02981 0xaad3, 0xaad5, 0xaad2, 0xcdb0, 0xcdac, 0xaad6, 0xaad0, 0xa87c,
02982 0xaad4, 0xcdaf, 0xcdae, 0xaacd, 0xd05b, 0xad47, 0xad48, 0xd05d,
02983 0xd057, 0xd05a, 0xd063, 0xd061, 0xad49, 0xd067, 0xad4c, 0xd064,
02984 0xd05c, 0xd059, 0xdb49, 0xd062, 0xad44, 0xd065, 0xd056, 0xd05f,

```

02985 0xad46, 0xad4b, 0xd060, 0xad4f, 0xad4d, 0xd058, 0xad4a, 0xd05e,
02986 0xad4e, 0xad45, 0xd066, 0xafda, 0xafe3, 0xafd8, 0xafd6, 0xd36a,
02987 0xafde, 0xafdb, 0xd36c, 0xafdd, 0xd36b, 0xd369, 0xd36e, 0xafe2,
02988 0xafe0, 0xdb48, 0xd36f, 0xd36d, 0xafd7, 0xafd9, 0xafdc, 0xafdf,
02989 0xafe1, 0xd74e, 0xb2e4, 0xd745, 0xd747, 0xd748, 0xd750, 0xd74c,
02990 0xd74a, 0xd74d, 0xd751, 0xb2e5, 0xb2e9, 0xd746, 0xd74f, 0xb2e7,
02991 0xb2e6, 0xd74b, 0xd749, 0xb2e3, 0xb2e8, 0xb5c8, 0xdb51, 0xdb4f,
02992 0xb5ca, 0xdb4a, 0xdfa1, 0xb5c9, 0xdb4e, 0xdb4b, 0xb5c5, 0xb5cb,
02993 0xdb50, 0xb5c7, 0xdb4d, 0xb47, 0xb5c6, 0xdb4c, 0xb5cc, 0xb5c4,
02994 0xb5c3, 0xdf77, 0xdf75, 0xdf7b, 0xdf73, 0xdfa2, 0xdf78, 0xdf72,
02995 0xb87b, 0xb8a3, 0xdf7d, 0xdf76, 0xb87e, 0xb87c, 0xdf7e, 0xb879,
02996 0xb878, 0xdf79, 0xb87d, 0xb5cd, 0xdf7c, 0xdf74, 0xb87a, 0xb8a1,
02997 0xb8a2, 0xbb4c, 0xbb48, 0xbb4d, 0xe3a6, 0xe3a5, 0xe3a7, 0xbb4a,
02998 0xe3a4, 0xbb4b, 0xe3aa, 0xe3a9, 0xe3a8, 0xbb49, 0xe741, 0xe744,
02999 0xbda8, 0xe743, 0xbda7, 0xbda3, 0xbda4, 0xbda5, 0xe740, 0xe6fe,
03000 0xbda6, 0xe742, 0xe6fd, 0xeae9, 0xeaf3, 0xbfb1, 0xbfb0, 0xeaed,
03001 0xeae6, 0xeaea, 0xeae8, 0xeaf1, 0xbfaf, 0xeaf0, 0xeaec,
03002 0xeaf2, 0xeae6, 0xc174, 0xede8, 0xede9, 0xc178, 0xc17a, 0xc177,
03003 0xc176, 0xc175, 0xc173, 0xede9, 0xedec, 0xc172, 0xeded, 0xc179,
03004 0xedeb, 0xede8, 0xc2c0, 0xc2c1, 0xf0a1, 0xf07d, 0xf07e, 0xf2c2,
03005 0xf2c1, 0xc3be, 0xf4b4, 0xc4a4, 0xf4b3, 0xf5f0, 0xf745, 0xc5a6,
03006 0xf943, 0xf944, 0xc5d8, 0xa6da, 0xaad7, 0xdb52, 0xbb4e, 0xc17b,
03007 0xedef, 0xa6db, 0xafe5, 0xafe4, 0xdb53, 0xeaf4, 0xa6dc, 0xad50,
03008 0xdb54, 0xdb55, 0xdb56, 0xbb4f, 0xbfb2, 0xa6dd, 0xaad8, 0xd068,
03009 0xafe6, 0xd370, 0xb2ea, 0xdb57, 0xb8a4, 0xbb50, 0xbfb3, 0xc17c,
03010 0xc2c2, 0xf4b5, 0xa6de, 0xaad9, 0xafe7, 0xd752, 0xb5ce, 0xbb51,
03011 0xe3ab, 0xe745, 0xa6df, 0xb5cf, 0xdfa3, 0xbb52, 0xa6e0, 0xcdb1,
03012 0xd069, 0xad51, 0xd372, 0xafea, 0xafe8, 0xafe9, 0xafeb, 0xd371,
03013 0xd757, 0xd754, 0xd756, 0xb2eb, 0xb2ed, 0xb2ec, 0xd753, 0xb2ee,
03014 0xd755, 0xdb58, 0xdb59, 0xdfa6, 0xdfa7, 0xdfa5, 0xdfa8,
03015 0xb8a5, 0xdfa4, 0xbb53, 0xe74a, 0xe746, 0xe749, 0xe74b, 0xe748,
03016 0xe747, 0xeaf5, 0xeaf6, 0xeaf7, 0xbfb4, 0xbfb5, 0xedf1, 0xedf0,
03017 0xedf2, 0xf0a3, 0xf0a2, 0xf2c4, 0xf2c5, 0xf2c3, 0xc4a5, 0xf4b6,
03018 0xf4b7, 0xf746, 0xf7ef, 0xf8bb, 0xa6e1, 0xa87d, 0xc17d, 0xa6e2,
03019 0xd758, 0xdb5b, 0xc641, 0xca4a, 0xca4b, 0xca4d, 0xa6e3, 0xca4e,
03020 0xca4c, 0xcba2, 0xcba3, 0xcb7b, 0xcba1, 0xa8a1, 0xa8a2, 0xcb7c,
03021 0xcb7a, 0xcb79, 0xcb7d, 0xa87e, 0xcb7e, 0xd06a, 0xcdb6, 0xaadc,
03022 0xcdb5, 0xcdb7, 0xaadb, 0xcdbc, 0xaadf, 0xcdb2, 0xcdbc, 0xcdb6,
03023 0xaae6, 0xcdbc3, 0xaae3, 0xcdb9, 0xcdbf, 0xcdb1, 0xcdb4, 0xaae2,
03024 0xaadd, 0xcdba, 0xaae4, 0xaae7, 0xaae1, 0xaada, 0xcdb8, 0xcdb8,
03025 0xcdb5, 0xaae9, 0xaae5, 0xaae0, 0xcdbd, 0xafec, 0xcdbb, 0xaade,
03026 0xaae8, 0xcdb3, 0xcdb2, 0xcdb4, 0xad62, 0xad5c, 0xad64, 0xad61,
03027 0xd071, 0xd074, 0xad5d, 0xd06b, 0xad56, 0xad60, 0xad63, 0xad65,
03028 0xd0a2, 0xd077, 0xad55, 0xd0a1, 0xad59, 0xad57, 0xad52, 0xd06f,
03029 0xd07e, 0xd073, 0xd076, 0xd0a5, 0xad66, 0xd07d, 0xad5e, 0xd078,
03030 0xd0a4, 0xd075, 0xd079, 0xd07c, 0xd06d, 0xd0a3, 0xd07b, 0xd06c,
03031 0xd070, 0xad5f, 0xad5a, 0xad53, 0xad58, 0xad54, 0xad67, 0xd06e,
03032 0xd3a5, 0xad5b, 0xd07a, 0xce41, 0xd3a8, 0xaffa, 0xd376, 0xd3a3,
03033 0xd37d, 0xd3b2, 0xd3aa, 0xd37e, 0xd3a9, 0xd378, 0xd37c, 0xd3b5,
03034 0xaffd, 0xd3ad, 0xd3a4, 0xafed, 0xd3b3, 0xd374, 0xd3ac, 0xaffc,
03035 0xafff, 0xd3f3, 0xafff, 0xafff, 0xd3ab, 0xafff, 0xafff, 0xafff,
03036 0xd072, 0xdb5c, 0xd3a6, 0xd37a, 0xafff, 0xd37b, 0xd3a1, 0xafff,
03037 0xd375, 0xd3af, 0xd3ae, 0xd3b6, 0xafff, 0xafff, 0xd3b4, 0xd3b0,
03038 0xd3a7, 0xd3a2, 0xafff, 0xafff, 0xd377, 0xafee, 0xd3b1, 0xafee,
03039 0xd379, 0xd75e, 0xd760, 0xd765, 0xd779, 0xb2fc, 0xb2f2, 0xd75d,
03040 0xb2fd, 0xb2fe, 0xd768, 0xd76f, 0xd775, 0xd762, 0xd769, 0xb340,
03041 0xd777, 0xd772, 0xb2fa, 0xb2f8, 0xd76e, 0xd76a, 0xd75c, 0xb2ef,
03042 0xd761, 0xd759, 0xb2f7, 0xb2f9, 0xd766, 0xd763, 0xb2f4, 0xd773,
03043 0xb2f1, 0xd764, 0xd77a, 0xd76c, 0xd76b, 0xb2f0, 0xb2fb, 0xb2f3,
03044 0xd75a, 0xd75e, 0xd770, 0xd776, 0xb341, 0xd75b, 0xd767, 0xd76d,
03045 0xb2f6, 0xd778, 0xd771, 0xd774, 0xb2f5, 0xdb6c, 0xdb60, 0xb5d7,
03046 0xdb7d, 0xdba7, 0xdbaa, 0xb5d5, 0xdb68, 0xdba3, 0xdb69, 0xdb77,
03047 0xb5e2, 0xdb73, 0xb5df, 0xdb74, 0xdb5d, 0xdba4, 0xb5e8, 0xdba1,
03048 0xdb75, 0xdbac, 0xdb70, 0xdfc8, 0xdbaf, 0xb5e6, 0xdb6e, 0xdb7a,
03049 0xb5e9, 0xb5d4, 0xdb72, 0xdbad, 0xdb6b, 0xdb64, 0xdb6f, 0xdb63,
03050 0xdb61, 0xb5d0, 0xdba5, 0xdb6a, 0xdba8, 0xdba9, 0xb5d8, 0xb5dd,
03051 0xb5d9, 0xb5e1, 0xdb7e, 0xb5da, 0xdb76, 0xdb66, 0xb5d2, 0xb5de,
03052 0xdba2, 0xdbab, 0xdb65, 0xb5e0, 0xdbb0, 0xdb71, 0xdb6d, 0xb5d1,
03053 0xb5e5, 0xdb7c, 0xb5e7, 0xdb78, 0xb5dc, 0xb5de, 0xb5d3, 0xb5d3,
03054 0xb5e4, 0xdb79, 0xdb67, 0xdb7b, 0xdb62, 0xdba6, 0xdbae, 0xdb5f,
03055 0xdfc7, 0xdfdd, 0xb855, 0xdfcc, 0xdfca, 0xdfb5, 0xb8a9, 0xdfc5,
03056 0xdfd9, 0xdfc1, 0xb8b1, 0xdfd8, 0xdfbf, 0xb5e3, 0xdfcf, 0xdfc0,
03057 0xdfd6, 0xb8b0, 0xb8a8, 0xdfaa, 0xdfb2, 0xdfcb, 0xdfc3, 0xdfdc,
03058 0xdfc6, 0xb8b6, 0xdfd7, 0xb8ad, 0xdfc9, 0xdfd1, 0xdfb6, 0xdfd0,
03059 0xdfel, 0xdfb1, 0xdfd2, 0xdfdf, 0xdfab, 0xb5db, 0xdfb9, 0xdfb8,
03060 0xb8af, 0xdfbc, 0xdfbe, 0xdfcd, 0xdfde, 0xb8b2, 0xb8b3, 0xdfb0,
03061 0xb8ab, 0xdfb4, 0xdfda, 0xb8b4, 0xb8ac, 0xb8ae, 0xb8b5, 0xdfb0,
03062 0xdfd3, 0xdfce, 0xdfbb, 0xdfba, 0xb8aa, 0xdfac, 0xb8a7, 0xdfc4,
03063 0xdfad, 0xdfc2, 0xdfb7, 0xdfdb, 0xb8a6, 0xdfb3, 0xdfaf, 0xdfd5,
03064 0xdfae, 0xb8b6, 0xe3d3, 0xe3c2, 0xe3ac, 0xe3ca, 0xb8b5, 0xe3bb,
03065 0xe3c5, 0xb8b5, 0xe3be, 0xb8b9, 0xe3af, 0xe3cd, 0xe3ae, 0xe3c1,
03066 0xe3ad, 0xe3bf, 0xe3c8, 0xe3c6, 0xe3ba, 0xe3b5, 0xe3b3, 0xe3b4,
03067 0xe3c7, 0xe3d2, 0xe3bc, 0xb8b5, 0xe3b7, 0xe3cb, 0xb8b5, 0xe3b6,
03068 0xe3b0, 0xe3c0, 0xb8b1, 0xb8b5, 0xb8b5, 0xe3b8, 0xe3b2, 0xb8b7,
03069 0xdfd4, 0xb8b5, 0xe3c3, 0xb8b4, 0xb8b6, 0xb8b5, 0xe3c4, 0xe3b9,
03070 0xe3b1, 0xe3cc, 0xe3bd, 0xb8b6, 0xe3d0, 0xb8b5, 0xe3cf, 0xe3c9,
03071 0xe3ce, 0xe3d1, 0xe773, 0xe774, 0xe767, 0xe766, 0xe762, 0xb8b4,

```
03072 0xbdac, 0xe776, 0xe775, 0xdfa9, 0xe75f, 0xe763, 0xe75d, 0xe770,
03073 0xe761, 0xe777, 0xe75a, 0xe758, 0xe764, 0xe76e, 0xe769, 0xbdb6,
03074 0xe74f, 0xe76d, 0xbdb7, 0xdfbd, 0xe75b, 0xe752, 0xe755, 0xe77b,
03075 0xe75c, 0xe753, 0xe751, 0xe74e, 0xbdb0, 0xe765, 0xbdaf, 0xbdb3,
03076 0xe760, 0xe768, 0xbda9, 0xe778, 0xe77c, 0xbdad, 0xe757, 0xe76b,
03077 0xe76f, 0xe754, 0xe779, 0xbdb2, 0xbdb1, 0xe74c, 0xbdb5, 0xe772,
03078 0xe756, 0xe76a, 0xe750, 0xe75e, 0xe759, 0xbdad, 0xbdae, 0xe76c,
03079 0xe77d, 0xe77a, 0xe771, 0xe74d, 0xbdaa, 0xeb49, 0xeb40, 0xeb43,
03080 0xbfbf, 0xeb45, 0xeaf9, 0xeb41, 0xeb47, 0xbfb8, 0xbfb6, 0xbfb6,
03081 0xeafb, 0xeb4c, 0xeb46, 0xeafc, 0xeb55, 0xeb4f, 0xeaf8, 0xee46,
03082 0xeafe, 0xbfb7, 0xeb4a, 0xeb54, 0xbfbf, 0xeb51, 0xeafd, 0xeb44,
03083 0xeb48, 0xeb42, 0xeb56, 0xeb53, 0xeb50, 0xbfb9, 0xbfb8, 0xbfb6,
03084 0xeafa, 0xeb57, 0xbfbf, 0xeb4d, 0xeb4b, 0xeb4e, 0xee53, 0xee40,
03085 0xee45, 0xee52, 0xee44, 0xedfb, 0xee41, 0xc1a2, 0xedf4, 0xee4d,
03086 0xee4f, 0xedf3, 0xc1a1, 0xee51, 0xee49, 0xc1a8, 0xee50, 0xee42,
03087 0xc1aa, 0xedf9, 0xeb52, 0xee4a, 0xee47, 0xedf5, 0xee55, 0xc1a4,
03088 0xc1a5, 0xedf7, 0xee48, 0xee54, 0xee4b, 0xedfd, 0xc1a7, 0xc1a3,
03089 0xee4c, 0xedf8, 0xee56, 0xedf8, 0xee43, 0xee4e, 0xedfa, 0xedfc,
03090 0xc2cb, 0xedf6, 0xc1a9, 0xc2c4, 0xc17e, 0xc1a6, 0xc2c8, 0xf0b3,
03091 0xf0a9, 0xf0a4, 0xf0aa, 0xf0b4, 0xf0b8, 0xf0b7, 0xc2ca, 0xc2c9,
03092 0xf0ab, 0xf0b9, 0xf0ae, 0xf0a6, 0xf0a8, 0xf0a7, 0xf0ad, 0xf0b2,
03093 0xf0a5, 0xf0ac, 0xf0b1, 0xc2c7, 0xf0af, 0xc2c5, 0xf0b0, 0xc2c3,
03094 0xc2c6, 0xf2d5, 0xf0b5, 0xc3c2, 0xf2cd, 0xf2d1, 0xf2c9, 0xf2cc,
03095 0xf2d4, 0xc3c0, 0xf2d9, 0xf2d2, 0xf2ca, 0xf2d3, 0xc3c3,
03096 0xc3c4, 0xf2d7, 0xf2cb, 0xc3bf, 0xc3c1, 0xf2c6, 0xf2ce, 0xf2c8,
03097 0xf2d8, 0xf2d6, 0xf2c7, 0xf2cf, 0xf4be, 0xc3c5, 0xf2d0, 0xc4a7,
03098 0xc4a9, 0xc4a6, 0xf4c3, 0xf4bb, 0xf4b9, 0xf4bd, 0xf4ba, 0xf4bf,
03099 0xf4c1, 0xc4aa, 0xc4ac, 0xf4c0, 0xc4ad, 0xc4ab, 0xf4c2, 0xc4a8,
03100 0xc4f4, 0xf5f1, 0xf5f7, 0xc4f6, 0xf4bc, 0xf5f6, 0xf5fd, 0xf5f4,
03101 0xf5fb, 0xc5da, 0xf5f8, 0xf5f5, 0xf0b6, 0xf5fe, 0xf5f3, 0xf5f8,
03102 0xf5fc, 0xf5f2, 0xf74a, 0xc4f5, 0xf5f9, 0xf7f4, 0xf74b, 0xf749,
03103 0xf747, 0xf748, 0xf74c, 0xc5d9, 0xf7f2, 0xf7f0, 0xf7f5, 0xf7f3,
03104 0xf7f6, 0xc5da, 0xf7f1, 0xf8bc, 0xf945, 0xf946, 0xf947, 0xf9c7,
03105 0xf9bd, 0xca4f, 0xaaea, 0xad68, 0xd3b8, 0xd3b7, 0xb040, 0xb342,
03106 0xd77c, 0xd77b, 0xb5ea, 0xb8b8, 0xb8b7, 0xb8b9, 0xe3d4, 0xe77e,
03107 0xeb58, 0xeb5a, 0xeb59, 0xc1ab, 0xee57, 0xf0ba, 0xf9a5, 0xa6e4,
03108 0xcdc9, 0xcdca, 0xcdc8, 0xcdc7, 0xaaeb, 0xd0a9, 0xd0a7, 0xd0a6,
03109 0xad69, 0xad6b, 0xad6a, 0xd0a8, 0xd3c4, 0xd3c1, 0xd3bf, 0xb041,
03110 0xd3c2, 0xb046, 0xd3bc, 0xd3cb, 0xd3cd, 0xd3bd, 0xb043, 0xd3ce,
03111 0xd3c9, 0xd3bb, 0xd3c0, 0xd3ca, 0xd3c6, 0xd3c3, 0xb048, 0xd3cc,
03112 0xd3be, 0xd3c7, 0xd3b9, 0xb047, 0xb044, 0xd3c5, 0xd3c8, 0xd3ba,
03113 0xb045, 0xb042, 0xb34c, 0xd7a5, 0xb34b, 0xd7a8, 0xd7ab, 0xb348,
03114 0xb346, 0xd77e, 0xd7a9, 0xd7a7, 0xd7a4, 0xd7ac, 0xd7ad, 0xd7af,
03115 0xd7b0, 0xd77d, 0xb345, 0xd7a2, 0xd7a1, 0xd7ae, 0xb347, 0xd7a3,
03116 0xb349, 0xb344, 0xd7a6, 0xb34d, 0xb34a, 0xd7aa, 0xb5f1, 0xdbbf,
03117 0xdbb4, 0xb5ee, 0xdfe7, 0xdbbd, 0xdbb1, 0xb5ec, 0xdbb6, 0xb5ef,
03118 0xdbba, 0xdbb8, 0xb5f2, 0xb5eb, 0xdbb2, 0xdbb5, 0xb5f0, 0xdbb3,
03119 0xdbbe, 0xdbbc, 0xdbb7, 0xdbb9, 0xdbbb, 0xb5ed, 0xdfe8, 0xdfee,
03120 0xdfe4, 0xdfea, 0xb8ba, 0xdfe6, 0xb8c0, 0xb8bf, 0xb8be, 0xdfed,
03121 0xb8c1, 0xb8c2, 0xdfe3, 0xdff0, 0xb8c3, 0xb8bd, 0xb8bc, 0xdfec,
03122 0xb8c4, 0xdfe2, 0xdfe5, 0xdfef, 0xdfeb, 0xe3f4, 0xe3e9, 0xb8bb,
03123 0xbb6a, 0xe3dd, 0xe3f2, 0xe3de, 0xbb65, 0xe3db, 0xe3e4, 0xe3dc,
03124 0xbb67, 0xe3de, 0xe3f1, 0xbb68, 0xe3ee, 0xe3ef, 0xe3d7, 0xbb6d,
03125 0xe3e6, 0xe3e0, 0xe3e7, 0xe3da, 0xe3f3, 0xe3eb, 0xe3e5, 0xe3d5,
03126 0xbb69, 0xe3ec, 0xbb6c, 0xe3f0, 0xe3ea, 0xbb66, 0xe3e8, 0xe3e2,
03127 0xbb64, 0xe3d9, 0xe3e1, 0xe3ed, 0xe3df, 0xe3e3, 0xbdc1, 0xdfe9,
03128 0xe7b2, 0xe7b3, 0xe7ad, 0xe7ad, 0xe7aa, 0xbdc2, 0xe7a8, 0xbb6b,
03129 0xe7a1, 0xbdc0, 0xe7a7, 0xbdbf, 0xe7ac, 0xe7a9, 0xe7b9, 0xe7b4,
03130 0xe7ae, 0xe7b3, 0xbdbb, 0xe7ab, 0xe7be, 0xe7a2, 0xe7a3, 0xe7ba,
03131 0xbdbc, 0xe7bf, 0xbdbe, 0xe7c0, 0xe7b0, 0xe3d8, 0xe7b6, 0xe7af,
03132 0xe7b8, 0xe7b5, 0xe7a6, 0xbdb9, 0xe7bd, 0xbdba, 0xe7a4, 0xbdbd,
03133 0xeb64, 0xe7b7, 0xe7bc, 0xeb61, 0xbdb8, 0xbfc0, 0xeb6b, 0xeb67,
03134 0xeb65, 0xeb60, 0xeb6f, 0xbfc4, 0xeb5c, 0xeb68, 0xeb69, 0xeb5f,
03135 0xeb5e, 0xeb6c, 0xeb62, 0xeb5d, 0xeb63, 0xeb6e, 0xeb5b, 0xeb6d,
03136 0xeb6a, 0xbfc2, 0xbfc1, 0xbfc3, 0xeb66, 0xf0cb, 0xee59, 0xc1b1,
03137 0xee5d, 0xee5a, 0xee61, 0xee67, 0xee5c, 0xee70, 0xc1ae, 0xee6a,
03138 0xee5f, 0xee6b, 0xee66, 0xee6d, 0xee5e, 0xc1b3, 0xc1b2, 0xee60,
03139 0xee6e, 0xee58, 0xee6c, 0xc1ac, 0xee64, 0xee63, 0xee68, 0xee5b,
03140 0xc1b0, 0xc1b4, 0xee62, 0xee69, 0xc1b5, 0xee65, 0xc1ad, 0xc1af,
03141 0xf0c7, 0xf0c5, 0xf0cc, 0xf0c9, 0xf0cd, 0xf0be, 0xf0c6, 0xf0d1,
03142 0xee6f, 0xf0c2, 0xc2cf, 0xe7a5, 0xf0bd, 0xf0ca, 0xf0c4, 0xf0c1,
03143 0xf0bc, 0xf0bb, 0xf0d0, 0xf0c0, 0xf0bf, 0xc2cd, 0xf0c8, 0xc2cc,
03144 0xc2ce, 0xf0c3, 0xf0cf, 0xf2de, 0xf2df, 0xc3c9, 0xf2dc, 0xc3c6,
03145 0xf2e4, 0xc3ca, 0xf2e6, 0xf2db, 0xf0ce, 0xf2e8, 0xf2dd, 0xc3c7,
03146 0xf2e3, 0xf2e5, 0xf2e0, 0xf2e7, 0xf2e2, 0xf2e1, 0xc3c8, 0xf4c5,
03147 0xf4c6, 0xf4c8, 0xc4ae, 0xc4af, 0xf4c9, 0xf4c7, 0xf4c4, 0xf642,
03148 0xf645, 0xf641, 0xc4fa, 0xf643, 0xc4f9, 0xc4f8, 0xc4f7, 0xf644,
03149 0xf751, 0xf74f, 0xf74e, 0xf640, 0xf750, 0xf646, 0xf74d, 0xf7f9,
03150 0xf7d7, 0xf7f7, 0xc5db, 0xf7f8, 0xf7fa, 0xf8bf, 0xc5fa, 0xf8be,
03151 0xf8bd, 0xc5fb, 0xc65a, 0xf96e, 0xf9a7, 0xf9a6, 0xf9a8, 0xa6e5,
03152 0xd0aa, 0xd3cf, 0xd3d0, 0xbdc0, 0xf647, 0xf8c0, 0xa6e6, 0xad6c,
03153 0xd0ab, 0xd7b1, 0xb34e, 0xbdc2, 0xbdc1, 0xb5f3, 0xb8c5, 0xe7c1,
03154 0xbdc3, 0xbdc4, 0xbfc5, 0xc5fc, 0xa6e7, 0xd0ac, 0xaaed, 0xd0ae,
03155 0xd0ad, 0xad6d, 0xd3d1, 0xd3d8, 0xb049, 0xd3d6, 0xd3d4, 0xd3db,
03156 0xd3d2, 0xd3d3, 0xb04a, 0xb04e, 0xd3dc, 0xb04d, 0xd3da, 0xd3d7,
03157 0xd3d5, 0xb04b, 0xb04c, 0xd3d9, 0xb350, 0xd7b2, 0xb355, 0xd7c2,
03158 0xb354, 0xd7c4, 0xd7b8, 0xb352, 0xd7c3, 0xd7b3, 0xb353, 0xd7bf,
```

03159 0xd7bb, 0xd7bd, 0xd7b7, 0xd7be, 0xb34f, 0xd7ba, 0xd7b9, 0xd7b5,
03160 0xd7c0, 0xd7bc, 0xd7b4, 0xd7b6, 0xb351, 0xd7c1, 0xb5f6, 0xdbcd,
03161 0xdbcb, 0xdbcb, 0xdbcb, 0xdbcb, 0xdbcb, 0xdbcb, 0xdbcb, 0xdbcb,
03162 0xdbcb, 0xb5f4, 0xb5f5, 0xdbcf, 0xb8cd, 0xdf2, 0xdf8, 0xdf3,
03163 0xdf4, 0xdf9, 0xb8cf, 0xb8c7, 0xb8ce, 0xdf1, 0dbcb, 0xb8ca,
03164 0xb8c8, 0xb8cf, 0xdf6, 0xb8c9, 0xb8cb, 0xdf5, 0xb8c6, 0xb8cc,
03165 0xe3f6, 0xbb74, 0xe442, 0xe441, 0xe3fb, 0xbb76, 0xe440, 0xe3f7,
03166 0xe3f8, 0xbb6e, 0xbb70, 0xe3fd, 0xe3f5, 0xbb72, 0xbb71, 0xe3f9,
03167 0xe3fe, 0xe3fc, 0xbb73, 0xe3fa, 0xdbce, 0xbb6f, 0xe7c2, 0xe7c9,
03168 0xbdc6, 0xe7cd, 0xbdc, 0xe7c5, 0xe7c3, 0xe7cc, 0xbdc5, 0xe7cb,
03169 0xbdc7, 0xbdc8, 0xe7c4, 0xbdc9, 0xe7ca, 0xe7c6, 0xe7c7, 0xe7c8,
03170 0xbb75, 0xeb70, 0xeb7c, 0xbfc, 0xeb77, 0xeb79, 0xbfc8, 0xeb71,
03171 0xeb75, 0xeb78, 0xbfc6, 0xbfc9, 0xeb7b, 0xeb73, 0xeb74, 0xeb7a,
03172 0xeb72, 0xeb76, 0xbfc7, 0xee72, 0xee71, 0xc1b7, 0xee77, 0xc1b9,
03173 0xc1b6, 0xee73, 0xc1ba, 0xee74, 0xee75, 0xee78, 0xc1b8, 0xf0d6,
03174 0xf0d9, 0xf0d3, 0xf0d5, 0xf0d4, 0xf0d7, 0xf0d8, 0xee76, 0xf0d2,
03175 0xc3cd, 0xf2ec, 0xf2ef, 0xf2f1, 0xf2ea, 0xf2eb, 0xf2ee, 0xf2f0,
03176 0xc3ce, 0xc3cc, 0xc3cb, 0xf2ed, 0xf2e9, 0xf4ca, 0xc4b0, 0xf4cb,
03177 0xf649, 0xc4fb, 0xf64b, 0xc4fc, 0xf648, 0xf64a, 0xc5a8, 0xf752,
03178 0xc5a7, 0xf7fd, 0xf7fc, 0xf7fb, 0xf948, 0xf949, 0xf94b, 0xf94a,
03179 0xca50, 0xa6e8, 0xad6e, 0xd7c5, 0xb5f7, 0xdfa, 0xc2d0, 0xf2f2,
03180 0xa8a3, 0xb357, 0xb356, 0xdbd0, 0xb5f8, 0xdbd2, 0xdbd1, 0xdfb,
03181 0xb8d0, 0xe443, 0xe446, 0xe445, 0xe444, 0xe7ce, 0xe7d0, 0xe7cf,
03182 0xbfcc, 0xbfc, 0xc1bb, 0xee79, 0xee7a, 0xc2d1, 0xf2f4,
03183 0xf2f3, 0xf4cc, 0xc4b1, 0xc4fd, 0xf754, 0xf753, 0xc65b, 0xa8a4,
03184 0xd0af, 0xad6f, 0xd7c8, 0xd7c6, 0xd7c7, 0xdbd4, 0xdbd5, 0xe043,
03185 0xdbd3, 0xdfc, 0xe041, 0xe040, 0xe042, 0xb8d1, 0xdfc, 0xdfd,
03186 0xe044, 0xe449, 0xe447, 0xe448, 0xe7d3, 0xe7d1, 0xe7d2, 0xeb7d,
03187 0xee7c, 0xee7d, 0xc2d2, 0xf2f5, 0xf4cd, 0xc4b2, 0xf64c, 0xf755,
03188 0xc5a9, 0xf7fe, 0xf94c, 0xa8a5, 0xad71, 0xad72, 0xd0b0, 0xd0b1,
03189 0xad70, 0xb054, 0xb052, 0xb051, 0xb058, 0xb050, 0xb059, 0xd3dd,
03190 0xb056, 0xb053, 0xb057, 0xb055, 0xb04f, 0xb35f, 0xb359, 0xd7cc,
03191 0xb35e, 0xb360, 0xb35a, 0xb35b, 0xd7ca, 0xb358, 0xd7cb, 0xb35d,
03192 0xd7c9, 0xb35c, 0xb644, 0xb646, 0xdbd8, 0xb645, 0xb5f9, 0xb5fd,
03193 0xb8e4, 0xe049, 0xbda, 0xb5fe, 0xbdd, 0xbde, 0xb643, 0xbde0,
03194 0xbde2, 0xbde3, 0xbdd7, 0xbdd6, 0xbde4, 0xb642, 0xbde1, 0xbddf,
03195 0xb640, 0xb5fb, 0xb647, 0xbddb, 0xbdbc, 0xbdb9, 0xb641, 0xb5fc,
03196 0xb5fa, 0xe048, 0xb8df, 0xb8da, 0xb8d5, 0xb8e5, 0xb8d6, 0xb8d2,
03197 0xb8e1, 0xb8de, 0xb8e0, 0xb8d7, 0xb8dc, 0xb8d3, 0xb8d4, 0xe050,
03198 0xe04d, 0xe045, 0xe04a, 0xb8e2, 0xe051, 0xb8e3, 0xb8d9, 0xe047,
03199 0xe04f, 0xe04b, 0xe04e, 0xe04c, 0xb8dd, 0xe046, 0xb8dd, 0xe44c,
03200 0xb78, 0xb7b, 0xe44e, 0xbba5, 0xe44d, 0xb7d, 0xbdcf, 0xe44f,
03201 0xbba4, 0xe44b, 0xbba6, 0xb79, 0xb8db, 0xb7c, 0xb7a, 0xb7e,
03202 0xbba2, 0xb77, 0xbba7, 0xbba3, 0xbba1, 0xe44a, 0xbdd6, 0xbdd2,
03203 0xbdd9, 0xb7d6, 0xbdda, 0xe7e2, 0xe7db, 0xbdc, 0xe7e3, 0xe7dd,
03204 0xbdd5, 0xe7de, 0xbdd4, 0xe7e1, 0xbdc, 0xe7df, 0xe7d5, 0xbdc,
03205 0xebaa, 0xbdd3, 0xbdd0, 0xbdd8, 0xe7d4, 0xe7d8, 0xbdcc, 0xe7d7,
03206 0xe7d9, 0xe7da, 0xbdd7, 0xe7dc, 0xe7e0, 0xe7e4, 0xbdd, 0xbfd2,
03207 0xeba5, 0xebab, 0xeba8, 0xeb7e, 0xebac, 0xeba1, 0xeba7, 0xbfc,
03208 0xbfd3, 0xebad, 0xbfcf, 0xbfd9, 0xbfd4, 0xebaf, 0xeba9, 0xbfd0,
03209 0xeba2, 0xbfd, 0xeba3, 0xbfd, 0xbfd8, 0xbdd1, 0xbfce,
03210 0xebb0, 0xbfdc, 0xbfd5, 0xebae, 0xbfd1, 0xbfd6, 0xbfd7, 0xc1c3,
03211 0xeea4, 0xeead, 0xeeaa, 0xeeac, 0xc1c0, 0xeea5, 0xeeab, 0xc1bc,
03212 0xeea7, 0xc1c4, 0xeea3, 0xeea8, 0xeeaf, 0xeba6, 0xeea9, 0xeea2,
03213 0xc1bd, 0xeea1, 0xc1be, 0xeeb0, 0xc1bf, 0xeeae, 0xc1c2, 0xee7e,
03214 0xc1c1, 0xeea6, 0xf0dc, 0xf0ea, 0xf0e5, 0xf0e7, 0xf0db, 0xc2d3,
03215 0xf0da, 0xc2d6, 0xf0e9, 0xf0e1, 0xf0de, 0xf0e4, 0xf0dd,
03216 0xf0df, 0xf0e8, 0xf0e6, 0xc2d4, 0xf0ed, 0xf0eb, 0xf0e2, 0xf0ec,
03217 0xf0e3, 0xf2f9, 0xc3cf, 0xf341, 0xf64f, 0xc3d6, 0xf0e0, 0xf2f7,
03218 0xc3d2, 0xf2f8, 0xf2fd, 0xc3d4, 0xc3d5, 0xf2f6, 0xf340, 0xf342,
03219 0xf2fa, 0xf2fc, 0xf2fe, 0xf2fb, 0xf343, 0xc3d1, 0xc3d7, 0xc3d3,
03220 0xc3d0, 0xf4d0, 0xc4b7, 0xf4ce, 0xf4d2, 0xf4d3, 0xc4b5, 0xf4d4,
03221 0xf4d1, 0xf4cf, 0xc4b8, 0xc4b4, 0xf4d5, 0xc4b6, 0xc4b3, 0xc4fe,
03222 0xc540, 0xf64e, 0xf64d, 0xf650, 0xf651, 0xc541, 0xf756, 0xf75b,
03223 0xc5aa, 0xf758, 0xf757, 0xf75a, 0xf759, 0xf843, 0xc5dc, 0xf842,
03224 0xf840, 0xf841, 0xc5fe, 0xc5fd, 0xf8c1, 0xf8c2, 0xc640, 0xf94d,
03225 0xf94e, 0xc667, 0xc66d, 0xf9a9, 0xf9c8, 0xa8a6, 0xd7cd, 0xd7ce,
03226 0xe052, 0xe450, 0xe7e5, 0xc1c6, 0xc1c5, 0xf0ee, 0xf344, 0xf844,
03227 0xa8a7, 0xd3de, 0xb05a, 0xb361, 0xe054, 0xe053, 0xbddc, 0xe7e6,
03228 0xbddd, 0xeeb1, 0xc2d7, 0xc676, 0xa8a8, 0xcdb, 0xd3df, 0xb362,
03229 0xd7cf, 0xd7d0, 0xbde5, 0xb648, 0xb8e6, 0xe056, 0xe055, 0xe057,
03230 0xe451, 0xe452, 0xbba8, 0xbfd, 0xbdde, 0xbfd, 0xeeb5, 0xeeb2,
03231 0xeeb4, 0xeeb3, 0xc1c7, 0xf0ef, 0xf346, 0xf345, 0xcba4, 0xb05c,
03232 0xb05b, 0xd3e0, 0xd7d1, 0xbde7, 0xbde6, 0xb649, 0xe059, 0xe05a,
03233 0xe058, 0xb8e8, 0xb8e7, 0xbbaa, 0xbba9, 0xe7e7, 0xeeb3, 0xeeb1,
03234 0xeeb2, 0xbfd, 0xeeb7, 0xeeb6, 0xf0f2, 0xf0f1, 0xf0f0, 0xf347,
03235 0xf9aa, 0xa8a9, 0xad73, 0xad74, 0xb05d, 0xb05e, 0xd3e2, 0xd3e1,
03236 0xd7d2, 0xb368, 0xb366, 0xb363, 0xb367, 0xb365, 0xb364, 0xb64a,
03237 0xbde, 0xb8ed, 0xb64c, 0xb651, 0xbdec, 0xb653, 0xb652, 0xb655,
03238 0xbdeb, 0xbde8, 0xb64f, 0xb64b, 0xbde9, 0xb654, 0xb650,
03239 0xb64e, 0xb8ef, 0xb8ee, 0xb8ec, 0xb8f0, 0xb8ea, 0xb8eb, 0xb8e9,
03240 0xe05b, 0xe454, 0xbba, 0xbbad, 0xbba, 0xe453, 0xe455, 0xe7ea,
03241 0xe7ec, 0xbde7, 0xbde7, 0xbde0, 0xe7e9, 0xbddf, 0xbde9, 0xbde5,
03242 0xbde6, 0xbde2, 0xe7e8, 0xbde1, 0xe7ee, 0xe7eb, 0xbde8, 0xbde3,
03243 0xbde4, 0xeeb5, 0xeeb7, 0xeeb6, 0xeeb8, 0xbfe0, 0xeeb4, 0xc1cb,
03244 0xeeb8, 0xc1c8, 0xc1cc, 0xc1ca, 0xc1c9, 0xf0f3, 0xf0f6, 0xf0f5,
03245 0xf0f4, 0xc2d8, 0xf348, 0xf349, 0xc3d8, 0xf34a, 0xc3d9, 0xc4ba,

```

03246 0xc4b9, 0xf652, 0xc542, 0xf653, 0xf75c, 0xc5ab, 0xc5ac, 0xf845,
03247 0xc642, 0xa8aa, 0xb36a, 0xb369, 0xe05c, 0xe05d, 0xbbae, 0xebb9,
03248 0xbdea, 0xbdea, 0xebba, 0xebb9, 0xa8ab, 0xd0b2, 0xad76, 0xad75, 0xd3e3,
03249 0xb05f, 0xd3e4, 0xd7d5, 0xd7d4, 0xd7d3, 0xdbee, 0xb658, 0xdbed,
03250 0xb657, 0xdbef, 0xb656, 0xe05f, 0xe062, 0xe060, 0xe061, 0xe065,
03251 0xe05e, 0xe066, 0xe063, 0xe064, 0xbbb0, 0xe456, 0xbbae, 0xe7f2,
03252 0xe7f0, 0xbdeb, 0xe7ef, 0xe7f1, 0xbdec, 0xebbb, 0xebbc, 0xc1cd,
03253 0xf34c, 0xf34e, 0xf34b, 0xf34d, 0xf4d6, 0xf654, 0xf96f, 0xa8ac,
03254 0xad77, 0xd3e5, 0xd3e7, 0xd3e6, 0xd7d8, 0xb36c, 0xd7d6, 0xb36b,
03255 0xd7d9, 0xd7da, 0xd7d7, 0xdbfb, 0xb660, 0xdbf3, 0xdbf9, 0xb65b,
03256 0xb65e, 0xdbf2, 0xb659, 0xdbf6, 0xe06c, 0xb65d, 0xdbf1, 0xdbf7,
03257 0xdbf4, 0xdbfa, 0xdbf0, 0xdbf8, 0xb65c, 0xb65f, 0xdbf5, 0xb65a,
03258 0xb8f2, 0xe068, 0xb8f1, 0xe06f, 0xe06e, 0xb8f8, 0xb8f9, 0xe070,
03259 0xb8f3, 0xe06d, 0xb8f7, 0xe072, 0xe069, 0xe06b, 0xb8f4, 0xe067,
03260 0xe06a, 0xe071, 0xb8f5, 0xe073, 0xb8f6, 0xbbb1, 0xe45b, 0xe461,
03261 0xe459, 0xe462, 0xe458, 0xe45d, 0xe463, 0xe460, 0xe45f, 0xe45e,
03262 0xe457, 0xe45c, 0xe45a, 0xbdf1, 0xbdee, 0xe7fb, 0xe841, 0xe843,
03263 0xe840, 0xe7f8, 0xe7fa, 0xe845, 0xe842, 0xe7fc, 0xe846, 0xe7f9,
03264 0xe844, 0xbdef, 0xbdf5, 0xbdf3, 0xe7f3, 0xbdf4, 0xbdf0, 0xe7f4,
03265 0xe7f6, 0xe7f5, 0xe7fd, 0xe7fe, 0xbdf2, 0xbded, 0xe7f7, 0xebc6,
03266 0xbfe2, 0xebbd, 0xbfe3, 0xbfe6, 0xebc2, 0xebbf, 0xbfe5, 0xebc3,
03267 0xebc4, 0xebbe, 0xebc7, 0xebc0, 0xebc5, 0xbfe4, 0xbfe1, 0xebc1,
03268 0xebbf, 0xc1d0, 0xc1ce, 0xc1d1, 0xc1cf, 0xeebe, 0xebb, 0eeba,
03269 0xebbd, 0xebbc, 0xf145, 0xc2de, 0xf0fb, 0xf0fa, 0xc2d9, 0xf141,
03270 0xf140, 0xf0f7, 0xf143, 0xf0fc, 0xc2dd, 0xf0f9, 0xf142, 0xf0f8,
03271 0xc2da, 0xc2dc, 0xf0fd, 0xc2db, 0xf0fe, 0xf144, 0xf352, 0xc3de,
03272 0xf34f, 0xf353, 0xc3db, 0xf351, 0xc3e0, 0xc3dd, 0xf350, 0xc3df,
03273 0xf354, 0xc3da, 0xc4bc, 0xc4be, 0xf4d9, 0xc4bd, 0xf4d7, 0xc3dc,
03274 0xf4d8, 0xc4bb, 0xc543, 0xc545, 0xf656, 0xc544, 0xf655, 0xf761,
03275 0xc5ad, 0xf760, 0xc5ae, 0xf75e, 0xf75d, 0xf762, 0xf763, 0xf846,
03276 0xf75f, 0xf8c6, 0xf8c3, 0xf8c4, 0xf8c5, 0xc65c, 0xf951, 0xf950,
03277 0xf94f, 0xf970, 0xf9be, 0xf9ab, 0xc66e, 0xa8ad, 0xb060, 0xb8fa,
03278 0xbdf6, 0xebc8, 0xc2df, 0xf355, 0xf9ac, 0xa8ae, 0xaaee, 0xad79,
03279 0xad78, 0xb063, 0xd3e8, 0xb061, 0xd3e9, 0xb062, 0xd7df, 0xd7db,
03280 0xb36d, 0xd7de, 0xd7dd, 0xd7dc, 0xb36e, 0xd7e0, 0xd7e1, 0xdc43,
03281 0xdc41, 0xdc45, 0xdc46, 0xdc4c, 0xdc48, 0xdc4a, 0xdc42, 0xdbfc,
03282 0xdc49, 0xdc4b, 0xdc44, 0xdc47, 0xdbfd, 0xb662, 0xdc40, 0xdbfe,
03283 0xb661, 0xb663, 0xb8fd, 0xe075, 0xe077, 0xe076, 0xe07b, 0xb8fb,
03284 0xe078, 0xe07a, 0xe079, 0xb8fc, 0xb8fe, 0xe07c, 0xe467,
03285 0xe466, 0xe464, 0xe465, 0xbbb3, 0xbbb5, 0xbbb2, 0xbbb4, 0xe84d,
03286 0xe84e, 0xe849, 0xe84a, 0xbdf8, 0xbdfd, 0xbdf7, 0xbdfc, 0xbdf9,
03287 0xe84b, 0xe84c, 0xe848, 0xbef0, 0xbdfb, 0xbdfa, 0xbdfc, 0xe847,
03288 0xebca, 0xbfe8, 0xebcc, 0xbfea, 0xebcf, 0xebcb, 0xebc9, 0xebce,
03289 0xbfe9, 0xebcd, 0xbfe7, 0xc1d3, 0xc1d6, 0xeec1, 0xc1d4, 0xeec0,
03290 0xc1d2, 0xc1d5, 0xf146, 0xf147, 0xf148, 0xc2e0, 0xf149, 0xc2e1,
03291 0xc3e2, 0xf358, 0xf359, 0xf357, 0xf356, 0xf35a, 0xc3e1, 0xf4dd,
03292 0xf4db, 0xf4dc, 0xf4de, 0xf4da, 0xf4df, 0xf658, 0xf659, 0xf657,
03293 0xc546, 0xf764, 0xc5af, 0xf765, 0xf848, 0xf847, 0xa8af, 0xb664,
03294 0xb940, 0xbbb6, 0xbfec, 0xbfeb, 0xc3e3, 0xc47c, 0xc547, 0xa8b0,
03295 0xb064, 0xb941, 0xf35b, 0xcba6, 0xa8b1, 0xa8b4, 0xa8b3, 0xa8b2,
03296 0xcba5, 0xcddc, 0xcddf, 0xaaef, 0xaaef, 0xcddc, 0xcddc, 0xaaef,
03297 0xcdd1, 0xcdd0, 0xcdd2, 0xd0b6, 0xd0b4, 0xad7c, 0xd0b3, 0xada3,
03298 0xad7e, 0xad7b, 0xada4, 0xad7d, 0xada2, 0xada1, 0xd0b5, 0xad7a,
03299 0xb06a, 0xd3eb, 0xd3f1, 0xb067, 0xb06e, 0xb069, 0xd3ee, 0xd3f0,
03300 0xb06c, 0xd3ea, 0xd3ed, 0xb068, 0xb065, 0xd3ec, 0xb06b, 0xd3ef,
03301 0xb06d, 0xb066, 0xd7e3, 0xd7e6, 0xb370, 0xb37a, 0xb37e, 0xd7e4,
03302 0xb37c, 0xb377, 0xb37c, 0xb372, 0xb36f, 0xb371, 0xb37d, 0xd7e5,
03303 0xb375, 0xb378, 0xb374, 0xb379, 0xd7e7, 0xb37b, 0xb373, 0xd7e2,
03304 0xdc4d, 0xb665, 0xdc4f, 0xb667, 0xb669, 0xdc4e, 0xb666, 0xb66a,
03305 0xb668, 0xb947, 0xe0a3, 0xb94f, 0xe07e, 0xb950, 0xb945, 0xe0a1,
03306 0xb94a, 0xe0a2, 0xb943, 0xb942, 0xb94d, 0xb94c, 0xb94b, 0xb949,
03307 0xb94e, 0xe07d, 0xb944, 0xb946, 0xb948, 0xbbb8, 0xbbbb, 0xbbbf,
03308 0xbbb9, 0xbbbe, 0xbbbc, 0xbbb7, 0xbbbd, 0xbbbb, 0xe852, 0xbe43,
03309 0xbe41, 0xe853, 0xbe44, 0xbe42, 0xe851, 0xe850, 0xbff0, 0xe84f,
03310 0xbfee, 0xbfed, 0xebd0, 0xbe45, 0xbfef, 0xebd1, 0xbff2, 0xebd2,
03311 0xbff1, 0xc1d8, 0xeec3, 0xc1d7, 0xc1dc, 0xc1da, 0xc1db, 0xc2e3,
03312 0xc1d9, 0xeec2, 0xebd3, 0xc2e2, 0xc2e4, 0xc3e4, 0xc3e5, 0xf4e0,
03313 0xc5de, 0xc5dd, 0xa8b6, 0xca55, 0xb06f, 0xca52, 0xca53, 0xca51,
03314 0xca54, 0xcbaa, 0xcba7, 0xcba8, 0xa8b7, 0xa8ba, 0xcba9, 0xcba9,
03315 0xa8b9, 0xcba8, 0xa8b8, 0xcdd5, 0xcdd7, 0xaaef, 0xcdd3, 0xcdd6,
03316 0xcdd4, 0xaaef, 0xaaef, 0xd0b8, 0xd0bc, 0xd0b9, 0xada7,
03317 0xada8, 0xd0bb, 0xd0bd, 0xd0bf, 0xada5, 0xd0be, 0xada6, 0xd7ee,
03318 0xd0ba, 0xd3f2, 0xd3fb, 0xd3f9, 0xd3f4, 0xd3f5, 0xd3fd, 0xd3fc,
03319 0xb071, 0xd3f7, 0xd3f2, 0xd3f3, 0xb070, 0xb072, 0xd3f6, 0xd3f8,
03320 0xb3a1, 0xd7f1, 0xd7e9, 0xd7ef, 0xd7f0, 0xb3a2, 0xd7e8, 0xd7ea,
03321 0xd0b7, 0xd7ec, 0xd7ed, 0xd7eb, 0xb66c, 0xdc56, 0xebd4, 0xdc57,
03322 0xdc54, 0xb3a3, 0xb66e, 0xdc53, 0xdc59, 0xdc58, 0xb66b, 0xdc5c,
03323 0xdc52, 0xdc5b, 0xdc50, 0xdc5a, 0xdc55, 0xb66d, 0xe0aa, 0xe0a5,
03324 0xe0ab, 0xe0ae, 0xe0a4, 0xe0a7, 0xb951, 0xe0a9, 0xe0aa, 0xb952,
03325 0xbbc1, 0xbbc0, 0xe46e, 0xe471, 0xe469, 0xe46d, 0xbbc2, 0xe46c,
03326 0xe46a, 0xe470, 0xe46b, 0xe468, 0xe46f, 0xe859, 0xbe48, 0xf14a,
03327 0xe856, 0xe857, 0xe855, 0xdc51, 0xbe47, 0xe85a, 0xe854, 0xbe46,
03328 0xbe49, 0xe858, 0xebd5, 0xbff3, 0xebd6, 0xebd7, 0xeec4, 0xc1dd,
03329 0xf14b, 0xf14c, 0xf14d, 0xf35d, 0xf35c, 0xf4e2, 0xf4e1, 0xf65b,
03330 0xf65c, 0xf65a, 0xf766, 0xc5b0, 0xa8bb, 0xadaa, 0xada9, 0xb075,
03331 0xb074, 0xd440, 0xd441, 0xd3fe, 0xb073, 0xd7f5, 0xd7f6, 0xd7f2,
03332 0xb3a4, 0xd7f3, 0xd7f4, 0xdc5f, 0xdc61, 0xdc5d, 0xdc60, 0xb66f,

```

03333 0xdc5e, 0xb670, 0xdd73, 0xb955, 0xb954, 0xb953, 0xe0ac, 0xe0ad,
03334 0xe473, 0xe475, 0xbbc6, 0xbbc3, 0xbbc5, 0xbbc4, 0xe474, 0xe472,
03335 0xe861, 0xe85e, 0xe85f, 0xe4d, 0xe860, 0xe85b, 0xe85c, 0xbe4a,
03336 0xbe4b, 0xe85d, 0xbe4c, 0xebdb, 0xebdc, 0xebd9, 0xebda, 0xbff4,
03337 0xebd8, 0xeec8, 0xeec5, 0xeec7, 0xc1e0, 0xeecb, 0xc1df, 0xeec9,
03338 0xeec6, 0xeeca, 0xeec6, 0xc1de, 0xf14f, 0xf150, 0xf14e, 0xf152,
03339 0xc2e5, 0xc2e6, 0xf35f, 0xc3e7, 0xf151, 0xf35e, 0xc3e6, 0xf4e5,
03340 0xf4e6, 0xc4bf, 0xf4e4, 0xf4e3, 0xf65d, 0xc548, 0xf849, 0xf8c8,
03341 0xf8c7, 0xc643, 0xc65d, 0xf8c9, 0xf971, 0xc66f, 0xa8bc, 0xaaf6,
03342 0xb956, 0xc4c0, 0xa8bd, 0xadab, 0xb3a5, 0xb671, 0xc2e7, 0xaaf7,
03343 0xd0c1, 0xd0c0, 0xd442, 0xb078, 0xb076, 0xb07a, 0xd444, 0xb079,
03344 0xb077, 0xd443, 0xb3a8, 0xd7fc, 0xb3a7, 0xb3a9, 0xd842, 0xb3ab,
03345 0xd7fe, 0xd840, 0xd7f7, 0xb3aa, 0xd843, 0xd7f9, 0xd7fa, 0xd7f8,
03346 0xb3a6, 0xd841, 0xd7fb, 0xd7fd, 0xdc6d, 0xdc6c, 0xdc6a, 0xdc62,
03347 0xdc71, 0xdc65, 0xdc6f, 0xdc76, 0xdc6e, 0xb679, 0xb675, 0xdc63,
03348 0xdc69, 0xb677, 0xdc68, 0xb678, 0xb67a, 0xdc6b, 0xb672, 0xb673,
03349 0xdc77, 0xdc75, 0xdc74, 0xdc66, 0xdc72, 0xb676, 0xb674, 0xdc73,
03350 0xdc64, 0xdc67, 0xdc70, 0xe4ba, 0xe0b7, 0xe0b0, 0xe0c3, 0xe0cc,
03351 0xe0b3, 0xb961, 0xe0c0, 0xb957, 0xb959, 0xb965, 0xe0b1, 0xb95a,
03352 0xb95c, 0xb966, 0xb95b, 0xb964, 0xe0b9, 0xe0ae, 0xb962, 0xe0b8,
03353 0xb95e, 0xe0ca, 0xb963, 0xe0c8, 0xe0bc, 0xe0c6, 0xb960, 0xe0af,
03354 0xe0c9, 0xe0c4, 0xe0cb, 0xb958, 0xb967, 0xb95d, 0xe0b5, 0xe0bd,
03355 0xe0c1, 0xe0c5, 0xb95f, 0xe0b4, 0xe0b2, 0xe0be, 0xe0bb, 0xe0ba,
03356 0xe0bf, 0xe0c2, 0xe0c7, 0xe478, 0xbbc7, 0xe4a4, 0xe47a, 0xbbcc,
03357 0xbbd0, 0xe4ad, 0xe4b5, 0xe4a6, 0xbbc8, 0xe4aa, 0xe0b6, 0xbbc9,
03358 0xe4b1, 0xe4b6, 0xe4ae, 0xe4b0, 0xe4b9, 0xe4b2, 0xe47e, 0xe4a9,
03359 0xbbd1, 0xbbcd, 0xe47c, 0xe4ab, 0xbbc, 0xe4a5, 0xbba, 0xe4b3,
03360 0xe4a2, 0xe479, 0xbbc, 0xe4b8, 0xe47b, 0xe4af, 0xe4ac, 0xe4a7,
03361 0xe477, 0xe476, 0xe4a1, 0xe4b4, 0xbbcf, 0xe4b7, 0xe47d, 0xe4a3,
03362 0xbe52, 0xbe5a, 0xbe55, 0xe8a4, 0xe8a1, 0xe867, 0xbe50, 0xbe4f,
03363 0xbe56, 0xe865, 0xbe54, 0xe871, 0xe863, 0xe864, 0xbe4e, 0xe8a3,
03364 0xbe58, 0xe874, 0xe879, 0xe873, 0xee, 0xe86f, 0xe877, 0xe875,
03365 0xe868, 0xe862, 0xe87d, 0xbe57, 0xe87e, 0xe878, 0xe86d, 0xe86b,
03366 0xe866, 0xe86e, 0xe87b, 0xe86a, 0xe87a, 0xe8a2, 0xbe53, 0xe876,
03367 0xe87c, 0xe872, 0xe86c, 0xbe51, 0xe4a8, 0xe870, 0xbe59, 0xe869,
03368 0xebf4, 0xbff4, 0xebf3, 0xebf0, 0xec44, 0xbff, 0xec41, 0xebf8,
03369 0xec43, 0xeb, 0xebf6, 0xbffd, 0xeb, 0xebdf, 0xec42, 0xec40,
03370 0xebfe, 0xebed, 0xebec, 0xeb, 0xc040, 0xeb, 0xebf2, 0xebfd,
03371 0xc043, 0xec45, 0xc1e8, 0xc045, 0xbffe, 0xeb, 0xebf, 0xebde,
03372 0xeb, 0xbff5, 0xc042, 0xbffa, 0xeb, 0xebf7, 0xebf1, 0xc041,
03373 0xebdd, 0xc1e3, 0xebf9, 0xebfc, 0xbffc, 0xeb, 0xc044, 0xbff9,
03374 0xbff8, 0xbff5, 0xebfb, 0xbff6, 0xeb, 0xebf, 0xeb, 0xeba,
03375 0xeed2, 0xeed7, 0xc1e5, 0xc1e7, 0xeedd, 0xc1e1, 0xeeec, 0xee3,
03376 0xeed8, 0xeed9, 0xee2, 0xc1ee, 0xee1, 0xeed1, 0xee0, 0xeed4,
03377 0xeeed, 0xc1ed, 0xc1eb, 0xeed5, 0xee8, 0xeeda, 0xee7, 0xee9,
03378 0xeed0, 0xc1e6, 0xeea, 0xeede, 0xc1ea, 0xeedb, 0xc1ec, 0xee4,
03379 0xc1e4, 0xeed6, 0xee5, 0xeedf, 0xeb, 0xee6, 0xeed3, 0xc1e9,
03380 0xeeb, 0xc1e2, 0xeece, 0xf160, 0xf159, 0xc2e9, 0xf154, 0xf163,
03381 0xf15b, 0xeedc, 0xf165, 0xf155, 0xc2e8, 0xf15f, 0xc2ea, 0xc2f2,
03382 0xc2f0, 0xf161, 0xc2f1, 0xf157, 0xf158, 0xf15d, 0xf162, 0xeecd,
03383 0xc2eb, 0xf16a, 0xf167, 0xf16b, 0xf15e, 0xf15a, 0xf168, 0xf36a,
03384 0xf15c, 0xc2ee, 0xc2ed, 0xeecf, 0xc2ef, 0xf164, 0xf166, 0xc2ec,
03385 0xf169, 0xf153, 0xf156, 0xf373, 0xf363, 0xc3eb, 0xf371, 0xf361,
03386 0xc3ec, 0xf36c, 0xf36f, 0xc3f1, 0xf372, 0xf362, 0xf365, 0xc3e9,
03387 0xf374, 0xf36d, 0xf370, 0xc3ef, 0xc3f4, 0xc3f2, 0xf369, 0xf364,
03388 0xc3ed, 0xc3ee, 0xf360, 0xc3ea, 0xc3e8, 0xc3f0, 0xf36f, 0xc3f3,
03389 0xf36b, 0xf375, 0xc3f5, 0xf367, 0xf36e, 0xf4f3, 0xf542, 0xf4f5,
03390 0xf4fc, 0xf366, 0xf4fa, 0xf4e9, 0xf540, 0xc4c3, 0xf4ed, 0xf4fe,
03391 0xf4f4, 0xc4c2, 0xf544, 0xf4f6, 0xf4fb, 0xf4fd, 0xf4e7, 0xf541,
03392 0xf4f2, 0xf4f7, 0xf4eb, 0xf4ef, 0xf543, 0xf4f9, 0xf4e8, 0xf4ec,
03393 0xf4ee, 0xf4f8, 0xc4c1, 0xf4f1, 0xf4ea, 0xf4f0, 0xf661, 0xf666,
03394 0xc54f, 0xf668, 0xc549, 0xf664, 0xf66a, 0xc54e, 0xc54a, 0xc54b,
03395 0xf660, 0xf667, 0xc54d, 0xf665, 0xc54c, 0xf65f, 0xf663, 0xf662,
03396 0xf65e, 0xf669, 0xc5b1, 0xf76d, 0xf770, 0xf76c, 0xf76e, 0xf76f,
03397 0xf769, 0xf76a, 0xf767, 0xf76b, 0xf768, 0xc5b2, 0xc5b3, 0xf84b,
03398 0xf84d, 0xf84c, 0xf84e, 0xc5e0, 0xf84a, 0xc5df, 0xc5e1, 0xf8cb,
03399 0xf8cc, 0xc644, 0xf8ca, 0xf953, 0xf952, 0xf954, 0xc65f, 0xf955,
03400 0xc65e, 0xf956, 0xf972, 0xf975, 0xf974, 0xc668, 0xf973, 0xc672,
03401 0xc670, 0xc671, 0xc677, 0xf9c0, 0xf9c1, 0xf9bf, 0xf9c9, 0xaaf8,
03402 0xd844, 0xdc78, 0xe8a5, 0xf376, 0xaaf9, 0xadac, 0xb07b, 0xd845,
03403 0xd846, 0xb3ac, 0xb67d, 0xdc7a, 0xdc79, 0xb6a3, 0xb67c, 0xdc7b,
03404 0xb67e, 0xb6a2, 0xb6a1, 0xb67b, 0xb698, 0xe0d0, 0xe0ce, 0xe0cf,
03405 0xe0cd, 0xbbd2, 0xbbd5, 0xbbd7, 0xbbd6, 0xbbd3, 0xbbd4, 0xe8a7,
03406 0xe8a6, 0xbe5b, 0xe8a8, 0xe8a9, 0xbe5c, 0xec4d, 0xec4b, 0xee3,
03407 0xec49, 0xec4a, 0xc046, 0xec46, 0xec4e, 0xec48, 0xec4c, 0xeeef,
03408 0xeeef, 0xeeef, 0xc1f3, 0xeeee, 0xc1f2, 0xeeef, 0xc1ef, 0xc1f0,
03409 0xc1f1, 0xec47, 0xc2f5, 0xf16e, 0xf16c, 0xf16d, 0xc2f3, 0xc2f6,
03410 0xc2f4, 0xf377, 0xf378, 0xc3f6, 0xf545, 0xf547, 0xf546, 0xc4c4,
03411 0xc550, 0xf66d, 0xf66c, 0xf66b, 0xaafa, 0xc9aa, 0xca58, 0xa6e9,
03412 0xca56, 0xca59, 0xca57, 0xcbae, 0xa8c1, 0xa8c2, 0xcbb0, 0xa8bf,
03413 0xcba, 0xcba, 0xa8c0, 0xa8be, 0xcdd8, 0xcddb, 0xaaf, 0xcdda,
03414 0xcdd9, 0xaaf, 0xaaf, 0xab40, 0xcddc, 0xaaf, 0xd0c6, 0xadae,
03415 0xadaf, 0xad, 0xd0c7, 0xd0c3, 0xadad, 0xd0c4, 0xd0c5, 0xd0c2,
03416 0xb0a4, 0xb0a1, 0xd445, 0xb0a2, 0xb0a5, 0xd446, 0xb07e, 0xb07c,
03417 0xb07d, 0xb0a3, 0xb3ad, 0xd849, 0xb3b5, 0xd848, 0xd84b, 0xb3b1,
03418 0xd84a, 0xb6ab, 0xb3af, 0xb3b2, 0xb3ae, 0xb3b3, 0xb3b4, 0xb3b0,
03419 0xd847, 0xb6a7, 0xdc7d, 0xdca3, 0xdca2, 0xb6ac, 0xb6a8, 0xb6a9,

```

03420 0xdc7c, 0xdc7e, 0xdca1, 0xb6a4, 0xb6a6, 0xb6aa, 0xb6a5, 0xe0d3,
03421 0xe0d1, 0xe0d2, 0xb96a, 0xb96b, 0xe0d4, 0xb969, 0xb9bd, 0xb9bd,
03422 0xb9bd9, 0xe4bb, 0xe4bc, 0xe8ab, 0xe8aa, 0xc047, 0xc048, 0xec4f,
03423 0xc049, 0xeef6, 0xeef4, 0xeef5, 0xc1f4, 0xf16f, 0xc3f7, 0xc1f5,
03424 0xab41, 0xb0a6, 0xd447, 0xd84c, 0xb3b6, 0xb6ad, 0xdca4, 0xdca6,
03425 0xb6af, 0xb6ae, 0xb6b0, 0xb6b1, 0xdca5, 0xb96e, 0xb96f, 0xb96d,
03426 0xb9db, 0xb96c, 0xe0d5, 0xb9dc, 0xe8ac, 0xec50, 0xc04a, 0xc1f6,
03427 0xf170, 0xf174, 0xc2f9, 0xf171, 0xc2fa, 0xc2f8, 0xf175, 0xc2fb,
03428 0xf173, 0xf379, 0xc2f7, 0xc3f8, 0xf8cd, 0xab42, 0xb3b8, 0xb3b7,
03429 0xb6b2, 0xdca8, 0xdca7, 0xb6b3, 0xe0d9, 0xb973, 0xb970, 0xe0d8,
03430 0xb972, 0xe0d6, 0xb971, 0xe0d7, 0xe4bd, 0xb9bd, 0xe8af, 0xbe5d,
03431 0xe8ad, 0xbe5e, 0xbe5f, 0xe8ae, 0xbe60, 0xec51, 0xc04e, 0xc04b,
03432 0xc050, 0xec53, 0xc04c, 0xec52, 0xc04f, 0xc04d, 0xeef9, 0xeefb,
03433 0xc1f7, 0xeef6, 0xc1f8, 0xeef8, 0xeef7, 0xf177, 0xf176, 0xc2fc,
03434 0xf178, 0xf37e, 0xc3fa, 0xf37d, 0xf37a, 0xc3f9, 0xf37b, 0xf37c,
03435 0xf548, 0xf549, 0xc4c5, 0xc553, 0xf66e, 0xc551, 0xc552, 0xf66f,
03436 0xc5b4, 0xc5b5, 0xf771, 0xc645, 0xf8cf, 0xc647, 0xf8ce, 0xf8d0,
03437 0xc646, 0xc647, 0xf957, 0xf9ad, 0xab43, 0xb974, 0xe4be, 0xe8b0, 0xc051,
03438 0xc052, 0xab44, 0xbe61, 0xc3fb, 0xadb1, 0xc053, 0xc5e2, 0xadb2,
03439 0xd84d, 0xdca9, 0xdcab, 0xdcaa, 0xe0dd, 0xe0da, 0xb975, 0xb976,
03440 0xe0db, 0xe0dc, 0xe4c0, 0xe4c5, 0xb9de, 0xe4bf, 0xe4c1, 0xe4c8,
03441 0xe4c3, 0xe4c7, 0xe4c4, 0xe4c2, 0xe4c6, 0xb9df, 0xe8b3, 0xe8b1,
03442 0xbe63, 0xbe62, 0xe8b2, 0xbe64, 0xec56, 0xec55, 0xc054, 0xec54,
03443 0xeefc, 0xeefc, 0xf41, 0xf41, 0xf41, 0xf41, 0xf41, 0xf41, 0xf41,
03444 0xf17d, 0xf1a2, 0xc2fe, 0xf17b, 0xf17e, 0xf17c, 0xf179, 0xc340,
03445 0xf17a, 0xf3a1, 0xf3a3, 0xf3a2, 0xf54a, 0xf54b, 0xf670, 0xc5b7,
03446 0xc5b6, 0xf8d2, 0xf8d4, 0xf850, 0xc648, 0xf8d1, 0xc669, 0xadbb, 0xb6b4,
03447 0xe4ca, 0xe4c9, 0xe8b5, 0xe8b4, 0xc1fa, 0xf43, 0xf42, 0xf1a5,
03448 0xf1a3, 0xf1a6, 0xf1a4, 0xc3fc, 0xf3a4, 0xf3a5, 0xf3a6, 0xf671,
03449 0xf772, 0xf8d2, 0xadbb, 0xec57, 0xf44, 0xadbb, 0xbbe0, 0xec58,
03450 0xc341, 0xf1a7, 0xc3fd, 0xf54c, 0xf54d, 0xc554, 0xf851, 0xadbb,
03451 0xb3bb, 0xb3bc, 0xd84e, 0xb6b5, 0xb6b6, 0xdcac, 0xb6b7, 0xb97a,
03452 0xb97c, 0xe0df, 0xe0e0, 0xe0de, 0xb977, 0xb978, 0xb97b, 0xb979,
03453 0xe4cb, 0xbbe1, 0xbbe2, 0xe8bc, 0xbe67, 0xe8b7, 0xe8b6, 0xe8bb,
03454 0xbe65, 0xc05b, 0xe8b8, 0xe8bd, 0xe8ba, 0xe8b9, 0xbe66, 0xc059,
03455 0xec5a, 0xc055, 0xec5b, 0xec59, 0xc058, 0xc056, 0xc05a, 0xc057,
03456 0xf45, 0xf4a, 0xf46, 0xf49, 0xc1fb, 0xedd4, 0xf48, 0xf47,
03457 0xc344, 0xc342, 0xc345, 0xc343, 0xf1a8, 0xf1a9, 0xf1aa, 0xc346,
03458 0xf3aa, 0xc440, 0xf3a8, 0xc441, 0xf3a7, 0xf3a9, 0xc3fe, 0xf551,
03459 0xf54e, 0xf54f, 0xf550, 0xf672, 0xc556, 0xc555, 0xf774, 0xf773,
03460 0xc5b8, 0xc5e3, 0xc649, 0xc660, 0xf958, 0xf9ae, 0xf9af, 0xadbb,
03461 0xdcad, 0xe0e1, 0xe4cc, 0xe4cd, 0xbbe3, 0xbbe4, 0xe8be, 0xbe68,
03462 0xc1fc, 0xf1ab, 0xc347, 0xf3ad, 0xc442, 0xf3ac, 0xf3ae, 0xf3ab,
03463 0xf675, 0xf552, 0xf553, 0xc4c6, 0xf674, 0xf673, 0xf775, 0xf9b0,
03464 0xadbb, 0xadbb, 0xb0a7, 0xd448, 0xd44f, 0xb6b8, 0xb6bb, 0xb6b9,
03465 0xdcae, 0xb6bd, 0xb6ba, 0xb6bc, 0xb97e, 0xe0e2, 0xe0e3, 0xe8c0,
03466 0xb97d, 0xb9a1, 0xb9a2, 0xe4cf, 0xe4ce, 0xbbe5, 0xbbe6, 0xe4d0,
03467 0xe8bf, 0xbbe8, 0xbe69, 0xbbe7, 0xc05c, 0xe8c1, 0xe8b6, 0xbe6a,
03468 0xe8c2, 0xe8c5, 0xe8c3, 0xe8c4, 0xbe6c, 0xc061, 0xc05f, 0xc05e,
03469 0xec5d, 0xc060, 0xec5c, 0xf4b, 0xec5e, 0xc05d, 0xec5f, 0xf4e,
03470 0xf4c, 0xf4d, 0xf52, 0xc34b, 0xf51, 0xf54, 0xf53, 0xf50,
03471 0xf4f, 0xc1fd, 0xf1ae, 0xf1ad, 0xc34a, 0xc348, 0xc349, 0xf1ac,
03472 0xf3b1, 0xc443, 0xf3b0, 0xf3af, 0xc444, 0xf558, 0xf557, 0xf555,
03473 0xf554, 0xc4c8, 0xc4c7, 0xf559, 0xf776, 0xc5b9, 0xf677, 0xc557,
03474 0xf676, 0xf556, 0xf777, 0xc5e4, 0xc661, 0xf959, 0xf9b1, 0xadba,
03475 0xd850, 0xf55, 0xadbb, 0xe4d2, 0xe4d1, 0xec60, 0xf557, 0xf556,
03476 0xc34c, 0xf3b2, 0xc4c9, 0xf3b3, 0xc4c9, 0xf9b2, 0xb0a8, 0xb6bf, 0xb6be,
03477 0xe0e4, 0xe0e6, 0xb9a4, 0xe0e5, 0xb9a3, 0xb9a5, 0xe0e7, 0xe4d4,
03478 0xe4d6, 0xe4d5, 0xe4d8, 0xbbe9, 0xe4d7, 0xe4d3, 0xe4d9, 0xe8cc,
03479 0xe8cf, 0xe8d1, 0xe8c7, 0xe8cb, 0xe8c8, 0xbe6e, 0xbe71, 0xbe73,
03480 0xe8c9, 0xe8ca, 0xbe72, 0xe8cd, 0xe8d0, 0xe8ce, 0xbe74, 0xbe70,
03481 0xe8c6, 0xbe6d, 0xbe6f, 0xc063, 0xec66, 0xec64, 0xec63, 0xec69,
03482 0xec68, 0xec6d, 0xec62, 0xc062, 0xec61, 0xec65, 0xc064, 0xf55a,
03483 0xf55e, 0xf55b, 0xf55d, 0xf55c, 0xf559, 0xf55f, 0xf62, 0xf60,
03484 0xf61, 0xc240, 0xc1fe, 0xf58, 0xf63, 0xf1b3, 0xf1b6, 0xf1b8,
03485 0xf1b7, 0xf1b1, 0xf1b5, 0xf1b0, 0xf1b2, 0xc34d, 0xf1af, 0xf1b4,
03486 0xf3c0, 0xf3b5, 0xc445, 0xc446, 0xf3b4, 0xf3b9, 0xf3bf, 0xf3b7,
03487 0xf3be, 0xf3bb, 0xf3ba, 0xf3bd, 0xf3b8, 0xf3b6, 0xf3bc, 0xf560,
03488 0xf55e, 0xc4ca, 0xf55d, 0xf563, 0xf561, 0xc4cb, 0xf55c, 0xf55a,
03489 0xf55b, 0xc4cd, 0xf55f, 0xc4cc, 0xf562, 0xf678, 0xf67e, 0xf679,
03490 0xc55b, 0xf6a1, 0xc55a, 0xf67d, 0xf67c, 0xc559, 0xf67b, 0xc558,
03491 0xf67a, 0xf77d, 0xf7a1, 0xf77e, 0xf77b, 0xc5bb, 0xf778, 0xf77c,
03492 0xf7a3, 0xf7a2, 0xf779, 0xf77a, 0xc5ba, 0xf852, 0xc5e7, 0xf853,
03493 0xc5e5, 0xc5e6, 0xf8d3, 0xc64a, 0xf976, 0xc66a, 0xf9b3, 0xc66b,
03494 0xf9b4, 0xf9b5, 0xf9c3, 0xf9c2, 0xc67a, 0xf9cd, 0xb0a9, 0xe0e9,
03495 0xe0e8, 0xbbea, 0xbbeb, 0xe4da, 0xe8d2, 0xec6c, 0xbe75, 0xc065,
03496 0xec6a, 0xec6b, 0xc066, 0xf64, 0xec6b, 0xf1b9, 0xc34e, 0xf3c1,
03497 0xf566, 0xf564, 0xf565, 0xf6a2, 0xc55c, 0xf7a4, 0xc5ea, 0xc5bc,
03498 0xc5e8, 0xc5e9, 0xf8d4, 0xc662, 0xb0aa, 0xf1ba, 0xd449, 0xb9a6,
03499 0xe4db, 0xbbec, 0xe4dc, 0xe8d4, 0xe8d3, 0xc068, 0xbe76, 0xbe77,
03500 0xe8d7, 0xe8d6, 0xe8d5, 0xec6e, 0xec71, 0xec70, 0xec6f, 0xc067,
03501 0xf68, 0xf66, 0xf65, 0xf67, 0xc34f, 0xf1bc, 0xf1bd, 0xc350,
03502 0xf1bb, 0xf3c3, 0xf3c2, 0xf3c5, 0xc447, 0xf3c4, 0xf567, 0xf569,
03503 0xf568, 0xf6a3, 0xf6a6, 0xf6a4, 0xf6a5, 0xf7a5, 0xc5bd, 0xf854,
03504 0xf855, 0xf856, 0xc64b, 0xc663, 0xf9b6, 0xb0ab, 0xbe78, 0xc069,
03505 0xf1be, 0xf7a6, 0xf9c4, 0xd44a, 0xc67b, 0xb0ac, 0xec72, 0xf1bf,
03506 0xf3c6, 0xf6a7, 0xf7a7, 0xb0ad, 0xe4dd, 0xe4de, 0xbbed, 0xbbee,

```


03507 0xe8d9, 0xbe7a, 0xbe79, 0xe8d8, 0xef69, 0xf1c0, 0xf1c2, 0xf1c1,
03508 0xc353, 0xc352, 0xc351, 0xc55e, 0xf6a8, 0xc55d, 0xf7a9, 0xf7a8,
03509 0xc64c, 0xf8d5, 0xb3bd, 0xe0ea, 0xe4e1, 0xe4df, 0xe4e0, 0xe8e2,
03510 0xe8dd, 0xe8da, 0xe8e1, 0xe8e3, 0xbe7c, 0xe8e0, 0xe8dc, 0xe8db,
03511 0xe8df, 0xe8de, 0xbe7b, 0xec7d, 0xec78, 0xec76, 0xeca1, 0xec77,
03512 0xec73, 0xec79, 0xec74, 0xef72, 0xec75, 0xeca2, 0xec7c, 0xc06a,
03513 0xec7b, 0xec7a, 0xec7e, 0xef6a, 0xef6d, 0xef6c, 0xef74, 0xef6f,
03514 0xef73, 0xef71, 0xef70, 0xef6e, 0xef6b, 0xc243, 0xc242, 0xc244,
03515 0xc241, 0xef75, 0xf1c8, 0xf1cb, 0xf1c9, 0xf1cd, 0xf1ce, 0xf1c6,
03516 0xc358, 0xf1c7, 0xf1c5, 0xf1cc, 0xf1c4, 0xf1c3, 0xc357, 0xc355,
03517 0xc354, 0xf1ca, 0xf3cf, 0xf3d5, 0xc44a, 0xf3d0, 0xf3d3, 0xf3d7,
03518 0xc44b, 0xf3d2, 0xf3ca, 0xf3c9, 0xf3d6, 0xf3cd, 0xf3cb, 0xf3d4,
03519 0xf3cc, 0xc449, 0xc448, 0xf3c7, 0xf3c8, 0xf3d1, 0xf3ce, 0xf56c,
03520 0xf56f, 0xc356, 0xf56d, 0xf573, 0xf571, 0xf56b, 0xf576, 0xf56a,
03521 0xc4cf, 0xf572, 0xf57e, 0xc4ce, 0xf575, 0xf574, 0xf6ab, 0xf6aa,
03522 0xf6b1, 0xf6ad, 0xf6b0, 0xc560, 0xf6ae, 0xf6af, 0xf6a9, 0xf6ac,
03523 0xc55f, 0xc5bf, 0xf7b4, 0xf7af, 0xf7b3, 0xf7b6, 0xf7b2, 0xf7ae,
03524 0xc5c1, 0xf7b1, 0xf7b5, 0xc5c0, 0xf7ac, 0xf570, 0xf7b0, 0xf7ad,
03525 0xf7aa, 0xf7ab, 0xc5be, 0xf85a, 0xf85c, 0xf85f, 0xf85b, 0xf860,
03526 0xf859, 0xf857, 0xc5eb, 0xf85d, 0xc5ed, 0xc5ec, 0xf858, 0xf85e,
03527 0xf8da, 0xc64d, 0xc64d, 0xf8db, 0xf8d9, 0xf8d6, 0xf8d8, 0xf8d7, 0xf95a,
03528 0xf95c, 0xf95b, 0xf979, 0xf978, 0xf977, 0xf97a, 0xc673, 0xc674,
03529 0xf9ca, 0xf9ce, 0xb3be, 0xdcaf, 0xe0ed, 0xb9a7, 0xe0eb, 0xe0ec,
03530 0xe4e2, 0xe4e3, 0xbbf1, 0xbbef, 0xe4e4, 0xbbf0, 0xe8e8, 0xe8eb,
03531 0xe8e5, 0xe8ec, 0xe8e4, 0xe8e6, 0xe8e7, 0xe8ea, 0xbea1, 0xe8ef,
03532 0xe8ee, 0xbe7d, 0xe8e9, 0xe8ed, 0xbe7e, 0xecac, 0xc06f, 0xeca7,
03533 0xc06b, 0xecad, 0xecaa, 0xecad, 0xc070, 0xeca9, 0xeca6, 0xecae,
03534 0xeca5, 0xecab, 0xc06c, 0xeca3, 0xc06d, 0xc06e, 0xeca8, 0xefa9,
03535 0xef7a, 0xef7b, 0xef7e, 0xef7c, 0xef76, 0xef79, 0xefa5, 0xef7d,
03536 0xc245, 0xc64f, 0xefa7, 0xefa4, 0xc246, 0xefa6, 0xef77, 0xefa2, 0xefa3,
03537 0xefa1, 0xf1d2, 0xf1d4, 0xf1d7, 0xf1d1, 0xc359, 0xf1d9, 0xf1d0,
03538 0xf1da, 0xf1d6, 0xf1d8, 0xf1dc, 0xf1d5, 0xf1dd, 0xf1d3, 0xf1cf,
03539 0xc35a, 0xf1db, 0xc35b, 0xc44d, 0xef78, 0xf3f1, 0xf3e8, 0xc44f,
03540 0xf3e4, 0xc450, 0xf3ed, 0xf3e7, 0xf3dd, 0xc44e, 0xf3ea, 0xf3e5,
03541 0xf3e6, 0xf3d8, 0xf3df, 0xf3ee, 0xf3eb, 0xf3e3, 0xf3ef, 0xf3de,
03542 0xf3d9, 0xf3ec, 0xf3db, 0xf3e9, 0xf3e0, 0xf3f0, 0xf3dc, 0xc44c,
03543 0xf3da, 0xf3e1, 0xf3e2, 0xf57d, 0xf57b, 0xf5a2, 0xf5ae, 0xf5a5,
03544 0xf57c, 0xf578, 0xf5a7, 0xf57e, 0xf5a3, 0xf57a, 0xf5aa, 0xf577,
03545 0xf5a1, 0xf5a6, 0xf5a8, 0xf5ab, 0xf579, 0xf5af, 0xf5b0, 0xf5a9,
03546 0xf5ad, 0xf5a4, 0xf6c1, 0xf6c4, 0xc561, 0xf6c3, 0xf6c8, 0xf6c6,
03547 0xc562, 0xf6bd, 0xf6b3, 0xf6b2, 0xc564, 0xf6bf, 0xf6c0, 0xf6bc,
03548 0xf6b4, 0xf6b9, 0xf5ac, 0xf6b5, 0xc563, 0xf6bb, 0xf6ba, 0xf6b6,
03549 0xf6c2, 0xf6b7, 0xf7bb, 0xf6c5, 0xf6c7, 0xf6be, 0xf6b8, 0xf7bc,
03550 0xf7be, 0xf7b8, 0xc5c2, 0xf7c5, 0xf7c3, 0xc5c3, 0xf7c2, 0xf7c1,
03551 0xf7ba, 0xf7b7, 0xf7bd, 0xf7c6, 0xf7b9, 0xf7bf, 0xf869, 0xf86e,
03552 0xf864, 0xf867, 0xc5ee, 0xf86b, 0xf872, 0xf7c0, 0xf865, 0xf86f,
03553 0xf873, 0xf86a, 0xf863, 0xf86d, 0xf86c, 0xf871, 0xf870, 0xf7c4,
03554 0xf868, 0xf862, 0xf866, 0xc64e, 0xc64f, 0xf861, 0xf8e6, 0xf8dd,
03555 0xf8e5, 0xf8e2, 0xf8e3, 0xf8dc, 0xf8df, 0xf8e7, 0xf8e1, 0xf8e0,
03556 0xf8de, 0xf8e4, 0xf95d, 0xf95e, 0xf960, 0xf95f, 0xf962, 0xf961,
03557 0xf97c, 0xf97b, 0xf97d, 0xf9b8, 0xf9c5, 0xc678, 0xc67c, 0xf9cf,
03558 0xc67d, 0xb3bf, 0xc4d0, 0xf6c9, 0xc650, 0xc651, 0xb3c0, 0xe0ee,
03559 0xb9a8, 0xe8f0, 0xecb0, 0xecb1, 0xecaf, 0xefab, 0xefaa, 0xc247,
03560 0xf1df, 0xefac, 0xf1de, 0xf3f3, 0xc451, 0xc453, 0xf3f2, 0xc452,
03561 0xf5b1, 0xf5b3, 0xf5b2, 0xf6ca, 0xc565, 0xc5ef, 0xf8e8, 0xf963,
03562 0xf9d2, 0xb3c1, 0xe4e5, 0xbea2, 0xecb3, 0xecb2, 0xefad, 0xc454,
03563 0xc4d1, 0xf7c7, 0xf9cb, 0xb3c2, 0xbbf2, 0xbea3, 0xf3f4, 0xf874,
03564 0xb6c0, 0xefae, 0xc664, 0xb6c1, 0xbea4, 0xc248, 0xf875, 0xb6c2,
03565 0xe8f1, 0xc072, 0xecb4, 0xecb5, 0xc071, 0xefaf, 0xc24c, 0xc24a,
03566 0xc24b, 0xc249, 0xf1e0, 0xc35c, 0xf5b5, 0xf5b4, 0xf5b7, 0xf5b6,
03567 0xc4d2, 0xf6cb, 0xf6cd, 0xf6cc, 0xc566, 0xf7c8, 0xf876, 0xf877,
03568 0xc5f0, 0xf964, 0xf97d, 0xc675, 0xdc0b, 0xecb6, 0xefb0, 0xf3f5,
03569 0xe0ef, 0xefb1, 0xf1e2, 0xf1e1, 0xf878, 0xc652, 0xf965, 0xf97e,
03570 0xb9a9, 0xe8f2, 0xe8f3, 0xecb7, 0xb9aa, 0xc35d, 0xf1e3, 0xf6cf,
03571 0xc567, 0xf6d0, 0xf6ce, 0xf879, 0xf8e9, 0xb9ab, 0xefb4, 0xefb3,
03572 0xf6b2, 0xf1e4, 0xf1e8, 0xf1e7, 0xf1e6, 0xf1e5, 0xc35e, 0xf3f6,
03573 0xf5b9, 0xc4d3, 0xf5b8, 0xf6d1, 0xf7cb, 0xf7ca, 0xc5c4, 0xf7c9,
03574 0xf87c, 0xf87b, 0xf87a, 0xbbf3, 0xecb8, 0xc24d, 0xf3f7, 0xf3f8,
03575 0xf7cc, 0xf87d, 0xf8ea, 0xf966, 0xf9b9, 0xf9d4, 0xbbf4, 0xc24e,
03576 0xf1e9, 0xf3f9, 0xf6d2, 0xf87e, 0xbea6, 0xefb5, 0xf1ea, 0xf3fa,
03577 0xf3fb, 0xf3fc, 0xf5be, 0xf5ba, 0xc568, 0xf5bd, 0xf5bc, 0xc4d4,
03578 0xf5bb, 0xc4d6, 0xc4d5, 0xf6d4, 0xf6d3, 0xc569, 0xc56a, 0xc5c6,
03579 0xf7cd, 0xc5c5, 0xf8a3, 0xf8a4, 0xf8a2, 0xf8a1, 0xc654, 0xf8eb,
03580 0xf8ec, 0xf8ed, 0xc653, 0xf967, 0xf96a, 0xf969, 0xf968, 0xf9d3,
03581 0xc073, 0xc365, 0xf5bf, 0xf6d5, 0xc5c7, 0xf7ce, 0xf9d5, 0xc074,
03582 0xefb6, 0xf7cf, 0xf9a1, 0xc94a, 0xddfc, 0xa14a, 0xa157, 0xa159,
03583 0xa15b, 0xa15f, 0xa160, 0xa163, 0xa164, 0xa167, 0xa168, 0xa16b,
03584 0xa16c, 0xa16f, 0xa170, 0xa173, 0xa174, 0xa177, 0xa178, 0xa17b,
03585 0xa17c, 0xa1c6, 0xa1c7, 0xa1ca, 0xa1cb, 0xa1c8, 0xa1c9, 0xa15c,
03586 0xa14d, 0xa14f, 0xa151, 0xa152, 0xa153, 0xa154, 0xa17d, 0xa17e,
03587 0xa1a1, 0xa1a2, 0xa1a3, 0xa1a4, 0xa1cc, 0xa1cd, 0xa1ce, 0xa1de,
03588 0xa1df, 0xa1e0, 0xa1e1, 0xa1e2, 0xa24c, 0xa24d, 0xa24e, 0xa149,
03589 0xa1ad, 0xa243, 0xa248, 0xa1ae, 0xa15d, 0xa15e, 0xa1af, 0xa1cf,
03590 0xa141, 0xa1d0, 0xa144, 0xa241, 0xa2af, 0xa2b0, 0xa2b1, 0xa2b2,
03591 0xa2b3, 0xa2b4, 0xa2b5, 0xa2b6, 0xa2b7, 0xa2b8, 0xa147, 0xa146,
03592 0xa1d5, 0xa1d7, 0xa1d6, 0xa148, 0xa249, 0xa2cf, 0xa2d0, 0xa2d1,
03593 0xa2d2, 0xa2d3, 0xa2d4, 0xa2d5, 0xa2d6, 0xa2d7, 0xa2d8, 0xa2d9,

```
03594 0xa2da, 0xa2db, 0xa2dc, 0xa2dd, 0xa2de, 0xa2df, 0xa2e0, 0xa2e1,
03595 0xa2e2, 0xa2e3, 0xa2e4, 0xa2e5, 0xa2e6, 0xa2e7, 0xa2e8, 0xa2e9,
03596 0xa2ea, 0xa2eb, 0xa2ec, 0xa2ed, 0xa2ee, 0xa2ef,
03597 0xa2f0, 0xa2f1, 0xa2f2, 0xa2f3, 0xa2f4, 0xa2f5, 0xa2f6, 0xa2f7,
03598 0xa2f8, 0xa2f9, 0xa2fa, 0xa2fb, 0xa2fc, 0xa2fd, 0xa2fe, 0xa2ff,
03599 0xa300, 0xa301, 0xa302, 0xa303, 0xa304, 0xa305, 0xa306, 0xa307,
03600 };
03601
03602 static const Summary16 big5_uni2indx_page00[16] = {
03603     /* 0x0000 */
03604     { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 },
03605     { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 },
03606     { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x00ac }, { 4, 0x0083 },
03607     { 7, 0x0000 }, { 7, 0x0080 }, { 8, 0x0000 }, { 8, 0x0080 },
03608 };
03609 static const Summary16 big5_uni2indx_page02[38] = {
03610     /* 0x0200 */
03611     { 9, 0x0000 }, { 9, 0x0000 }, { 9, 0x0000 }, { 9, 0x0000 },
03612     { 9, 0x0000 }, { 9, 0x0000 }, { 9, 0x0000 }, { 9, 0x0000 },
03613     { 9, 0x0000 }, { 9, 0x0000 }, { 9, 0x0000 }, { 9, 0x0000 },
03614     { 9, 0x0080 }, { 13, 0x0200 }, { 14, 0x0000 }, { 14, 0x0000 },
03615     /* 0x0300 */
03616     { 14, 0x0000 }, { 14, 0x0000 }, { 14, 0x0000 }, { 14, 0x0000 },
03617     { 14, 0x0000 }, { 14, 0x0000 }, { 14, 0x0000 }, { 14, 0x0000 },
03618     { 14, 0x0000 }, { 14, 0xffff }, { 29, 0x03fb }, { 38, 0xffff },
03619     { 53, 0x03fb }, { 62, 0x0000 }, { 62, 0x0000 }, { 62, 0x0000 },
03620     /* 0x0400 */
03621     { 62, 0x0002 }, { 63, 0x1fff }, { 72, 0xffff }, { 85, 0xffff },
03622     { 101, 0xffff }, { 117, 0x0002 },
03623 };
03624 static const Summary16 big5_uni2indx_page20[44] = {
03625     /* 0x2000 */
03626     { 118, 0x0000 }, { 118, 0x3318 }, { 124, 0x0064 }, { 127, 0x4824 },
03627     { 131, 0x0000 }, { 131, 0x0000 }, { 131, 0x0000 }, { 131, 0x0000 },
03628     { 131, 0x0000 }, { 131, 0x0000 }, { 131, 0x0000 }, { 131, 0x0000 },
03629     { 131, 0x0000 }, { 131, 0x0000 }, { 131, 0x0000 }, { 131, 0x0000 },
03630     /* 0x2100 */
03631     { 131, 0x0228 }, { 134, 0x0000 }, { 134, 0x0000 }, { 134, 0x0000 },
03632     { 134, 0x0000 }, { 134, 0x0000 }, { 134, 0x03ff }, { 144, 0x0000 },
03633     { 144, 0x0000 }, { 144, 0x03cf }, { 152, 0x0000 }, { 152, 0x0000 },
03634     { 152, 0x0000 }, { 152, 0x0000 }, { 152, 0x0000 }, { 152, 0x0000 },
03635     /* 0x2200 */
03636     { 152, 0x0000 }, { 152, 0xc400 }, { 155, 0x4e29 }, { 162, 0x1030 },
03637     { 165, 0x0000 }, { 165, 0x0004 }, { 166, 0x00c3 }, { 170, 0x0000 },
03638     { 170, 0x0000 }, { 170, 0x0000 }, { 170, 0x0020 }, { 171, 0x8000 },
03639 };
03640 static const Summary16 big5_uni2indx_page24[37] = {
03641     /* 0x2400 */
03642     { 172, 0x0000 }, { 172, 0x0000 }, { 172, 0x0000 }, { 172, 0x0000 },
03643     { 172, 0x0000 }, { 172, 0x0000 }, { 172, 0x03ff }, { 182, 0x3fff },
03644     { 192, 0x0000 }, { 192, 0x0000 }, { 192, 0x0000 }, { 192, 0x0000 },
03645     { 192, 0x0000 }, { 192, 0x0000 }, { 192, 0x0000 }, { 192, 0x0000 },
03646     /* 0x2500 */
03647     { 192, 0x1005 }, { 195, 0x1111 }, { 199, 0x1010 }, { 201, 0x1010 },
03648     { 203, 0x0000 }, { 203, 0x4001 }, { 205, 0xe402 }, { 210, 0x000f },
03649     { 214, 0xffff }, { 229, 0x0030 }, { 231, 0x0003 }, { 233, 0x300c },
03650     { 237, 0xc8c0 }, { 242, 0x0000 }, { 242, 0x003c }, { 246, 0x0000 },
03651     /* 0x2600 */
03652     { 246, 0x0260 }, { 249, 0x0000 }, { 249, 0x0000 }, { 249, 0x0000 },
03653     { 249, 0x0007 },
03654 };
03655 static const Summary16 big5_uni2indx_page30[62] = {
03656     /* 0x3000 */
03657     { 252, 0xff2f }, { 265, 0x6037 }, { 272, 0x03fe }, { 281, 0x0000 },
03658     { 281, 0xffff }, { 296, 0xffff }, { 312, 0xffff }, { 328, 0xffff },
03659     { 344, 0xffff }, { 360, 0x600f }, { 366, 0xffff }, { 381, 0xffff },
03660     { 397, 0xffff }, { 413, 0xffff }, { 429, 0xffff }, { 445, 0x407f },
03661     /* 0x3100 */
03662     { 453, 0xffe0 }, { 464, 0xffff }, { 480, 0x03ff }, { 490, 0x0000 },
03663     { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 },
03664     { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 },
03665     { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 },
03666     /* 0x3200 */
03667     { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 },
03668     { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0000 },
03669     { 490, 0x0000 }, { 490, 0x0000 }, { 490, 0x0008 }, { 491, 0x0000 },
03670     { 491, 0x0000 }, { 491, 0x0000 }, { 491, 0x0000 }, { 491, 0x0000 },
03671     /* 0x3300 */
03672     { 491, 0x0000 }, { 491, 0x0000 }, { 491, 0x0000 }, { 491, 0x0000 },
03673     { 491, 0x0000 }, { 491, 0x0000 }, { 491, 0x0000 }, { 491, 0x0000 },
03674     { 491, 0xc000 }, { 493, 0x7000 }, { 496, 0x0002 }, { 497, 0x0000 },
03675     { 497, 0x4010 }, { 499, 0x0026 },
03676 };
03677 static const Summary16 big5_uni2indx_page4e[1307] = {
03678     /* 0x4e00 */
03679     { 502, 0xff8b }, { 514, 0xc373 }, { 523, 0x6840 }, { 527, 0x1b0f },
03680     { 535, 0xe9ac }, { 544, 0xf34c }, { 553, 0x0200 }, { 554, 0xc008 },
```

```
03681 { 557, 0x795c }, { 566, 0xca3e }, { 575, 0x7976 }, { 585, 0x0648 },
03682 { 589, 0x2fdf }, { 601, 0xf7f0 }, { 612, 0x033a }, { 618, 0xa8ff },
03683 /* 0x4f00 */
03684 { 629, 0xef37 }, { 641, 0x233f }, { 650, 0xb004 }, { 654, 0xfd59 },
03685 { 665, 0xf3ca }, { 675, 0xffff }, { 691, 0xde9f }, { 703, 0xffff9 },
03686 { 717, 0xabff }, { 730, 0x7df7 }, { 743, 0xc000 }, { 745, 0x8eec },
03687 { 754, 0xeebf }, { 767, 0xffdb }, { 781, 0xd003 }, { 786, 0x45fa },
03688 /* 0x5000 */
03689 { 795, 0xfae1 }, { 805, 0xdffe }, { 819, 0xbfef }, { 833, 0x10ab },
03690 { 839, 0xffeb }, { 853, 0xfcaa }, { 863, 0xef3f }, { 876, 0x24fd },
03691 { 885, 0x78ad }, { 894, 0x7f76 }, { 906, 0xf00c }, { 912, 0xedff },
03692 { 926, 0xcff6 }, { 938, 0x2cfa }, { 947, 0xf7f9 }, { 960, 0xeb6b },
03693 /* 0x5100 */
03694 { 971, 0x1ffd }, { 983, 0x95bf }, { 994, 0x6677 }, { 1004, 0xbfbf },
03695 { 1018, 0x3bfb }, { 1030, 0xfeb4 }, { 1041, 0x7bae }, { 1052, 0x11e2 },
03696 { 1058, 0xa681 }, { 1064, 0x41be }, { 1072, 0x1435 }, { 1078, 0x72c3 },
03697 { 1086, 0x7d70 }, { 1095, 0x7191 }, { 1102, 0x0003 }, { 1104, 0x276b },
03698 /* 0x5200 */
03699 { 1113, 0x57cb }, { 1123, 0x70cf }, { 1132, 0x4732 }, { 1139, 0x0def },
03700 { 1149, 0x7eda }, { 1160, 0xfc74 }, { 1170, 0xfe06 }, { 1179, 0xbdb4 },
03701 { 1189, 0x3f9f }, { 1201, 0x8bca }, { 1209, 0x7e49 }, { 1218, 0x5800 },
03702 { 1221, 0x228f }, { 1228, 0xebec }, { 1239, 0x8a5c }, { 1246, 0xddbb },
03703 /* 0x5300 */
03704 { 1258, 0xef60 }, { 1267, 0xb6e7 }, { 1278, 0xa40f }, { 1285, 0xf293 },
03705 { 1294, 0x37bb }, { 1305, 0x549e }, { 1313, 0xd04b }, { 1320, 0x9baf },
03706 { 1331, 0xc414 }, { 1336, 0xf7d4 }, { 1347, 0x30b0 }, { 1352, 0x0a14 },
03707 { 1356, 0x2f08 }, { 1362, 0x88d0 }, { 1367, 0xff7e }, { 1381, 0x192f },
03708 /* 0x5400 */
03709 { 1389, 0xffda }, { 1402, 0xfb07 }, { 1412, 0x7ff1 }, { 1424, 0x7beb },
03710 { 1436, 0xc5ef }, { 1447, 0x0010 }, { 1448, 0x99ff }, { 1460, 0xfddf },
03711 { 1475, 0x79d7 }, { 1486, 0x0567 }, { 1493, 0xffe7 }, { 1507, 0xfdc3 },
03712 { 1519, 0xc3ff }, { 1531, 0x4040 }, { 1533, 0x6ff7 }, { 1546, 0xbd8e },
03713 /* 0x5500 */
03714 { 1556, 0xdffa }, { 1569, 0x0497 }, { 1575, 0xf4c0 }, { 1582, 0x5bff },
03715 { 1595, 0xed7b }, { 1607, 0xd0e7 }, { 1616, 0x047e }, { 1623, 0xf8e0 },
03716 { 1631, 0xff9f }, { 1645, 0xb73e }, { 1656, 0x7dfe }, { 1669, 0x882e },
03717 { 1675, 0xfffd }, { 1690, 0xbe7f }, { 1703, 0x83fe }, { 1713, 0xf6c4 },
03718 /* 0x5600 */
03719 { 1722, 0xf357 }, { 1733, 0xb8fd }, { 1744, 0xd680 }, { 1750, 0xef7d },
03720 { 1763, 0x5767 }, { 1773, 0x4788 }, { 1779, 0xff7d }, { 1793, 0xc3df },
03721 { 1804, 0xf0ff }, { 1816, 0x37a9 }, { 1825, 0x7de0 }, { 1834, 0x70fc },
03722 { 1843, 0x3f6f }, { 1855, 0xec9a }, { 1864, 0x4cb3 }, { 1872, 0x8681 },
03723 /* 0x5700 */
03724 { 1877, 0x3f9e }, { 1888, 0xdd5c }, { 1898, 0xf70d }, { 1908, 0x4819 },
03725 { 1913, 0xfea3 }, { 1924, 0x0007 }, { 1927, 0xaf56 }, { 1937, 0x38ff },
03726 { 1948, 0x980d }, { 1954, 0xefb8 }, { 1965, 0x403d }, { 1971, 0xb760 },
03727 { 1979, 0xd8ce }, { 1988, 0x9035 }, { 1994, 0x72bf }, { 2005, 0x3fff },
03728 /* 0x5800 */
03729 { 2019, 0x7ff7 }, { 2033, 0x7a11 }, { 2040, 0xf7bb }, { 2053, 0xabff },
03730 { 2066, 0xff00 }, { 2074, 0x6fbe }, { 2086, 0xa93c }, { 2094, 0xfe72 },
03731 { 2105, 0xcfef }, { 2118, 0xf11b }, { 2127, 0xdb6b }, { 2138, 0xf40a },
03732 { 2145, 0xc3e6 }, { 2154, 0xef7e }, { 2167, 0x9b9c }, { 2176, 0xf610 },
03733 /* 0x5900 */
03734 { 2183, 0xf048 }, { 2189, 0x16f4 }, { 2197, 0xfeb5 }, { 2209, 0x5182 },
03735 { 2214, 0xc7b1 }, { 2223, 0x15bb }, { 2232, 0x6e87 }, { 2241, 0xfdbf },
03736 { 2255, 0xe43f }, { 2265, 0x63cd }, { 2274, 0xc1ff }, { 2285, 0x7e7e },
03737 { 2297, 0xfdeb }, { 2310, 0x7d5f }, { 2322, 0x777b }, { 2334, 0xfcfe },
03738 /* 0x5a00 */
03739 { 2347, 0x960b }, { 2354, 0xdbea }, { 2365, 0x6229 }, { 2371, 0x53e8 },
03740 { 2379, 0x37df }, { 2391, 0xfdef }, { 2405, 0x36f5 }, { 2415, 0xbd81 },
03741 { 2423, 0xdc18 }, { 2430, 0xfcbd }, { 2442, 0xd2e4 }, { 2450, 0xffff },
03742 { 2466, 0x3fd7 }, { 2478, 0xffe0 }, { 2489, 0x7f6f }, { 2502, 0xabf8 },
03743 /* 0x5b00 */
03744 { 2512, 0x9bae }, { 2522, 0x6ed9 }, { 2532, 0xf5fb }, { 2545, 0xf115 },
03745 { 2553, 0x79a9 }, { 2562, 0xbdfb }, { 2575, 0x5a3c }, { 2583, 0xadaf },
03746 { 2594, 0xdbba }, { 2605, 0x1fac }, { 2614, 0x71fc }, { 2624, 0x8379 },
03747 { 2632, 0x7cf7 }, { 2644, 0xc35f }, { 2654, 0xdfff }, { 2669, 0x0567 },
03748 /* 0x5c00 */
03749 { 2676, 0xff9a }, { 2688, 0x8467 }, { 2695, 0x1534 }, { 2701, 0xdf8b },
03750 { 2712, 0xf9f3 }, { 2724, 0xa373 }, { 2733, 0xf7bd }, { 2746, 0x5e1a },
03751 { 2754, 0xbf40 }, { 2762, 0xa03f }, { 2770, 0xffff }, { 2786, 0x01eb },
03752 { 2793, 0xdfc0 }, { 2802, 0xcfd9 }, { 2814, 0x7500 }, { 2819, 0xabd3 },
03753 /* 0x5d00 */
03754 { 2829, 0xf8c3 }, { 2838, 0xeed6 }, { 2849, 0x43fd }, { 2859, 0xb7ff },
03755 { 2873, 0x5eaf }, { 2884, 0x4227 }, { 2890, 0x9bac }, { 2899, 0xf686 },
03756 { 2908, 0x27d7 }, { 2918, 0xf6bc }, { 2929, 0xf787 }, { 2940, 0x35b7 },
03757 { 2950, 0xaacd }, { 2959, 0xe176 }, { 2968, 0x49e7 }, { 2977, 0xe29f },
03758 /* 0x5e00 */
03759 { 2987, 0x545c }, { 2994, 0xaf2 }, { 3005, 0x2b3f }, { 3015, 0x61d8 },
03760 { 3022, 0xfc3b }, { 3033, 0xbbb8 }, { 3043, 0xffcf }, { 3057, 0x7bd },
03761 { 3069, 0xbf95 }, { 3080, 0x1ce0 }, { 3086, 0x7dfd }, { 3099, 0x43ff },
03762 { 3110, 0x5ff6 }, { 3122, 0xfffe }, { 3137, 0xd3ef }, { 3149, 0xc4ce },
03763 /* 0x5f00 */
03764 { 3157, 0x8db6 }, { 3166, 0xadbc }, { 3176, 0x63dc }, { 3185, 0x11eb },
03765 { 3193, 0xdf59 }, { 3204, 0x23d0 }, { 3210, 0xbdb4 }, { 3220, 0xf3db },
03766 { 3232, 0x1fe7 }, { 3243, 0xdbc7 }, { 3254, 0xff63 }, { 3266, 0xfae4 },
03767 { 3276, 0xb22b }, { 3284, 0x63f7 }, { 3295, 0xed3b }, { 3306, 0xadba },
```

```

03768  /* 0x6000 */
03769  { 3316, 0xfe01 }, { 3324, 0x7eff }, { 3338, 0xffff7 }, { 3353, 0x02bc },
03770  { 3359, 0x32ff }, { 3370, 0xef39 }, { 3381, 0xffffc }, { 3395, 0x8005 },
03771  { 3398, 0x77fb }, { 3411, 0xbc5f }, { 3422, 0x010d }, { 3426, 0xffff7 },
03772  { 3441, 0xfffb }, { 3456, 0xbf3a }, { 3467, 0x0057 }, { 3472, 0xdfff },
03773  /* 0x6100 */
03774  { 3487, 0xef7b }, { 3500, 0xbd7d }, { 3512, 0xdb88 }, { 3520, 0xc8d4 },
03775  { 3527, 0xffff3 }, { 3541, 0xed7c }, { 3552, 0x5dee }, { 3563, 0x56ff },
03776  { 3575, 0x7e0d }, { 3584, 0xac5f }, { 3594, 0xff96 }, { 3606, 0xd57f },
03777  { 3618, 0x3fee }, { 3630, 0xc140 }, { 3634, 0x6ff9 }, { 3646, 0xffe7 },
03778  /* 0x6200 */
03779  { 3660, 0x779b }, { 3671, 0x8e77 }, { 3681, 0x6ebf }, { 3693, 0xe45d },
03780  { 3702, 0x6fcf }, { 3714, 0x5f1f }, { 3725, 0xe07f }, { 3735, 0xfedf },
03781  { 3749, 0xd7db }, { 3761, 0x01fe }, { 3769, 0xff00 }, { 3777, 0xfb7b },
03782  { 3790, 0xffd4 }, { 3802, 0x1fdf }, { 3814, 0xf800 }, { 3819, 0xffff },
03783  /* 0x6300 */
03784  { 3835, 0xfb8f }, { 3847, 0x007b }, { 3853, 0xbf00 }, { 3860, 0x7f5c },
03785  { 3871, 0xfffff }, { 3887, 0x07f3 }, { 3896, 0xeba0 }, { 3904, 0x3de7 },
03786  { 3915, 0xf7bf }, { 3929, 0xfbd7 }, { 3942, 0xffbf }, { 3957, 0x6003 },
03787  { 3961, 0xfffd }, { 3976, 0xbfed }, { 3989, 0xefbb }, { 4002, 0x027f },
03788  /* 0x6400 */
03789  { 4010, 0xfe40 }, { 4018, 0xddfd }, { 4031, 0xfdff }, { 4046, 0xe2f9 },
03790  { 4056, 0x680b }, { 4062, 0xfbf1f }, { 4074, 0xfbe3 }, { 4086, 0xaffd },
03791  { 4099, 0x9fa4 }, { 4108, 0xf7ed }, { 4121, 0x7a7d }, { 4132, 0xf80f },
03792  { 4141, 0xeebe }, { 4153, 0x0fd5 }, { 4162, 0xbb5d }, { 4173, 0xfd9f },
03793  /* 0x6500 */
03794  { 4186, 0xf2db }, { 4197, 0x3bf9 }, { 4208, 0xfe7f }, { 4222, 0xebcc },
03795  { 4232, 0x876a }, { 4240, 0x73fa }, { 4251, 0x95fc }, { 4261, 0x9ffc },
03796  { 4273, 0x109f }, { 4280, 0xfaf7 }, { 4293, 0xddb7 }, { 4305, 0xbbcd },
03797  { 4316, 0xf87e }, { 4327, 0xeecd }, { 4337, 0xf366 }, { 4347, 0x3c3f },
03798  /* 0x6600 */
03799  { 4357, 0xffffd }, { 4372, 0xb03f }, { 4381, 0xe9f7 }, { 4393, 0x067e },
03800  { 4401, 0x96ae }, { 4410, 0xfe06 }, { 4419, 0xd576 }, { 4429, 0x5fd7 },
03801  { 4441, 0x3fd1 }, { 4451, 0xa3f3 }, { 4461, 0xcf07 }, { 4470, 0x6fb7 },
03802  { 4482, 0x9fd1 }, { 4492, 0x7f44 }, { 4501, 0x7b59 }, { 4511, 0xd3dd },
03803  /* 0x6700 */
03804  { 4522, 0xaf3b }, { 4533, 0xa9bd }, { 4543, 0x7dcf }, { 4555, 0xff3a },
03805  { 4567, 0xfbe0 }, { 4577, 0xf6eb }, { 4589, 0xb401 }, { 4594, 0xfffff },
03806  { 4610, 0x7afa }, { 4621, 0xb7bf }, { 4634, 0xc000 }, { 4636, 0x0ffd },
03807  { 4647, 0xff7f }, { 4662, 0xff1f }, { 4675, 0xfefc }, { 4688, 0x95ff },
03808  /* 0x6800 */
03809  { 4700, 0x0000 }, { 4700, 0xb5dc }, { 4710, 0xef63 }, { 4721, 0x3f3e },
03810  { 4732, 0xfb7f }, { 4746, 0x001b }, { 4750, 0xe800 }, { 4754, 0xfb6f },
03811  { 4767, 0x9eef }, { 4779, 0xb8df }, { 4790, 0xff9f }, { 4804, 0x003f },
03812  { 4810, 0x7bd0 }, { 4819, 0xf5ff }, { 4833, 0xd5db }, { 4846, 0x3fff },
03813  /* 0x6900 */
03814  { 4860, 0xfdf0 }, { 4871, 0x00bf }, { 4878, 0x8420 }, { 4881, 0xbbbd },
03815  { 4893, 0xdf37 }, { 4905, 0xffde }, { 4919, 0xff6d }, { 4932, 0x0ff3 },
03816  { 4942, 0x604c }, { 4947, 0x5efb }, { 4959, 0xffffb }, { 4974, 0xfafb },
03817  { 4987, 0xfe5e }, { 4999, 0x0219 }, { 5003, 0x79f4 }, { 5013, 0xf9de },
03818  /* 0x6a00 */
03819  { 5025, 0xa7f7 }, { 5037, 0xebfa }, { 5049, 0x01eb }, { 5056, 0xff34 },
03820  { 5067, 0xebd3 }, { 5078, 0xef73 }, { 5090, 0xafd7 }, { 5102, 0xc040 },
03821  { 5105, 0x72bb }, { 5115, 0xdcff }, { 5128, 0xf17f }, { 5140, 0x2fd8 },
03822  { 5149, 0xb8ec }, { 5158, 0xfe0b }, { 5168, 0xdda3 }, { 5178, 0x1f0b },
03823  /* 0x6b00 */
03824  { 5186, 0x8f1d }, { 5195, 0x47cf }, { 5205, 0xb12b }, { 5213, 0xffde },
03825  { 5227, 0x7fee }, { 5240, 0xda73 }, { 5250, 0x24ff }, { 5260, 0xc0c4 },
03826  { 5268, 0xf75d }, { 5280, 0xcbf2 }, { 5290, 0xecfd }, { 5302, 0xb4ed },
03827  { 5312, 0xbff9 }, { 5325, 0x4ddd }, { 5335, 0x99dd }, { 5345, 0xf8d8 },
03828  /* 0x6c00 */
03829  { 5356, 0xbb7f }, { 5369, 0xaf7b }, { 5381, 0xddfb }, { 5394, 0xc959 },
03830  { 5402, 0xfc4f }, { 5413, 0xfab5 }, { 5424, 0xafe3 }, { 5435, 0x6d5f },
03831  { 5446, 0xfffff }, { 5462, 0x3f7d }, { 5474, 0x7800 }, { 5478, 0xffdb },
03832  { 5492, 0xb6ff }, { 5505, 0x7eff }, { 5519, 0xfba7 }, { 5532, 0x022f },
03833  /* 0x6d00 */
03834  { 5538, 0xff9b }, { 5551, 0xefc7 }, { 5563, 0xffa5 }, { 5575, 0xfffff },
03835  { 5591, 0x0007 }, { 5594, 0xc700 }, { 5599, 0xf7ff }, { 5614, 0xffff1 },
03836  { 5627, 0x7ffd }, { 5641, 0x01bf }, { 5649, 0xdc00 }, { 5654, 0xfdbc },
03837  { 5666, 0xbff5 }, { 5679, 0xfffff }, { 5695, 0xff7f }, { 5710, 0x3eff },
03838  /* 0x6e00 */
03839  { 5723, 0x0029 }, { 5726, 0xbe00 }, { 5732, 0xf9ff }, { 5746, 0xff7f },
03840  { 5761, 0x6efb }, { 5773, 0xf7e }, { 5786, 0xcbff }, { 5799, 0x039e },
03841  { 5806, 0xe300 }, { 5811, 0xfbdd }, { 5824, 0xccff }, { 5836, 0xf6df },
03842  { 5849, 0xfffff }, { 5865, 0x117f }, { 5874, 0xf800 }, { 5879, 0xfb6f },
03843  /* 0x6f00 */
03844  { 5892, 0xe7ef }, { 5905, 0xd73c }, { 5915, 0xfeef }, { 5929, 0xdfe7 },
03845  { 5943, 0xc00b }, { 5948, 0xedbf }, { 5961, 0xfedf }, { 5975, 0xfdc7 },
03846  { 5987, 0x7bf5 }, { 5999, 0x40fd }, { 6007, 0xfffff }, { 6023, 0xb75f },
03847  { 6035, 0xffdf }, { 6050, 0xf930 }, { 6058, 0xfdbf }, { 6072, 0xdc97 },
03848  /* 0x7000 */
03849  { 6082, 0xfef3 }, { 6095, 0xbff2 }, { 6107, 0x8fdf }, { 6119, 0xdfbf },
03850  { 6133, 0x177f }, { 6144, 0xede6 }, { 6155, 0x0f7f }, { 6166, 0x3553 },
03851  { 6174, 0x447c }, { 6181, 0x877e }, { 6191, 0xfa12 }, { 6199, 0x45bb },
03852  { 6208, 0xede0 }, { 6217, 0x779e }, { 6228, 0x8017 }, { 6233, 0xbfd9 },
03853  /* 0x7100 */
03854  { 6245, 0x7e55 }, { 6255, 0xde89 }, { 6264, 0xc16f }, { 6273, 0x0447 },

```

```
03855 { 6278, 0x7ade }, { 6289, 0xf75d }, { 6301, 0x57ff }, { 6314, 0x2905 },
03856 { 6319, 0x86f7 }, { 6329, 0xfe95 }, { 6340, 0x97b3 }, { 6350, 0xf32f },
03857 { 6361, 0xcfff }, { 6375, 0x9f75 }, { 6386, 0x71f7 }, { 6397, 0xfb17 },
03858 /* 0x7200 */
03859 { 6408, 0x34ee }, { 6417, 0xee19 }, { 6426, 0x37cc }, { 6435, 0xef61 },
03860 { 6445, 0x9fd6 }, { 6456, 0xef4c }, { 6466, 0xd68f }, { 6476, 0xfbdd },
03861 { 6489, 0x7b73 }, { 6500, 0x6def }, { 6512, 0xd7fe }, { 6525, 0xa431 },
03862 { 6531, 0x5e7f }, { 6543, 0x97d7 }, { 6554, 0x0f5b }, { 6563, 0xffd8 },
03863 /* 0x7300 */
03864 { 6575, 0x9d83 }, { 6583, 0x7bce }, { 6594, 0x22ec }, { 6601, 0xdcff },
03865 { 6614, 0x763d }, { 6624, 0xef87 }, { 6635, 0xdfe7 }, { 6648, 0xfded },
03866 { 6661, 0x4fff }, { 6674, 0xa0fc }, { 6682, 0x3b77 }, { 6693, 0xdbfc },
03867 { 6705, 0x3ded }, { 6716, 0x7fdc }, { 6728, 0x6fa9 }, { 6738, 0xf570 },
03868 /* 0x7400 */
03869 { 6747, 0x3ffb }, { 6760, 0x2c40 }, { 6764, 0xff7f }, { 6779, 0x847f },
03870 { 6788, 0xec57 }, { 6798, 0xdeb7 }, { 6810, 0xe69c }, { 6819, 0xf22f },
03871 { 6829, 0x0feb }, { 6839, 0xd5b5 }, { 6849, 0xafeb }, { 6861, 0xede7 },
03872 { 6873, 0x8c2f }, { 6881, 0xffff }, { 6893, 0x537f }, { 6904, 0xe8f0 },
03873 /* 0x7500 */
03874 { 6912, 0xb99d }, { 6922, 0xb5ff }, { 6935, 0xff66 }, { 6947, 0xe78f },
03875 { 6958, 0xd981 }, { 6965, 0xbe10 }, { 6972, 0x9c7c }, { 6981, 0xe3c1 },
03876 { 6989, 0x9cd1 }, { 6997, 0x2733 }, { 7005, 0x0cbc }, { 7012, 0xff6d },
03877 { 7025, 0xfcb7 }, { 7037, 0xfcb7 }, { 7050, 0xa0df }, { 7059, 0xffff },
03878 /* 0x7600 */
03879 { 7075, 0xbf0b }, { 7085, 0xfe7b }, { 7098, 0xa3ff }, { 7110, 0x353f },
03880 { 7120, 0x13cc }, { 7127, 0x97cd }, { 7137, 0x7637 }, { 7147, 0xfb27 },
03881 { 7158, 0xcfd6 }, { 7169, 0x7e6c }, { 7179, 0xec50 }, { 7186, 0xed31 },
03882 { 7195, 0x677c }, { 7205, 0xfc1c }, { 7214, 0xf6fa }, { 7226, 0x5fbf },
03883 /* 0x7700 */
03884 { 7239, 0x0fba }, { 7248, 0xae2f }, { 7258, 0xa3ad }, { 7267, 0x7ffe },
03885 { 7281, 0xfc0f }, { 7291, 0xde74 }, { 7301, 0xffef }, { 7316, 0xf200 },
03886 { 7321, 0xfbbf }, { 7335, 0xfea2 }, { 7345, 0x3daf }, { 7356, 0xbccf },
03887 { 7369, 0xf694 }, { 7378, 0x5fb9 }, { 7389, 0xf3ad }, { 7400, 0x3f8f },
03888 /* 0x7800 */
03889 { 7411, 0xf26c }, { 7420, 0xa01f }, { 7427, 0xffef }, { 7442, 0x01bf },
03890 { 7450, 0x7728 }, { 7458, 0x7005 }, { 7463, 0xff35 }, { 7475, 0xda03 },
03891 { 7482, 0xd2f9 }, { 7492, 0xc7fa }, { 7503, 0x3fbf }, { 7516, 0x5c1d },
03892 { 7524, 0xff3a }, { 7536, 0xec33 }, { 7545, 0xb7af }, { 7557, 0xfe9c },
03893 /* 0x7900 */
03894 { 7568, 0x5236 }, { 7575, 0x7a9f }, { 7586, 0xbffa }, { 7599, 0xe722 },
03895 { 7607, 0x9ff7 }, { 7620, 0xfcff }, { 7634, 0x2fbb }, { 7645, 0xb61d },
03896 { 7654, 0xed06 }, { 7662, 0x1dfd }, { 7673, 0x7dd7 }, { 7685, 0xefdf },
03897 { 7699, 0xeb23 }, { 7708, 0xf166 }, { 7717, 0x7ed9 }, { 7728, 0x0dc0 },
03898 /* 0x7a00 */
03899 { 7733, 0x3d3d }, { 7743, 0xdfbf }, { 7757, 0xc945 }, { 7764, 0xba83 },
03900 { 7772, 0x7dd1 }, { 7782, 0x9dd0 }, { 7790, 0x7b87 }, { 7800, 0xcf73 },
03901 { 7811, 0x9ff3 }, { 7823, 0xc3f5 }, { 7833, 0xdf0d }, { 7843, 0xc5fe },
03902 { 7854, 0x0cb3 }, { 7861, 0x8302 }, { 7865, 0xe879 }, { 7874, 0xaec0 },
03903 /* 0x7b00 */
03904 { 7881, 0xc773 }, { 7891, 0x6f0f }, { 7901, 0xfd7d }, { 7914, 0x093f },
03905 { 7922, 0xffff }, { 7935, 0x0157 }, { 7941, 0x62fb }, { 7951, 0x01ff },
03906 { 7960, 0xfdb4 }, { 7971, 0x3bf3 }, { 7982, 0xb013 }, { 7988, 0x43b2 },
03907 { 7995, 0x5ed3 }, { 8005, 0xff30 }, { 8015, 0x0fff }, { 8027, 0xeb9f },
03908 /* 0x7c00 */
03909 { 8039, 0xfeef }, { 8053, 0xf203 }, { 8060, 0x3fef }, { 8073, 0xfb89 },
03910 { 8083, 0x37a9 }, { 8092, 0x9e99 }, { 8101, 0xdef9 }, { 8113, 0xa72c },
03911 { 8121, 0x3733 }, { 8130, 0xc1f6 }, { 8139, 0x812e }, { 8145, 0xfe3e },
03912 { 8157, 0x5d20 }, { 8163, 0xf2f7 }, { 8175, 0xd585 }, { 8183, 0x69d7 },
03913 /* 0x7d00 */
03914 { 8193, 0xffff }, { 8209, 0xffff }, { 8225, 0xdb07 }, { 8234, 0xff6f },
03915 { 8248, 0xc4ff }, { 8259, 0xd97f }, { 8271, 0xefce }, { 8283, 0xbe0f },
03916 { 8293, 0xf17b }, { 8304, 0xf05e }, { 8313, 0xf6cf }, { 8325, 0xffb7 },
03917 { 8339, 0x5ef7 }, { 8351, 0xef84 }, { 8360, 0xd7cb }, { 8371, 0x0edf },
03918 /* 0x7e00 */
03919 { 8381, 0xff08 }, { 8390, 0xfcff }, { 8404, 0xee3f }, { 8416, 0xffff },
03920 { 8432, 0x13ff }, { 8443, 0xd7ff }, { 8457, 0xaf0f }, { 8467, 0x7ffd },
03921 { 8481, 0xbdc7 }, { 8492, 0x1ffa }, { 8503, 0x0000 }, { 8503, 0x0000 },
03922 { 8503, 0x0000 }, { 8503, 0x0000 }, { 8503, 0x0000 }, { 8503, 0x0000 },
03923 /* 0x7f00 */
03924 { 8503, 0x0000 }, { 8503, 0x0000 }, { 8503, 0x0000 }, { 8503, 0xe740 },
03925 { 8510, 0xbd38 }, { 8519, 0xf933 }, { 8529, 0x7feb }, { 8542, 0xfeed },
03926 { 8555, 0x7fe8 }, { 8566, 0x7c76 }, { 8576, 0xb3f7 }, { 8588, 0xffef },
03927 { 8603, 0xfeaf }, { 8616, 0xd8b7 }, { 8626, 0xff6f }, { 8640, 0xfbbf },
03928 /* 0x8000 */
03929 { 8654, 0xf8fb }, { 8666, 0xdbf7 }, { 8679, 0x1752 }, { 8686, 0xe2f9 },
03930 { 8696, 0x85c8 }, { 8702, 0x7547 }, { 8711, 0x9090 }, { 8715, 0xe3ef },
03931 { 8727, 0x9ef4 }, { 8737, 0x3f6d }, { 8748, 0xee2e }, { 8758, 0x0536 },
03932 { 8764, 0xf7bc }, { 8776, 0x7ff3 }, { 8789, 0xa07b }, { 8797, 0x7f3f },
03933 /* 0x8100 */
03934 { 8810, 0x0567 }, { 8817, 0xeb60 }, { 8825, 0xbabe }, { 8836, 0x6601 },
03935 { 8841, 0xfcd8 }, { 8851, 0x583f }, { 8860, 0xcaf7 }, { 8871, 0x87df },
03936 { 8882, 0xbfcd }, { 8894, 0xffa0 }, { 8904, 0x5bcd }, { 8914, 0xfebf },
03937 { 8928, 0xb6fd }, { 8940, 0xefa7 }, { 8952, 0x77ef }, { 8965, 0xdf9c },
03938 /* 0x8200 */
03939 { 8976, 0x3fb7 }, { 8988, 0xf877 }, { 8999, 0x9d27 }, { 9008, 0xb7fc },
03940 { 9020, 0xcab5 }, { 9029, 0xdfef }, { 9043, 0xfb5a }, { 9054, 0xf1b6 },
03941 { 9064, 0xec39 }, { 9073, 0xef1f }, { 9085, 0xfbbf }, { 9099, 0x7ffb },
```

```

03942 { 9113, 0x000d }, { 9116, 0xdaf6 }, { 9128, 0xbdff }, { 9141, 0x4e7f },
03943 /* 0x8300 */
03944 { 9152, 0x33ff }, { 9164, 0x5ac0 }, { 9170, 0xbff5 }, { 9183, 0x9ffe },
03945 { 9196, 0xffbf }, { 9211, 0x005f }, { 9217, 0x0000 }, { 9217, 0xfdf8 },
03946 { 9229, 0xffca }, { 9241, 0x6ffd }, { 9254, 0xcffd }, { 9267, 0xa001 },
03947 { 9270, 0xdfff }, { 9285, 0xfbf2 }, { 9297, 0xdfbf }, { 9311, 0xff7f },
03948 /* 0x8400 */
03949 { 9326, 0xfeda }, { 9338, 0x080f }, { 9343, 0xba08 }, { 9349, 0xbfff },
03950 { 9364, 0x7afd }, { 9376, 0xeed7 }, { 9388, 0xfbeb }, { 9401, 0x67f9 },
03951 { 9412, 0xe044 }, { 9417, 0xff93 }, { 9429, 0xdf97 }, { 9441, 0x9f57 },
03952 { 9452, 0xfef7 }, { 9466, 0x08df }, { 9474, 0xdf80 }, { 9482, 0xfedf },
03953 /* 0x8500 */
03954 { 9496, 0xffc5 }, { 9508, 0xf7fe }, { 9522, 0xffffb }, { 9537, 0x6803 },
03955 { 9542, 0x67fb }, { 9554, 0x6bfa }, { 9565, 0x7fff }, { 9580, 0x5fe2 },
03956 { 9590, 0xfffff }, { 9606, 0xff73 }, { 9619, 0x87df }, { 9630, 0xe7fb },
03957 { 9643, 0xebfd }, { 9656, 0xf7a7 }, { 9668, 0xbf7e }, { 9681, 0xefc7 },
03958 /* 0x8600 */
03959 { 9693, 0x1ef3 }, { 9703, 0xdf82 }, { 9712, 0x76ff }, { 9725, 0xdf7e },
03960 { 9738, 0x79c9 }, { 9747, 0xda7d }, { 9758, 0xefbe }, { 9771, 0x1e9b },
03961 { 9780, 0x7ce0 }, { 9788, 0x77fb }, { 9801, 0x87be }, { 9811, 0xffffb },
03962 { 9826, 0x1bff }, { 9838, 0xffdb }, { 9852, 0x3f5c }, { 9862, 0x4fe0 },
03963 /* 0x8700 */
03964 { 9870, 0x7fff }, { 9885, 0x5f0e }, { 9894, 0x77ff }, { 9908, 0xddbf },
03965 { 9921, 0xf04f }, { 9930, 0xfffff }, { 9946, 0xfffff }, { 9962, 0x0ff8 },
03966 { 9971, 0xa3be }, { 9981, 0xfddf }, { 9995, 0xfc1c }, { 10004, 0xffffd },
03967 { 10019, 0x1f7d }, { 10030, 0xfb9e }, { 10042, 0xbddf }, { 10056, 0xdedc },
03968 /* 0x8800 */
03969 { 10067, 0x3f6f }, { 10079, 0xbafb }, { 10091, 0xdf7f }, { 10105, 0xfbef },
03970 { 10119, 0x7d1b }, { 10129, 0x2eec }, { 10138, 0xaf8e }, { 10148, 0xf2f7 },
03971 { 10160, 0x7b0f }, { 10170, 0xcfee }, { 10182, 0x1d96 }, { 10190, 0x77c6 },
03972 { 10200, 0x7e07 }, { 10209, 0xffff5 }, { 10223, 0xd982 }, { 10230, 0x7fdf },
03973 /* 0x8900 */
03974 { 10244, 0x5ee6 }, { 10254, 0xc7ff }, { 10267, 0xfeee }, { 10280, 0x79ef },
03975 { 10292, 0x9a56 }, { 10300, 0xffcf }, { 10314, 0xfe5f }, { 10327, 0xde5e },
03976 { 10338, 0x896e }, { 10346, 0xf9e8 }, { 10356, 0xf45e }, { 10366, 0xe6c4 },
03977 { 10374, 0x0001 }, { 10375, 0xbe7c }, { 10386, 0x3b7f }, { 10398, 0xdddf },
03978 /* 0x8a00 */
03979 { 10411, 0xd59d }, { 10421, 0xe9ef }, { 10433, 0x34ac }, { 10440, 0xde53 },
03980 { 10450, 0xf573 }, { 10461, 0x4bf7 }, { 10472, 0x7b4f }, { 10483, 0x9eff },
03981 { 10496, 0xb8fe }, { 10507, 0x476e }, { 10516, 0x0dfb }, { 10526, 0xffff45 },
03982 { 10537, 0xabfd }, { 10549, 0xfbfef }, { 10563, 0xe9d7 }, { 10574, 0xddff },
03983 /* 0x8b00 */
03984 { 10588, 0xedf7 }, { 10601, 0x7fff }, { 10616, 0xddfd }, { 10629, 0x7eeb },
03985 { 10641, 0xcfe7 }, { 10653, 0xb7ff }, { 10667, 0xbde9 }, { 10678, 0xef91 },
03986 { 10688, 0x5d75 }, { 10698, 0xd77c }, { 10709, 0x0000 }, { 10709, 0x0000 },
03987 { 10709, 0x0000 }, { 10709, 0x0000 }, { 10709, 0x0000 }, { 10709, 0x0000 },
03988 /* 0x8c00 */
03989 { 10709, 0x0000 }, { 10709, 0x0000 }, { 10709, 0x0000 }, { 10709, 0xfa80 },
03990 { 10716, 0xffee }, { 10730, 0xb4f1 }, { 10739, 0xbf76 }, { 10751, 0x2fef },
03991 { 10763, 0xb677 }, { 10774, 0x77bf }, { 10787, 0x9fbf }, { 10800, 0xffffd },
03992 { 10815, 0x95bf }, { 10826, 0xf6ae }, { 10837, 0x75ff }, { 10850, 0x7f3b },
03993 /* 0x8d00 */
03994 { 10862, 0xa7f5 }, { 10873, 0x0af9 }, { 10881, 0x0000 }, { 10881, 0x0000 },
03995 { 10881, 0x0000 }, { 10881, 0x0000 }, { 10881, 0xfbd0 }, { 10891, 0x2bdd },
03996 { 10901, 0xf633 }, { 10911, 0x9a7f }, { 10922, 0xfdab }, { 10934, 0xd6fc },
03997 { 10945, 0xf9e6 }, { 10956, 0xbfeb }, { 10969, 0xdfdf }, { 10983, 0xf41f },
03998 /* 0x8e00 */
03999 { 10993, 0xa6fd }, { 11004, 0xfffff }, { 11020, 0x4aff }, { 11031, 0xf37b },
04000 { 11043, 0x7fb7 }, { 11056, 0xfef9 }, { 11069, 0xb6ff }, { 11082, 0x1d5c },
04001 { 11090, 0x7ff6 }, { 11103, 0xe5ff }, { 11116, 0x1f7b }, { 11127, 0x2404 },
04002 { 11130, 0xbe05 }, { 11138, 0xf99e }, { 11149, 0xdbe3 }, { 11160, 0xdfd2 },
04003 /* 0x8f00 */
04004 { 11172, 0x6fef }, { 11185, 0xfdff }, { 11200, 0xd679 }, { 11210, 0xcbfc },
04005 { 11221, 0xebfd }, { 11234, 0xfffff }, { 11249, 0x001f }, { 11254, 0x0000 },
04006 { 11254, 0x0000 }, { 11254, 0x9800 }, { 11257, 0xe148 }, { 11263, 0x8017 },
04007 { 11268, 0x6a74 }, { 11276, 0x00fe }, { 11283, 0x6d7f }, { 11295, 0xfdfl },
04008 /* 0x9000 */
04009 { 11307, 0xb87f }, { 11318, 0xfef3 }, { 11331, 0xe01f }, { 11339, 0xf176 },
04010 { 11349, 0xee96 }, { 11359, 0x7b3f }, { 11371, 0xeb8d }, { 11381, 0xffffd },
04011 { 11396, 0xadff }, { 11409, 0xcbb3 }, { 11419, 0x84ef }, { 11428, 0xe17f },
04012 { 11439, 0xad4a }, { 11447, 0xbff0 }, { 11458, 0xbf3f }, { 11471, 0xfe3f },
04013 /* 0x9100 */
04014 { 11484, 0xebff }, { 11498, 0xffd7 }, { 11512, 0xffdf }, { 11527, 0xcf7f },
04015 { 11540, 0xfffff }, { 11555, 0x85ed }, { 11564, 0xd73f }, { 11576, 0x07bc },
04016 { 11584, 0xaeff }, { 11597, 0xfe0f }, { 11608, 0xfdaf }, { 11621, 0x76bf },
04017 { 11633, 0xfaef }, { 11646, 0x37bb }, { 11657, 0x7fdc }, { 11669, 0xa3ba },
04018 /* 0x9200 */
04019 { 11678, 0xb6ff }, { 11691, 0x56f7 }, { 11702, 0x60f8 }, { 11709, 0xe7df },
04020 { 11722, 0xff61 }, { 11733, 0x4cdf }, { 11743, 0xb0fb }, { 11753, 0xff45 },
04021 { 11764, 0x7ded }, { 11776, 0x3ffa }, { 11788, 0xfffff }, { 11801, 0x18fc },
04022 { 11809, 0xfffff }, { 11825, 0xe3af }, { 11836, 0xc7d3 }, { 11846, 0xdf83 },
04023 /* 0x9300 */
04024 { 11856, 0xfb57 }, { 11868, 0xef7d }, { 11881, 0xfffff }, { 11896, 0x1378 },
04025 { 11903, 0xfec0 }, { 11912, 0x5ff7 }, { 11925, 0x34bb }, { 11934, 0x5ee3 },
04026 { 11944, 0xf70d }, { 11954, 0xfffff }, { 11967, 0xd7fe }, { 11980, 0x00bf },
04027 { 11987, 0xf59d }, { 11998, 0xf7f7 }, { 12012, 0x51de }, { 12021, 0xffe0 },
04028 /* 0x9400 */

```

```

04029 { 12032, 0xfec9 }, { 12043, 0x037f }, { 12052, 0x5f01 }, { 12059, 0xbfef },
04030 { 12073, 0x9ff1 }, { 12084, 0x60a7 }, { 12091, 0xef1d }, { 12102, 0xf1ff },
04031 { 12115, 0x000f }, { 12119, 0x0000 }, { 12119, 0x0000 }, { 12119, 0x0000 },
04032 { 12119, 0x0000 }, { 12119, 0x0000 }, { 12119, 0x0000 }, { 12119, 0x0000 },
04033 /* 0x9500 */
04034 { 12119, 0x0000 }, { 12119, 0x0000 }, { 12119, 0x0000 }, { 12119, 0x0000 },
04035 { 12119, 0x0000 }, { 12119, 0x0000 }, { 12119, 0x0000 }, { 12119, 0x3c80 },
04036 { 12124, 0xfbd4 }, { 12135, 0xd91f }, { 12145, 0x7b3a }, { 12155, 0xfef3 },
04037 { 12167, 0x3fe9 }, { 12178, 0xdc7f }, { 12190, 0x003f }, { 12196, 0x0000 },
04038 /* 0x9600 */
04039 { 12196, 0x0000 }, { 12196, 0x5000 }, { 12198, 0xf51f }, { 12209, 0xbe07 },
04040 { 12218, 0xfc1d }, { 12228, 0xf91b }, { 12238, 0xbc1e }, { 12247, 0x71ff },
04041 { 12259, 0x6ff9 }, { 12271, 0x5bbe }, { 12282, 0x5796 }, { 12291, 0x9b1b },
04042 { 12300, 0x7fff }, { 12315, 0xffffc }, { 12329, 0x872e }, { 12337, 0xaf7e },
04043 /* 0x9700 */
04044 { 12349, 0xebf5 }, { 12361, 0xf34f }, { 12372, 0xdffd }, { 12386, 0xe725 },
04045 { 12395, 0x0bdc }, { 12403, 0x5d44 }, { 12410, 0x5747 }, { 12419, 0xfddd },
04046 { 12432, 0xed3f }, { 12444, 0x7790 }, { 12452, 0x7d7f }, { 12465, 0x8ac8 },
04047 { 12471, 0xfafa }, { 12483, 0xf3f9 }, { 12495, 0x202a }, { 12499, 0xef4b },
04048 /* 0x9800 */
04049 { 12510, 0xf5ff }, { 12524, 0x79cf }, { 12535, 0xabd3 }, { 12545, 0x0ba5 },
04050 { 12552, 0xf77a }, { 12564, 0xfb8f }, { 12576, 0x8ebd }, { 12586, 0x001f },
04051 { 12591, 0x0000 }, { 12591, 0x0000 }, { 12591, 0xf300 }, { 12597, 0xfd4e },
04052 { 12608, 0x1a57 }, { 12616, 0x8800 }, { 12618, 0xaeac }, { 12627, 0x7654 },
04053 /* 0x9900 */
04054 { 12635, 0x17ad }, { 12644, 0xcdff }, { 12657, 0xffb2 }, { 12669, 0xf42f },
04055 { 12679, 0x5baa }, { 12688, 0xdbff }, { 12702, 0x0002 }, { 12703, 0x0000 },
04056 { 12703, 0x0000 }, { 12703, 0x73c0 }, { 12710, 0xf9ea }, { 12721, 0x2e3f },
04057 { 12731, 0xfa8e }, { 12741, 0xbbff }, { 12755, 0x76bc }, { 12765, 0xffd3 },
04058 /* 0x9a00 */
04059 { 12778, 0xeefe }, { 12791, 0x7e72 }, { 12801, 0x7ebd }, { 12813, 0xe7f7 },
04060 { 12826, 0xf77f }, { 12840, 0xcefd }, { 12852, 0x0ff5 }, { 12862, 0x0000 },
04061 { 12862, 0x0000 }, { 12862, 0x0000 }, { 12862, 0xa900 }, { 12866, 0xdb9b },
04062 { 12877, 0xa4c7 }, { 12885, 0x917f }, { 12895, 0xf8ca }, { 12904, 0x7ece },
04063 /* 0x9b00 */
04064 { 12915, 0x7d7a }, { 12926, 0xc7e7 }, { 12937, 0xcbbd }, { 12948, 0xdcae },
04065 { 12958, 0xfd7e }, { 12971, 0x8f76 }, { 12981, 0x91d3 }, { 12989, 0x7cf3 },
04066 { 13000, 0x01e5 }, { 13006, 0x4c2f }, { 13014, 0xed77 }, { 13026, 0xa360 },
04067 { 13032, 0x07db }, { 13041, 0x5ef8 }, { 13051, 0x1df7 }, { 13062, 0x2181 },
04068 /* 0x9c00 */
04069 { 13066, 0x6be0 }, { 13074, 0x309c }, { 13080, 0x3b3a }, { 13089, 0xfade },
04070 { 13101, 0x7f53 }, { 13112, 0xc3f5 }, { 13122, 0x61cd }, { 13130, 0x07ba },
04071 { 13138, 0x0000 }, { 13138, 0x0000 }, { 13138, 0x0000 }, { 13138, 0x0000 },
04072 { 13138, 0x0000 }, { 13138, 0x0000 }, { 13138, 0x26e0 }, { 13144, 0xbefe },
04073 /* 0x9d00 */
04074 { 13157, 0x03f9 }, { 13165, 0xebb5 }, { 13176, 0xe36d }, { 13186, 0xe9cb },
04075 { 13196, 0x9c2f }, { 13205, 0xbfde }, { 13218, 0x9f83 }, { 13227, 0xabbf },
04076 { 13239, 0x1ff7 }, { 13251, 0xffd5 }, { 13264, 0xb7df }, { 13277, 0xdffe },
04077 { 13291, 0xfdae }, { 13303, 0xffef }, { 13318, 0xfb7e }, { 13331, 0xfffd },
04078 /* 0x9e00 */
04079 { 13345, 0xaaaf }, { 13357, 0x6ebf }, { 13369, 0x0000 }, { 13369, 0x0000 },
04080 { 13369, 0x0000 }, { 13369, 0x0000 }, { 13369, 0x0000 }, { 13369, 0xb620 },
04081 { 13375, 0x7fcd }, { 13387, 0xbe9e }, { 13398, 0x62b3 }, { 13406, 0x58f1 },
04082 { 13414, 0xf10d }, { 13422, 0xfd7b }, { 13435, 0xe9f1 }, { 13445, 0xbefd },
04083 /* 0x9f00 */
04084 { 13458, 0xc6c3 }, { 13466, 0x5f6d }, { 13477, 0xff3d }, { 13490, 0x69ff },
04085 { 13502, 0xffcf }, { 13516, 0xfbf4 }, { 13528, 0xdcfb }, { 13540, 0x4ff7 },
04086 { 13552, 0x2000 }, { 13553, 0x1137 }, { 13560, 0x0015 },
04087 };
04088 static const Summary16 big5_uni2indx_pagefa[1] = {
04089 /* 0xfa00 */
04090 { 13563, 0x3000 },
04091 };
04092 static const Summary16 big5_uni2indx_pagefe[23] = {
04093 /* 0xfe00 */
04094 { 13565, 0x0000 }, { 13565, 0x0000 }, { 13565, 0x0000 }, { 13565, 0xfffb },
04095 { 13580, 0xfelf }, { 13592, 0xfef5 }, { 13605, 0x0e7f }, { 13615, 0x0000 },
04096 { 13615, 0x0000 }, { 13615, 0x0000 }, { 13615, 0x0000 }, { 13615, 0x0000 },
04097 { 13615, 0x0000 }, { 13615, 0x0000 }, { 13615, 0x0000 }, { 13615, 0x0000 },
04098 /* 0xff00 */
04099 { 13615, 0xff7a }, { 13628, 0xffff }, { 13644, 0xffff }, { 13660, 0x97ff },
04100 { 13673, 0xfffe }, { 13688, 0x3fff }, { 13702, 0x0010 },
04101 };
04102
04103 static int
04104 big5_wctomb (conv_t conv, unsigned char *, ucs4_t wc, int n)
04105 {
04106     (void)conv;
04107     if (n >= 2) {
04108         const Summary16 *summary = NULL;
04109         if (wc < 0x0100)
04110             summary = &big5_uni2indx_page00[(wc>>4)];
04111         else if (wc >= 0x0200 && wc < 0x0460)
04112             summary = &big5_uni2indx_page02[(wc>>4)-0x020];
04113         else if (wc >= 0x2000 && wc < 0x22c0)
04114             summary = &big5_uni2indx_page20[(wc>>4)-0x200];
04115         else if (wc >= 0x2400 && wc < 0x2650)

```

```

04116     summary = &big5_uni2indx_page24[(wc>>4)-0x240];
04117     else if (wc >= 0x3000 && wc < 0x33e0)
04118         summary = &big5_uni2indx_page30[(wc>>4)-0x300];
04119     else if (wc >= 0x4e00 && wc < 0x9fb0)
04120         summary = &big5_uni2indx_page4e[(wc>>4)-0x4e0];
04121     else if (wc >= 0xfa00 && wc < 0xfa10)
04122         summary = &big5_uni2indx_pagefa[(wc>>4)-0xfa0];
04123     else if (wc >= 0xfe00 && wc < 0xff70)
04124         summary = &big5_uni2indx_pagefe[(wc>>4)-0xfe0];
04125     if (summary) {
04126         unsigned short used = summary->used;
04127         unsigned int i = wc & 0x0f;
04128         if (used & ((unsigned short) 1 << i)) {
04129             unsigned short c;
04130             /* Keep in 'used' only the bits 0..i-1. */
04131             used &= ((unsigned short) 1 << i) - 1;
04132             /* Add 'summary->indx' and the number of bits set in 'used'. */
04133             used = (used & 0x5555) + ((used & 0xaaaa) >> 1);
04134             used = (used & 0x3333) + ((used & 0xcccc) >> 2);
04135             used = (used & 0x0f0f) + ((used & 0xf0f0) >> 4);
04136             used = (used & 0x00ff) + (used >> 8);
04137             c = big5_2charset[summary->indx + used];
04138             r[0] = (c >> 8); r[1] = (c & 0xff);
04139             return 2;
04140         }
04141     }
04142     return RET_ILSEQ;
04143 }
04144 return RET_TOOSMALL;
04145 }
04146 #endif /* NEED_TOMB */

```

12.260 big5_emacs.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/big5_emacs.h,v 1.1 2000/11/28 18:50:06 dawes Exp $ */
00002
00003 /*
00004  * BIG5-0 and BIG5-1
00005  */
00006
00007 /*
00008  BIG5 with its 13494 characters doesn't fit in a single 94x94 or 96x96
00009  block. Therefore Emacs/Mule developers, in a typically Japanese way of
00010  thinking, have developed an alternative encoding of BIG5 in two 94x94
00011  planes, very similar to the SHIFT_JIS encoding for JISX0208.
00012
00013  Conversion between BIG5 codes (s1,s2) and BIG5-0 codes (c1,c2):
00014  Example. (s1,s2) = 0xA140, (c1,c2) = 0x2121.
00015  0xA1 <= s1 <= 0xC7, 0x40 <= s2 <= 0x7E || 0xA1 <= s2 <= 0xFE,
00016  0x21 <= c1 <= 0x62, 0x21 <= c2 <= 0x7E.
00017  Invariant:
00018      157*(s1-0xA1) + (s2 < 0x80 ? s2-0x40 : s2-0x62)
00019      = 94*(c1-0x21)+(c2-0x21)
00020  Conversion (s1,s2) -> (c1,c2):
00021      t := 157*(s1-0xA1) + (s2 < 0x80 ? s2-0x40 : s2-0x62)
00022      c1 := (t div 94) + 0x21
00023      c2 := (t mod 94) + 0x21
00024  Conversion (c1,c2) -> (s1,s2):
00025      t := 94*(c1-0x21)+(c2-0x21)
00026      t2 := t mod 157
00027      s1 := (t div 157) + 0xA1
00028      s2 := (t2 < 0x3F ? t2+0x40 : t2+0x62)
00029
00030  Conversion between BIG5 codes (s1,s2) and BIG5-1 codes (c1,c2):
00031  Example. (s1,s2) = 0xC940, (c1,c2) = 0x2121.
00032  0xC9 <= s1 <= 0xF9, 0x40 <= s2 <= 0x7E || 0xA1 <= s2 <= 0xFE,
00033  0x21 <= c1 <= 0x72, 0x21 <= c2 <= 0x7E.
00034  Invariant:
00035      157*(s1-0xC9) + (s2 < 0x80 ? s2-0x40 : s2-0x62)
00036      = 94*(c1-0x21)+(c2-0x21)
00037  Conversion (s1,s2) -> (c1,c2):
00038      t := 157*(s1-0xC9) + (s2 < 0x80 ? s2-0x40 : s2-0x62)
00039      c1 := (t div 94) + 0x21
00040      c2 := (t mod 94) + 0x21
00041  Conversion (c1,c2) -> (s1,s2):
00042      t := 94*(c1-0x21)+(c2-0x21)
00043      t2 := t mod 157
00044      s1 := (t div 157) + 0xC9
00045      s2 := (t2 < 0x3F ? t2+0x40 : t2+0x62)
00046  */
00047
00048 static int
00049 big5_0_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00050 {
00051     unsigned char c1 = s[0];

```



```

00052     if (c1 >= 0x21 && c1 <= 0x62) {
00053         if (n >= 2) {
00054             unsigned char c2 = s[1];
00055             if (c2 >= 0x21 && c2 <= 0x7e) {
00056                 unsigned int i = 94 * (c1 - 0x21) + (c2 - 0x21);
00057                 if (0) {
00058                     /* Unoptimized. */
00059                     unsigned char buf[2];
00060                     buf[0] = (i / 157) + 0xa1;
00061                     i = i % 157;
00062                     buf[1] = i + (i < 0x3f ? 0x40 : 0x62);
00063                     return big5_mbtowc(conv,pwc,buf,2);
00064                 } else {
00065                     /* Inline the implementation of big5_mbtowc. */
00066                     if (i < 6121) {
00067                         unsigned short wc = big5_2uni_pagea1[i];
00068                         if (wc != 0xffffd) {
00069                             *pwc = (ucs4_t) wc;
00070                             return 2;
00071                         }
00072                     }
00073                 }
00074             }
00075             return RET_ILSEQ;
00076         }
00077         return RET_TOOFEW(0);
00078     }
00079     return RET_ILSEQ;
00080 }
00081
00082 static int
00083 big5_l_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00084 {
00085     unsigned char c1 = s[0];
00086     if (c1 >= 0x21 && c1 <= 0x72) {
00087         if (n >= 2) {
00088             unsigned char c2 = s[1];
00089             if (c2 >= 0x21 && c2 <= 0x7e) {
00090                 unsigned int i = 94 * (c1 - 0x21) + (c2 - 0x21);
00091                 if (0) {
00092                     /* Unoptimized. */
00093                     unsigned char buf[2];
00094                     buf[0] = (i / 157) + 0xc9;
00095                     i = i % 157;
00096                     buf[1] = i + (i < 0x3f ? 0x40 : 0x62);
00097                     return big5_mbtowc(conv,pwc,buf,2);
00098                 } else {
00099                     /* Inline the implementation of big5_mbtowc. */
00100                     if (i < 7652) {
00101                         unsigned short wc = big5_2uni_pagec9[i];
00102                         if (wc != 0xffffd) {
00103                             *pwc = (ucs4_t) wc;
00104                             return 2;
00105                         }
00106                     }
00107                 }
00108             }
00109             return RET_ILSEQ;
00110         }
00111         return RET_TOOFEW(0);
00112     }
00113     return RET_ILSEQ;
00114 }
00115
00116 static int
00117 big5_0_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00118 {
00119     if (n >= 2) {
00120         unsigned char buf[2];
00121         int ret = big5_wctomb(conv,buf,wc,2);
00122         if (ret != RET_ILSEQ) {
00123             unsigned char s1, s2;
00124             if (ret != 2) abort();
00125             s1 = buf[0];
00126             s2 = buf[1];
00127             if (!(s1 >= 0xa1)) abort();
00128             if (!(s2 >= 0x40 && s2 <= 0x7e) || (s2 >= 0xa1 && s2 <= 0xfe)) abort();
00129             if (s1 < 0xc9) {
00130                 unsigned int t = 157 * (s1 - 0xa1) + s2 - (s2 < 0x80 ? 0x40 : 0x62);
00131                 r[0] = (t / 94) + 0x21;
00132                 r[1] = (t % 94) + 0x21;
00133                 return 2;
00134             }
00135         }
00136         return RET_ILSEQ;
00137     }
00138     return RET_TOOSMALL;

```

```

00139 }
00140
00141 static int
00142 big5_1_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00143 {
00144     if (n >= 2) {
00145         unsigned char buf[2];
00146         int ret = big5_wctomb(conv,buf,wc,2);
00147         if (ret != RET_ILSEQ) {
00148             unsigned char s1, s2;
00149             if (ret != 2) abort();
00150             s1 = buf[0];
00151             s2 = buf[1];
00152             if (!(s1 <= 0xf9)) abort();
00153             if (!((s2 >= 0x40 && s2 <= 0x7e) || (s2 >= 0xa1 && s2 <= 0xfe))) abort();
00154             if (s1 >= 0xc9) {
00155                 unsigned int t = 157 * (s1 - 0xc9) + s2 - (s2 < 0x80 ? 0x40 : 0x62);
00156                 r[0] = (t / 94) + 0x21;
00157                 r[1] = (t % 94) + 0x21;
00158                 return 2;
00159             }
00160         }
00161         return RET_ILSEQ;
00162     }
00163     return RET_TOOSMALL;
00164 }

```

12.261 cp1133.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/cp1133.h,v 1.3 2000/11/29 17:40:28 dawes Exp $ */
00002
00003 /*
00004  * IBM-CP1133
00005  */
00006
00007 static const unsigned short cp1133_2uni_1[64] = {
00008     /* 0xa0 */
00009     0x00a0, 0x0e81, 0x0e82, 0x0e84, 0x0e87, 0x0e88, 0x0eaa, 0x0e8a,
00010     0x0e8d, 0x0e94, 0x0e95, 0x0e96, 0x0e97, 0x0e99, 0x0e9a, 0x0e9b,
00011     /* 0xb0 */
00012     0x0e9c, 0x0e9d, 0x0e9e, 0x0e9f, 0x0ea1, 0x0ea2, 0x0ea3, 0x0ea5,
00013     0x0ea7, 0x0eab, 0x0ead, 0x0eae, 0xfffd, 0xfffd, 0xfffd, 0xeaf,
00014     /* 0xc0 */
00015     0x0eb0, 0x0eb2, 0x0eb3, 0x0eb4, 0x0eb5, 0x0eb6, 0x0eb7, 0x0eb8,
00016     0x0eb9, 0x0ebc, 0xeb1, 0xebb, 0xebd, 0xfffd, 0xfffd, 0xfffd,
00017     /* 0xd0 */
00018     0x0ec0, 0x0ec1, 0x0ec2, 0x0ec3, 0x0ec4, 0x0ec8, 0x0ec9, 0x0eca,
00019     0x0ecb, 0x0ecc, 0x0ecd, 0x0ec6, 0xfffd, 0x0edc, 0x0edd, 0x20ad,
00020 };
00021 static const unsigned short cp1133_2uni_2[16] = {
00022     /* 0xf0 */
00023     0x0ed0, 0x0ed1, 0x0ed2, 0x0ed3, 0x0ed4, 0x0ed5, 0x0ed6, 0x0ed7,
00024     0x0ed8, 0x0ed9, 0xfffd, 0xfffd, 0x00a2, 0x00ac, 0x00a6, 0xfffd,
00025 };
00026
00027 static int
00028 cp1133_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00029 {
00030     unsigned char c = *s;
00031     if (c < 0xa0) {
00032         *pwc = (ucs4_t) c;
00033         return 1;
00034     }
00035     else if (c < 0xe0) {
00036         unsigned short wc = cp1133_2uni_1[c-0xa0];
00037         if (wc != 0xfffd) {
00038             *pwc = (ucs4_t) wc;
00039             return 1;
00040         }
00041     }
00042     else if (c < 0xf0) {
00043     }
00044     else {
00045         unsigned short wc = cp1133_2uni_2[c-0xf0];
00046         if (wc != 0xfffd) {
00047             *pwc = (ucs4_t) wc;
00048             return 1;
00049         }
00050     }
00051     return RET_ILSEQ;
00052 }
00053
00054 static const unsigned char cp1133_page00[16] = {
00055     0xa0, 0x00, 0xfc, 0x00, 0x00, 0x00, 0xfe, 0x00, /* 0xa0-0xa7 */
00056     0x00, 0x00, 0x00, 0x00, 0xfd, 0x00, 0x00, 0x00, /* 0xa8-0xaf */

```

```

00057 };
00058 static const unsigned char cp1133_page0e[96] = {
00059     0x00, 0xa1, 0xa2, 0x00, 0xa3, 0x00, 0x00, 0xa4, /* 0x80-0x87 */
00060     0xa5, 0x00, 0xa7, 0x00, 0x00, 0xa8, 0x00, 0x00, /* 0x88-0x8f */
00061     0x00, 0x00, 0x00, 0x00, 0xa9, 0xaa, 0xab, 0xac, /* 0x90-0x97 */
00062     0x00, 0xad, 0xae, 0xaf, 0xb0, 0xb1, 0xb2, 0xb3, /* 0x98-0x9f */
00063     0x00, 0xb4, 0xb5, 0xb6, 0x00, 0xb7, 0x00, 0xb8, /* 0xa0-0xa7 */
00064     0x00, 0x00, 0xa6, 0xb9, 0x00, 0xba, 0xbb, 0xbf, /* 0xa8-0xaf */
00065     0xc0, 0xca, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, /* 0xb0-0xb7 */
00066     0xc7, 0xc8, 0x00, 0xcb, 0xc9, 0xcc, 0x00, 0x00, /* 0xb8-0xbf */
00067     0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0x00, 0xdb, 0x00, /* 0xc0-0xc7 */
00068     0xd5, 0xd6, 0xd7, 0xd8, 0xd9, 0xda, 0x00, 0x00, /* 0xc8-0xcf */
00069     0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0xd0-0xd7 */
00070     0xf8, 0xf9, 0x00, 0x00, 0xdd, 0xde, 0x00, 0x00, /* 0xd8-0xdf */
00071 };
00072
00073 static int
00074 cp1133_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00075 {
00076     unsigned char c = 0;
00077     if (wc < 0x00a0) {
00078         *r = wc;
00079         return 1;
00080     }
00081     else if (wc >= 0x00a0 && wc < 0x00b0)
00082         c = cp1133_page00[wc-0x00a0];
00083     else if (wc >= 0x00e80 && wc < 0x00ee0)
00084         c = cp1133_page0e[wc-0x00e80];
00085     else if (wc == 0x20ad)
00086         c = 0xdf;
00087     if (c != 0) {
00088         *r = c;
00089         return 1;
00090     }
00091     return RET_ILSEQ;
00092 }

```

12.262 cp1251.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/cp1251.h,v 1.1 2000/12/04 18:49:32 dawes Exp $ */
00002
00003 /*
00004  * CP1251
00005  */
00006 #ifdef NEED_TOWC
00007
00008 static const unsigned short cp1251_2uni[128] = {
00009     /* 0x80 */
00010     0x0402, 0x0403, 0x201a, 0x0453, 0x201e, 0x2026, 0x2020, 0x2021,
00011     0x20ac, 0x2030, 0x0409, 0x2039, 0x040a, 0x040c, 0x040b, 0x040f,
00012     /* 0x90 */
00013     0x0452, 0x2018, 0x2019, 0x201c, 0x201d, 0x2022, 0x2013, 0x2014,
00014     0xffff, 0x2122, 0x0459, 0x203a, 0x045a, 0x045c, 0x045b, 0x045f,
00015     /* 0xa0 */
00016     0x00a0, 0x040e, 0x045e, 0x0408, 0x00a4, 0x0490, 0x00a6, 0x00a7,
00017     0x0401, 0x00a9, 0x0404, 0x00ab, 0x00ac, 0x00ad, 0x00ae, 0x0407,
00018     /* 0xb0 */
00019     0x00b0, 0x00b1, 0x0406, 0x0456, 0x0491, 0x00b5, 0x00b6, 0x00b7,
00020     0x0451, 0x2116, 0x0454, 0x00bb, 0x0458, 0x0405, 0x0455, 0x0457,
00021     /* 0xc0 */
00022     0x0410, 0x0411, 0x0412, 0x0413, 0x0414, 0x0415, 0x0416, 0x0417,
00023     0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e, 0x041f,
00024     /* 0xd0 */
00025     0x0420, 0x0421, 0x0422, 0x0423, 0x0424, 0x0425, 0x0426, 0x0427,
00026     0x0428, 0x0429, 0x042a, 0x042b, 0x042c, 0x042d, 0x042e, 0x042f,
00027     /* 0xe0 */
00028     0x0430, 0x0431, 0x0432, 0x0433, 0x0434, 0x0435, 0x0436, 0x0437,
00029     0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e, 0x043f,
00030     /* 0xf0 */
00031     0x0440, 0x0441, 0x0442, 0x0443, 0x0444, 0x0445, 0x0446, 0x0447,
00032     0x0448, 0x0449, 0x044a, 0x044b, 0x044c, 0x044d, 0x044e, 0x044f,
00033 };
00034
00035 static int
00036 cp1251_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00037 {
00038     unsigned char c = *s;
00039     if (c < 0x80) {
00040         *pwc = (ucs4_t) c;
00041         return 1;
00042     }
00043     else {
00044         unsigned short wc = cp1251_2uni[c-0x80];
00045         if (wc != 0xffff) {
00046             *pwc = (ucs4_t) wc;

```

```

00047     return 1;
00048 }
00049 }
00050 return RET_ILSEQ;
00051 }
00052 #endif /* NEED_TOWC */
00053
00054 #ifdef NEED_TOMB
00055 static const unsigned char cp1251_page00[32] = {
00056     0xa0, 0x00, 0x00, 0x00, 0xa4, 0x00, 0xa6, 0xa7, /* 0xa0-0xa7 */
00057     0x00, 0xa9, 0x00, 0xab, 0xac, 0xad, 0xae, 0x00, /* 0xa8-0xaf */
00058     0xb0, 0xb1, 0x00, 0x00, 0x00, 0xb5, 0xb6, 0xb7, /* 0xb0-0xb7 */
00059     0x00, 0x00, 0x00, 0xbb, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00060 };
00061 static const unsigned char cp1251_page04[152] = {
00062     0x00, 0xa8, 0x80, 0x81, 0xaa, 0xbd, 0xb2, 0xaf, /* 0x00-0x07 */
00063     0xa3, 0x8a, 0x8c, 0x8e, 0x8d, 0x00, 0xa1, 0x8f, /* 0x08-0x0f */
00064     0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0x10-0x17 */
00065     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0x18-0x1f */
00066     0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, /* 0x20-0x27 */
00067     0xd8, 0xd9, 0xda, 0xdb, 0xdc, 0xdd, 0xde, 0xdf, /* 0x28-0x2f */
00068     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0x30-0x37 */
00069     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0x38-0x3f */
00070     0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0x40-0x47 */
00071     0xf8, 0xf9, 0xfa, 0xfb, 0xfc, 0xfd, 0xfe, 0xff, /* 0x48-0x4f */
00072     0x00, 0xb8, 0x90, 0x83, 0xba, 0xbe, 0xb3, 0xbf, /* 0x50-0x57 */
00073     0xbc, 0x9a, 0x9c, 0x9e, 0x9d, 0x00, 0xa2, 0x9f, /* 0x58-0x5f */
00074     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00075     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00076     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00077     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00078     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00079     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00080     0xa5, 0xb4, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00081 };
00082 static const unsigned char cp1251_page20[48] = {
00083     0x00, 0x00, 0x00, 0x96, 0x97, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00084     0x91, 0x92, 0x82, 0x00, 0x93, 0x94, 0x84, 0x00, /* 0x18-0x1f */
00085     0x86, 0x87, 0x95, 0x00, 0x00, 0x00, 0x85, 0x00, /* 0x20-0x27 */
00086     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00087     0x89, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00088     0x00, 0x8b, 0x9b, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00089 };
00090
00091 static int
00092 cp1251_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00093 {
00094     (void)conv; (void)n;
00095     unsigned char c = 0;
00096     if (wc < 0x0080) {
00097         *r = wc;
00098         return 1;
00099     }
00100     else if (wc >= 0x00a0 && wc < 0x00c0)
00101         c = cp1251_page00[wc-0x00a0];
00102     else if (wc >= 0x0400 && wc < 0x0498)
00103         c = cp1251_page04[wc-0x0400];
00104     else if (wc >= 0x2010 && wc < 0x2040)
00105         c = cp1251_page20[wc-0x2010];
00106     else if (wc == 0x20ac)
00107         c = 0x88;
00108     else if (wc == 0x2116)
00109         c = 0xb9;
00110     else if (wc == 0x2122)
00111         c = 0x99;
00112     if (c != 0) {
00113         *r = c;
00114         return 1;
00115     }
00116     return RET_ILSEQ;
00117 }
00118 #endif /* NEED_TOMB */

```

12.263 cp1255.h

```

00001 /* $XFree86: xc/lib/X11/1cUniConv/cp1255.h,v 1.1 2000/12/04 18:49:33 dawes Exp $ */
00002
00003 /*
00004  * CP1255
00005  */
00006
00007 static const unsigned short cp1255_2uni[128] = {
00008     /* 0x80 */
00009     0x20ac, 0xffffd, 0x201a, 0x0192, 0x201e, 0x2026, 0x2020, 0x2021,
00010     0x02c6, 0x2030, 0xffffd, 0x2039, 0xffffd, 0xffffd, 0xffffd, 0xffffd,

```

```

00011  /* 0x90 */
00012  0xfffd, 0x2018, 0x2019, 0x201c, 0x201d, 0x2022, 0x2013, 0x2014,
00013  0x02dc, 0x2122, 0xffff, 0x203a, 0xffff, 0xffff, 0xffff, 0xffff,
00014  /* 0xa0 */
00015  0x00a0, 0x00a1, 0x00a2, 0x00a3, 0x20aa, 0x00a5, 0x00a6, 0x00a7,
00016  0x00a8, 0x00a9, 0x00d7, 0x00ab, 0x00ac, 0x00ad, 0x00ae, 0x00af,
00017  /* 0xb0 */
00018  0x00b0, 0x00b1, 0x00b2, 0x00b3, 0x00b4, 0x00b5, 0x00b6, 0x00b7,
00019  0x00b8, 0x00b9, 0x00f7, 0x00bb, 0x00bc, 0x00bd, 0x00be, 0x00bf,
00020  /* 0xc0 */
00021  0x05b0, 0x05b1, 0x05b2, 0x05b3, 0x05b4, 0x05b5, 0x05b6, 0x05b7,
00022  0x05b8, 0x05b9, 0xffff, 0x05bb, 0x05bc, 0x05bd, 0x05be, 0x05bf,
00023  /* 0xd0 */
00024  0x05c0, 0x05c1, 0x05c2, 0x05c3, 0x05f0, 0x05f1, 0x05f2, 0x05f3,
00025  0x05f4, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00026  /* 0xe0 */
00027  0x05d0, 0x05d1, 0x05d2, 0x05d3, 0x05d4, 0x05d5, 0x05d6, 0x05d7,
00028  0x05d8, 0x05d9, 0x05da, 0x05db, 0x05dc, 0x05dd, 0x05de, 0x05df,
00029  /* 0xf0 */
00030  0x05e0, 0x05e1, 0x05e2, 0x05e3, 0x05e4, 0x05e5, 0x05e6, 0x05e7,
00031  0x05e8, 0x05e9, 0x05ea, 0xffff, 0xffff, 0x200e, 0x200f, 0xffff,
00032  };
00033
00034  static int
00035  cp1255_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00036  {
00037      unsigned char c = *s;
00038      if (c < 0x80) {
00039          *pwc = (ucs4_t) c;
00040          return 1;
00041      }
00042      else {
00043          unsigned short wc = cp1255_2uni[c-0x80];
00044          if (wc != 0xffff) {
00045              *pwc = (ucs4_t) wc;
00046              return 1;
00047          }
00048      }
00049      return RET_ILSEQ;
00050  }
00051
00052  static const unsigned char cp1255_page00[88] = {
00053      0xa0, 0xa1, 0xa2, 0xa3, 0x00, 0xa5, 0xa6, 0xa7, /* 0xa0-0xa7 */
00054      0xa8, 0xa9, 0x00, 0xab, 0xac, 0xad, 0xae, 0xaf, /* 0xa8-0xaf */
00055      0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5, 0xb6, 0xb7, /* 0xb0-0xb7 */
00056      0xb8, 0xb9, 0x00, 0xbb, 0xbc, 0xbd, 0xbe, 0xbf, /* 0xb8-0xbf */
00057      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00058      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00059      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xaa, /* 0xd0-0xd7 */
00060      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00061      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe0-0xe7 */
00062      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe8-0xef */
00063      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xba, /* 0xf0-0xf7 */
00064  };
00065  static const unsigned char cp1255_page02[32] = {
00066      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x88, 0x00, /* 0xc0-0xc7 */
00067      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00068      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00069      0x00, 0x00, 0x00, 0x00, 0x98, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00070  };
00071  static const unsigned char cp1255_page05[72] = {
00072      0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0xb0-0xb7 */
00073      0xc8, 0xc9, 0x00, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0xb8-0xbf */
00074      0xd0, 0xd1, 0xd2, 0xd3, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00075      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00076      0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0xd0-0xd7 */
00077      0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0xd8-0xdf */
00078      0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0xe0-0xe7 */
00079      0xf8, 0xf9, 0xfa, 0xfb, 0x00, 0x00, 0x00, 0x00, /* 0xe8-0xef */
00080      0xd4, 0xd5, 0xd6, 0xd7, 0xd8, 0x00, 0x00, 0x00, /* 0xf0-0xf7 */
00081  };
00082  static const unsigned char cp1255_page20[56] = {
00083      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfd, 0xfe, /* 0x08-0x0f */
00084      0x00, 0x00, 0x00, 0x96, 0x97, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00085      0x91, 0x92, 0x82, 0x00, 0x93, 0x94, 0x84, 0x00, /* 0x18-0x1f */
00086      0x86, 0x87, 0x95, 0x00, 0x00, 0x00, 0x85, 0x00, /* 0x20-0x27 */
00087      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00088      0x89, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00089      0x00, 0x8b, 0x9b, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00090  };
00091
00092  static int
00093  cp1255_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00094  {
00095      unsigned char c = 0;
00096      if (wc < 0x0080) {
00097          *r = wc;

```

```

00098     return 1;
00099 }
00100 else if (wc >= 0x00a0 && wc < 0x00f8)
00101     c = cp1255_page00[wc-0x00a0];
00102 else if (wc == 0x0192)
00103     c = 0x83;
00104 else if (wc >= 0x02c0 && wc < 0x02e0)
00105     c = cp1255_page02[wc-0x02c0];
00106 else if (wc >= 0x05b0 && wc < 0x05f8)
00107     c = cp1255_page05[wc-0x05b0];
00108 else if (wc >= 0x2008 && wc < 0x2040)
00109     c = cp1255_page20[wc-0x2008];
00110 else if (wc == 0x20aa)
00111     c = 0xa4;
00112 else if (wc == 0x20ac)
00113     c = 0x80;
00114 else if (wc == 0x2122)
00115     c = 0x99;
00116 if (c != 0) {
00117     *r = c;
00118     return 1;
00119 }
00120 return RET_ILSEQ;
00121 }

```

12.264 cp1256.h

```

00001 /* $XFree86: xc/lib/X11/1cUniConv/cp1256.h,v 1.1 2000/12/04 18:49:34 dawes Exp $ */
00002
00003 /*
00004  * CP1256
00005  */
00006
00007 static const unsigned short cp1256_2uni[128] = {
00008     /* 0x80 */
00009     0x20ac, 0x067e, 0x201a, 0x0192, 0x201e, 0x2026, 0x2020, 0x2021,
00010     0x02c6, 0x2030, 0x0679, 0x2039, 0x0152, 0x0686, 0x0698, 0x0688,
00011     /* 0x90 */
00012     0x06af, 0x2018, 0x2019, 0x201c, 0x201d, 0x2022, 0x2013, 0x2014,
00013     0x06a9, 0x2122, 0x0691, 0x203a, 0x0153, 0x200c, 0x200d, 0x06ba,
00014     /* 0xa0 */
00015     0x00a0, 0x060c, 0x00a2, 0x00a3, 0x00a4, 0x00a5, 0x00a6, 0x00a7,
00016     0x00a8, 0x00a9, 0x06be, 0x00ab, 0x00ac, 0x00ad, 0x00ae, 0x00af,
00017     /* 0xb0 */
00018     0x00b0, 0x00b1, 0x00b2, 0x00b3, 0x00b4, 0x00b5, 0x00b6, 0x00b7,
00019     0x00b8, 0x00b9, 0x061b, 0x00bb, 0x00bc, 0x00bd, 0x00be, 0x061f,
00020     /* 0xc0 */
00021     0x06c1, 0x0621, 0x0622, 0x0623, 0x0624, 0x0625, 0x0626, 0x0627,
00022     0x0628, 0x0629, 0x062a, 0x062b, 0x062c, 0x062d, 0x062e, 0x062f,
00023     /* 0xd0 */
00024     0x0630, 0x0631, 0x0632, 0x0633, 0x0634, 0x0635, 0x0636, 0x00d7,
00025     0x0637, 0x0638, 0x0639, 0x063a, 0x0640, 0x0641, 0x0642, 0x0643,
00026     /* 0xe0 */
00027     0x00e0, 0x0644, 0x00e2, 0x0645, 0x0646, 0x0647, 0x0648, 0x00e7,
00028     0x00e8, 0x00e9, 0x00ea, 0x00eb, 0x0649, 0x064a, 0x00ee, 0x00ef,
00029     /* 0xf0 */
00030     0x064b, 0x064c, 0x064d, 0x064e, 0x00f4, 0x064f, 0x0650, 0x00f7,
00031     0x0651, 0x00f9, 0x0652, 0x00fb, 0x00fc, 0x200e, 0x200f, 0x06d2,
00032 };
00033
00034 static int
00035 cp1256_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00036 {
00037     unsigned char c = *s;
00038     if (c < 0x80)
00039         *pwc = (ucs4_t) c;
00040     else
00041         *pwc = (ucs4_t) cp1256_2uni[c-0x80];
00042     return 1;
00043 }
00044
00045 static const unsigned char cp1256_page00[96] = {
00046     0xa0, 0x00, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, /* 0xa0-0xa7 */
00047     0xa8, 0xa9, 0x00, 0xab, 0xac, 0xad, 0xae, 0xaf, /* 0xa8-0xaf */
00048     0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5, 0xb6, 0xb7, /* 0xb0-0xb7 */
00049     0xb8, 0xb9, 0x00, 0xbb, 0xbc, 0xbd, 0xbe, 0x00, /* 0xb8-0xbf */
00050     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00051     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00052     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xd7, /* 0xd0-0xd7 */
00053     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00054     0xe0, 0x00, 0xe2, 0x00, 0x00, 0x00, 0x00, 0xe7, /* 0xe0-0xe7 */
00055     0xe8, 0xe9, 0xea, 0xeb, 0x00, 0xee, 0xef, /* 0xe8-0xef */
00056     0x00, 0x00, 0x00, 0x00, 0xf4, 0x00, 0x00, 0xf7, /* 0xf0-0xf7 */
00057     0x00, 0xf9, 0x00, 0xfb, 0xfc, 0x00, 0x00, 0x00, /* 0xf8-0xff */
00058 };

```

```

00059 static const unsigned char cp1256_page01[72] = {
00060     0x00, 0x00, 0x8c, 0x9c, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00061     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00062     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00063     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00064     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00065     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00066     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00067     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00068     0x00, 0x00, 0x83, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00069 };
00070 static const unsigned char cp1256_page06[208] = {
00071     0x00, 0x00, 0x00, 0x00, 0xa1, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00072     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00073     0x00, 0x00, 0x00, 0x00, 0xba, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00074     0x00, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0x20-0x27 */
00075     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0x28-0x2f */
00076     0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd8, /* 0x30-0x37 */
00077     0xd9, 0xda, 0xdb, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00078     0xdc, 0xdd, 0xde, 0xdf, 0xe1, 0xe3, 0xe4, 0xe5, /* 0x40-0x47 */
00079     0xe6, 0xec, 0xed, 0xf0, 0xf1, 0xf2, 0xf3, 0xf5, /* 0x48-0x4f */
00080     0xf6, 0xf8, 0xfa, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00081     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00082     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00083     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00084     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00085     0x00, 0x8a, 0x00, 0x00, 0x00, 0x00, 0x00, 0x81, /* 0x78-0x7f */
00086     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x8d, 0x00, /* 0x80-0x87 */
00087     0x8f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00088     0x00, 0x9a, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00089     0x8e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x98-0x9f */
00090     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00091     0x00, 0x98, 0x00, 0x00, 0x00, 0x00, 0x00, 0x90, /* 0xa8-0xaf */
00092     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb0-0xb7 */
00093     0x00, 0x00, 0x9f, 0x00, 0x00, 0x00, 0xaa, 0x00, /* 0xb8-0xbf */
00094     0x00, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00095     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00096     0x00, 0x00, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00097 };
00098 static const unsigned char cp1256_page20[56] = {
00099     0x00, 0x00, 0x00, 0x00, 0x9d, 0x9e, 0xfd, 0xfe, /* 0x08-0x0f */
00100     0x00, 0x00, 0x00, 0x00, 0x96, 0x97, 0x00, 0x00, /* 0x10-0x17 */
00101     0x91, 0x92, 0x82, 0x00, 0x93, 0x94, 0x84, 0x00, /* 0x18-0x1f */
00102     0x86, 0x87, 0x95, 0x00, 0x00, 0x00, 0x85, 0x00, /* 0x20-0x27 */
00103     0x00, 0x98, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00104     0x89, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00105     0x00, 0x8b, 0x9b, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00106 };
00107
00108 static int
00109 cp1256_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00110 {
00111     unsigned char c = 0;
00112     if (wc < 0x0080) {
00113         *r = wc;
00114         return 1;
00115     }
00116     else if (wc >= 0x00a0 && wc < 0x0100)
00117         c = cp1256_page00[wc-0x00a0];
00118     else if (wc >= 0x0150 && wc < 0x0198)
00119         c = cp1256_page01[wc-0x0150];
00120     else if (wc == 0x02c6)
00121         c = 0x88;
00122     else if (wc >= 0x0608 && wc < 0x06d8)
00123         c = cp1256_page06[wc-0x0608];
00124     else if (wc >= 0x2008 && wc < 0x2040)
00125         c = cp1256_page20[wc-0x2008];
00126     else if (wc == 0x20ac)
00127         c = 0x80;
00128     else if (wc == 0x2122)
00129         c = 0x99;
00130     if (c != 0) {
00131         *r = c;
00132         return 1;
00133     }
00134     return RET_ILSEQ;
00135 }

```

12.265 cp936ext.h

```

00001 /*
00002  * Character encoding support for the Fast Light Tool Kit (FLTK).
00003  *
00004  * Copyright 1998-2025 by Bill Spitzak and others.
00005  *

```

```
00006  * This library is free software. Distribution and use rights are outlined in
00007  * the file "COPYING" which should have been included with this file.  If this
00008  * file is missing or damaged, see the license at:
00009  *
00010  *   https://www.fltk.org/COPYING.php
00011  *
00012  * Please see the following page on how to report bugs and issues:
00013  *
00014  *   https://www.fltk.org/bugs.php
00015  */
00016
00017 #ifndef CP936
00018 #ifdef NEED_TOWC
00019 static int
00020 cp936ext_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00021 {
00022     return 0;
00023 }
00024 #endif /* NEED_TOWC */
00025
00026 #ifdef NEED_TOMB
00027 static int
00028 cp936ext_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00029 {
00030     (void)conv; (void)r; (void)wc; (void)n;
00031     return 0;
00032 }
00033 #endif /* NEED_TOMB */
00034
00035 #else /* CP936 */
00036 /*
00037  * CP936EXT
00038  */
00039 #ifdef NEED_TOWC
00040
00041 static const unsigned short cp936ext_2uni_page81[23766] = {
00042     /* 0x81 */
00043     0x4e02, 0x4e04, 0x4e05, 0x4e06, 0x4e0f, 0x4e12, 0x4e17, 0x4e1f,
00044     0x4e20, 0x4e21, 0x4e23, 0x4e26, 0x4e29, 0x4e2e, 0x4e2f, 0x4e31,
00045     0x4e33, 0x4e35, 0x4e37, 0x4e3c, 0x4e40, 0x4e41, 0x4e42, 0x4e44,
00046     0x4e46, 0x4e4a, 0x4e51, 0x4e55, 0x4e57, 0x4e5a, 0x4e5b, 0x4e62,
00047     0x4e63, 0x4e64, 0x4e65, 0x4e67, 0x4e68, 0x4e6a, 0x4e6b, 0x4e6c,
00048     0x4e6d, 0x4e6e, 0x4e6f, 0x4e72, 0x4e74, 0x4e75, 0x4e76, 0x4e77,
00049     0x4e78, 0x4e79, 0x4e7a, 0x4e7b, 0x4e7c, 0x4e7d, 0x4e7f, 0x4e80,
00050     0x4e81, 0x4e82, 0x4e83, 0x4e84, 0x4e85, 0x4e87, 0x4e8a, 0x4e90,
00051     0x4e96, 0x4e97, 0x4e99, 0x4e9c, 0x4e9d, 0x4e9e, 0x4ea3, 0x4eaa,
00052     0x4eaf, 0x4eb0, 0x4eb1, 0x4eb4, 0x4eb6, 0x4eb7, 0x4eb8, 0x4eb9,
00053     0x4ebc, 0x4ebd, 0x4ebe, 0x4ec8, 0x4ecc, 0x4ecf, 0x4ed0, 0x4ed2,
00054     0x4eda, 0x4edb, 0x4edc, 0x4ee0, 0x4ee2, 0x4ee6, 0x4ee7, 0x4ee9,
00055     0x4eed, 0x4eee, 0x4eef, 0x4ef1, 0x4ef4, 0x4ef8, 0x4ef9, 0x4efa,
00056     0x4efc, 0x4efc, 0x4efc, 0x4f00, 0x4f02, 0x4f03, 0x4f04, 0x4f05, 0x4f06,
00057     0x4f07, 0x4f08, 0x4f0b, 0x4f0c, 0x4f12, 0x4f13, 0x4f14, 0x4f15,
00058     0x4f16, 0x4f1c, 0x4f1d, 0x4f21, 0x4f23, 0x4f28, 0x4f29, 0x4f2c,
00059     0x4f2d, 0x4f2e, 0x4f31, 0x4f33, 0x4f35, 0x4f37, 0x4f39, 0x4f3b,
00060     0x4f3e, 0x4f3f, 0x4f40, 0x4f41, 0x4f42, 0x4f44, 0x4f45, 0x4f47,
00061     0x4f48, 0x4f49, 0x4f4a, 0x4f4b, 0x4f4c, 0x4f52, 0x4f54, 0x4f56,
00062     0x4f61, 0x4f62, 0x4f66, 0x4f68, 0x4f6a, 0x4f6b, 0x4f6d, 0x4f6e,
00063     0x4f71, 0x4f72, 0x4f75, 0x4f77, 0x4f78, 0x4f79, 0x4f7a, 0x4f7d,
00064     0x4f80, 0x4f81, 0x4f82, 0x4f85, 0x4f86, 0x4f87, 0x4f8a, 0x4f8c,
00065     0x4f8e, 0x4f90, 0x4f92, 0x4f93, 0x4f95, 0x4f96, 0x4f98, 0x4f99,
00066     0x4f9a, 0x4f9c, 0x4f9e, 0x4f9f, 0x4fa1, 0x4fa2,
00067     /* 0x82 */
00068     0x4fa4, 0x4fab, 0x4fad, 0x4fb0, 0x4fb1, 0x4fb2, 0x4fb3, 0x4fb4,
00069     0x4fb6, 0x4fb7, 0x4fb8, 0x4fb9, 0x4fba, 0x4fbb, 0x4fbc, 0x4fbd,
00070     0x4fbe, 0x4fc0, 0x4fc1, 0x4fc2, 0x4fc6, 0x4fc7, 0x4fc8, 0x4fc9,
00071     0x4fcb, 0x4fcc, 0x4fcd, 0x4fd2, 0x4fd3, 0x4fd4, 0x4fd5, 0x4fd6,
00072     0x4fd9, 0x4fdb, 0x4fe0, 0x4fe2, 0x4fe4, 0x4fe5, 0x4fe7, 0x4feb,
00073     0x4fec, 0x4ff0, 0x4ff2, 0x4ff4, 0x4ff5, 0x4ff6, 0x4ff7, 0x4ff9,
00074     0x4ffb, 0x4ffc, 0x4ffd, 0x4fff, 0x5000, 0x5001, 0x5002, 0x5003,
00075     0x5004, 0x5005, 0x5006, 0x5007, 0x5008, 0x5009, 0x500a, 0x500b,
00076     0x500e, 0x5010, 0x5011, 0x5013, 0x5015, 0x5016, 0x5017, 0x501b,
00077     0x501d, 0x501e, 0x5020, 0x5022, 0x5023, 0x5024, 0x5027, 0x502b,
00078     0x502f, 0x5030, 0x5031, 0x5032, 0x5033, 0x5034, 0x5035, 0x5036,
00079     0x5037, 0x5038, 0x5039, 0x503b, 0x503d, 0x503f, 0x5040, 0x5041,
00080     0x5042, 0x5044, 0x5045, 0x5046, 0x5049, 0x504a, 0x504b, 0x504d,
00081     0x5050, 0x5051, 0x5052, 0x5053, 0x5054, 0x5056, 0x5057, 0x5058,
00082     0x5059, 0x505b, 0x505d, 0x505e, 0x505f, 0x5060, 0x5061, 0x5062,
00083     0x5063, 0x5064, 0x5066, 0x5067, 0x5068, 0x5069, 0x506a, 0x506b,
00084     0x506d, 0x506e, 0x506f, 0x5070, 0x5071, 0x5072, 0x5073, 0x5074,
00085     0x5075, 0x5078, 0x5079, 0x507a, 0x507c, 0x507d, 0x5081, 0x5082,
00086     0x5083, 0x5084, 0x5086, 0x5087, 0x5089, 0x508a, 0x508b, 0x508c,
00087     0x508e, 0x508f, 0x5090, 0x5091, 0x5092, 0x5093, 0x5094, 0x5095,
00088     0x5096, 0x5097, 0x5098, 0x5099, 0x509a, 0x509b, 0x509c, 0x509d,
00089     0x509e, 0x509f, 0x50a0, 0x50a1, 0x50a2, 0x50a4, 0x50a6, 0x50aa,
00090     0x50ab, 0x50ad, 0x50ae, 0x50af, 0x50b0, 0x50b1, 0x50b3, 0x50b4,
00091     0x50b5, 0x50b6, 0x50b7, 0x50b8, 0x50b9, 0x50bc,
00092     /* 0x83 */
```



```
00093 0x50bd, 0x50be, 0x50bf, 0x50c0, 0x50c1, 0x50c2, 0x50c3, 0x50c4,
00094 0x50c5, 0x50c6, 0x50c7, 0x50c8, 0x50c9, 0x50ca, 0x50cb, 0x50cc,
00095 0x50cd, 0x50ce, 0x50cf, 0x50d0, 0x50d1, 0x50d2, 0x50d3, 0x50d4, 0x50d5,
00096 0x50d7, 0x50d8, 0x50d9, 0x50db, 0x50dc, 0x50dd, 0x50de, 0x50df,
00097 0x50e0, 0x50e1, 0x50e2, 0x50e3, 0x50e4, 0x50e5, 0x50e6, 0x50e9,
00098 0x50ea, 0x50eb, 0x50ec, 0x50ef, 0x50f0, 0x50f1, 0x50f2, 0x50f4, 0x50f6,
00099 0x50f7, 0x50f8, 0x50f9, 0x50fa, 0x50fc, 0x50fd, 0x50fe, 0x50ff,
00100 0x5100, 0x5101, 0x5102, 0x5103, 0x5104, 0x5105, 0x5108, 0x5109,
00101 0x510a, 0x510c, 0x510d, 0x510e, 0x510f, 0x5110, 0x5111, 0x5113,
00102 0x5114, 0x5115, 0x5116, 0x5117, 0x5118, 0x5119, 0x511a, 0x511b,
00103 0x511c, 0x511d, 0x511e, 0x511f, 0x5120, 0x5122, 0x5123, 0x5124,
00104 0x5125, 0x5126, 0x5127, 0x5128, 0x5129, 0x512a, 0x512b, 0x512c,
00105 0x512d, 0x512e, 0x512f, 0x5130, 0x5131, 0x5132, 0x5133, 0x5134,
00106 0x5135, 0x5136, 0x5137, 0x5138, 0x5139, 0x513a, 0x513b, 0x513c,
00107 0x513d, 0x513e, 0x513f, 0x5142, 0x5147, 0x514a, 0x514c, 0x514e, 0x514f,
00108 0x5150, 0x5152, 0x5153, 0x5157, 0x5158, 0x5159, 0x515b, 0x515d,
00109 0x515e, 0x515f, 0x5160, 0x5161, 0x5163, 0x5164, 0x5166, 0x5167,
00110 0x5169, 0x516a, 0x516f, 0x5172, 0x517a, 0x517e, 0x517f, 0x5183,
00111 0x5184, 0x5186, 0x5187, 0x518a, 0x518b, 0x518e, 0x518f, 0x5190,
00112 0x5191, 0x5193, 0x5194, 0x5198, 0x519a, 0x519d, 0x519e, 0x519f,
00113 0x51a1, 0x51a3, 0x51a6, 0x51a7, 0x51a8, 0x51a9, 0x51aa, 0x51ad,
00114 0x51ae, 0x51b4, 0x51b8, 0x51b9, 0x51ba, 0x51be, 0x51bf, 0x51c1,
00115 0x51c2, 0x51c3, 0x51c5, 0x51c8, 0x51ca, 0x51cd, 0x51ce, 0x51d0,
00116 0x51d2, 0x51d3, 0x51d4, 0x51d5, 0x51d6, 0x51d7,
00117 /* 0x84 */
00118 0x51d8, 0x51d9, 0x51da, 0x51dc, 0x51de, 0x51df, 0x51e2, 0x51e3,
00119 0x51e5, 0x51e6, 0x51e7, 0x51e8, 0x51e9, 0x51ea, 0x51ec, 0x51ee,
00120 0x51f1, 0x51f2, 0x51f4, 0x51f7, 0x51fe, 0x5204, 0x5205, 0x5209,
00121 0x520b, 0x520c, 0x520f, 0x5210, 0x5213, 0x5214, 0x5215, 0x521c,
00122 0x521e, 0x521f, 0x5221, 0x5222, 0x5223, 0x5225, 0x5226, 0x5227,
00123 0x522a, 0x522c, 0x522f, 0x5231, 0x5232, 0x5234, 0x5235, 0x523c,
00124 0x523e, 0x5244, 0x5245, 0x5246, 0x5247, 0x5248, 0x5249, 0x524b,
00125 0x524e, 0x524f, 0x5252, 0x5253, 0x5255, 0x5257, 0x5258, 0x5259,
00126 0x525a, 0x525b, 0x525d, 0x525f, 0x5260, 0x5262, 0x5263, 0x5264,
00127 0x5266, 0x5268, 0x526b, 0x526c, 0x526d, 0x526e, 0x5270, 0x5271,
00128 0x5273, 0x5274, 0x5275, 0x5276, 0x5277, 0x5278, 0x5279, 0x527a,
00129 0x527b, 0x527c, 0x527e, 0x5280, 0x5283, 0x5284, 0x5285, 0x5286,
00130 0x5287, 0x5289, 0x528a, 0x528b, 0x528c, 0x528d, 0x528e, 0x528f,
00131 0x5291, 0x5292, 0x5294, 0x5295, 0x5296, 0x5297, 0x5298, 0x5299,
00132 0x529a, 0x529c, 0x52a4, 0x52a5, 0x52a6, 0x52a7, 0x52ae, 0x52af,
00133 0x52b0, 0x52b4, 0x52b5, 0x52b6, 0x52b7, 0x52b8, 0x52b9, 0x52ba,
00134 0x52bb, 0x52bc, 0x52bd, 0x52c0, 0x52c1, 0x52c2, 0x52c4, 0x52c5,
00135 0x52c6, 0x52c8, 0x52ca, 0x52cc, 0x52cd, 0x52ce, 0x52cf, 0x52d1,
00136 0x52d3, 0x52d4, 0x52d5, 0x52d7, 0x52d9, 0x52da, 0x52db, 0x52dc,
00137 0x52dd, 0x52de, 0x52e0, 0x52e1, 0x52e2, 0x52e3, 0x52e5, 0x52e6,
00138 0x52e7, 0x52e8, 0x52e9, 0x52ea, 0x52eb, 0x52ec, 0x52ed, 0x52ee,
00139 0x52ef, 0x52f1, 0x52f2, 0x52f3, 0x52f4, 0x52f5, 0x52f6, 0x52f7,
00140 0x52f8, 0x52fb, 0x52fc, 0x52fd, 0x5301, 0x5302, 0x5303, 0x5304,
00141 0x5307, 0x5309, 0x530a, 0x530b, 0x530c, 0x530e,
00142 /* 0x85 */
00143 0x5311, 0x5312, 0x5313, 0x5314, 0x5318, 0x531b, 0x531c, 0x531e,
00144 0x531f, 0x5322, 0x5324, 0x5325, 0x5327, 0x5328, 0x5329, 0x532b,
00145 0x532c, 0x532d, 0x532f, 0x5330, 0x5331, 0x5332, 0x5333, 0x5334,
00146 0x5335, 0x5336, 0x5337, 0x5338, 0x533c, 0x533d, 0x5340, 0x5342,
00147 0x5344, 0x5346, 0x534b, 0x534c, 0x534d, 0x5350, 0x5354, 0x5358,
00148 0x5359, 0x535b, 0x535d, 0x5365, 0x5368, 0x536a, 0x536c, 0x536d,
00149 0x5372, 0x5376, 0x5379, 0x537b, 0x537c, 0x537d, 0x537e, 0x5380,
00150 0x5381, 0x5383, 0x5387, 0x5388, 0x538a, 0x538e, 0x538f, 0x5390,
00151 0x5391, 0x5392, 0x5393, 0x5394, 0x5396, 0x5397, 0x5399, 0x539b,
00152 0x539c, 0x539e, 0x53a0, 0x53a1, 0x53a4, 0x53a7, 0x53aa, 0x53ab,
00153 0x53ac, 0x53ad, 0x53af, 0x53b0, 0x53b1, 0x53b2, 0x53b3, 0x53b4,
00154 0x53b5, 0x53b7, 0x53b8, 0x53b9, 0x53ba, 0x53bc, 0x53bd, 0x53be,
00155 0x53c0, 0x53c3, 0x53c4, 0x53c5, 0x53c6, 0x53c7, 0x53ce, 0x53cf,
00156 0x53d0, 0x53d2, 0x53d3, 0x53d5, 0x53da, 0x53dc, 0x53dd, 0x53de,
00157 0x53e1, 0x53e2, 0x53e7, 0x53f4, 0x53fa, 0x53fe, 0x53ff, 0x5400,
00158 0x5402, 0x5405, 0x5407, 0x540b, 0x5414, 0x5418, 0x5419, 0x541a,
00159 0x541c, 0x5422, 0x5424, 0x5425, 0x542a, 0x5430, 0x5433, 0x5436,
00160 0x5437, 0x543a, 0x543d, 0x543f, 0x5441, 0x5442, 0x5444, 0x5445,
00161 0x5447, 0x5449, 0x544c, 0x544d, 0x544e, 0x544f, 0x5451, 0x545a,
00162 0x545d, 0x545e, 0x545f, 0x5460, 0x5461, 0x5463, 0x5465, 0x5467,
00163 0x5469, 0x546a, 0x546b, 0x546c, 0x546d, 0x546e, 0x546f, 0x5470,
00164 0x5474, 0x5479, 0x547a, 0x547e, 0x547f, 0x5481, 0x5483, 0x5485,
00165 0x5487, 0x5488, 0x5489, 0x548a, 0x548d, 0x5491, 0x5493, 0x5497,
00166 0x5498, 0x549c, 0x549e, 0x549f, 0x54a0, 0x54a1,
00167 /* 0x86 */
00168 0x54a2, 0x54a5, 0x54ae, 0x54b0, 0x54b2, 0x54b5, 0x54b6, 0x54b7,
00169 0x54b9, 0x54ba, 0x54bc, 0x54be, 0x54c3, 0x54c5, 0x54ca, 0x54cb,
00170 0x54d6, 0x54d8, 0x54db, 0x54de, 0x54e1, 0x54e2, 0x54e3, 0x54e4,
00171 0x54eb, 0x54ec, 0x54ef, 0x54f0, 0x54f1, 0x54f4, 0x54f5, 0x54f6,
00172 0x54f7, 0x54f8, 0x54f9, 0x54fb, 0x54fe, 0x5500, 0x5502, 0x5503,
00173 0x5504, 0x5505, 0x5508, 0x550a, 0x550b, 0x550c, 0x550d, 0x550e,
00174 0x5512, 0x5513, 0x5515, 0x5516, 0x5517, 0x5518, 0x5519, 0x551a,
00175 0x551c, 0x551d, 0x551e, 0x551f, 0x5521, 0x5525, 0x5526, 0x5528,
00176 0x5529, 0x552b, 0x552d, 0x5532, 0x5534, 0x5535, 0x5536, 0x5538,
00177 0x5539, 0x553a, 0x553b, 0x553d, 0x5540, 0x5542, 0x5545, 0x5547,
00178 0x5548, 0x554b, 0x554c, 0x554d, 0x554e, 0x554f, 0x5551, 0x5552,
00179 0x5553, 0x5554, 0x5557, 0x5558, 0x5559, 0x555a, 0x555b, 0x555d,
```

```

00180 0x555e, 0x555f, 0x5560, 0x5562, 0x5563, 0x5568, 0x5569, 0x556b,
00181 0x556f, 0x5570, 0x5571, 0x5572, 0x5573, 0x5574, 0x5579, 0x557a,
00182 0x557d, 0x557f, 0x5585, 0x5586, 0x558c, 0x558d, 0x558e, 0x5590,
00183 0x5592, 0x5593, 0x5595, 0x5596, 0x5597, 0x559a, 0x559b, 0x559e,
00184 0x55a0, 0x55a1, 0x55a2, 0x55a3, 0x55a4, 0x55a5, 0x55a6, 0x55a8,
00185 0x55a9, 0x55aa, 0x55ab, 0x55ac, 0x55ad, 0x55ae, 0x55af, 0x55b0,
00186 0x55b2, 0x55b4, 0x55b6, 0x55b8, 0x55ba, 0x55bc, 0x55bf, 0x55c0,
00187 0x55c1, 0x55c2, 0x55c3, 0x55c6, 0x55c7, 0x55c8, 0x55ca, 0x55cb,
00188 0x55ce, 0x55cf, 0x55d0, 0x55d5, 0x55d7, 0x55d8, 0x55d9, 0x55da,
00189 0x55db, 0x55de, 0x55e0, 0x55e2, 0x55e7, 0x55e9, 0x55ed, 0x55ee,
00190 0x55f0, 0x55f1, 0x55f4, 0x55f6, 0x55f8, 0x55f9, 0x55fa, 0x55fb,
00191 0x55fc, 0x55ff, 0x5602, 0x5603, 0x5604, 0x5605,
00192 /* 0x87 */
00193 0x5606, 0x5607, 0x560a, 0x560b, 0x560d, 0x5610, 0x5611, 0x5612,
00194 0x5613, 0x5614, 0x5615, 0x5616, 0x5617, 0x5619, 0x561a, 0x561c,
00195 0x561d, 0x5620, 0x5621, 0x5622, 0x5625, 0x5626, 0x5628, 0x5629,
00196 0x562a, 0x562b, 0x562e, 0x562f, 0x5630, 0x5633, 0x5635, 0x5637,
00197 0x5638, 0x563a, 0x563c, 0x563d, 0x563e, 0x5640, 0x5641, 0x5642,
00198 0x5643, 0x5644, 0x5645, 0x5646, 0x5647, 0x5648, 0x5649, 0x564a,
00199 0x564b, 0x564f, 0x5650, 0x5651, 0x5652, 0x5653, 0x5655, 0x5656,
00200 0x565a, 0x565b, 0x565d, 0x565e, 0x565f, 0x5660, 0x5661, 0x5663,
00201 0x5665, 0x5666, 0x5667, 0x566d, 0x566e, 0x566f, 0x5670, 0x5672,
00202 0x5673, 0x5674, 0x5675, 0x5677, 0x5678, 0x5679, 0x567a, 0x567d,
00203 0x567e, 0x567f, 0x5680, 0x5681, 0x5682, 0x5683, 0x5684, 0x5687,
00204 0x5688, 0x5689, 0x568a, 0x568b, 0x568c, 0x568d, 0x5690, 0x5691,
00205 0x5692, 0x5694, 0x5695, 0x5696, 0x5697, 0x5698, 0x5699, 0x569a,
00206 0x569b, 0x569c, 0x569d, 0x569e, 0x569f, 0x56a0, 0x56a1, 0x56a2,
00207 0x56a4, 0x56a5, 0x56a6, 0x56a7, 0x56a8, 0x56a9, 0x56aa, 0x56ab,
00208 0x56ac, 0x56ad, 0x56ae, 0x56b0, 0x56b1, 0x56b2, 0x56b3, 0x56b4,
00209 0x56b5, 0x56b6, 0x56b8, 0x56b9, 0x56ba, 0x56bb, 0x56bd, 0x56be,
00210 0x56bf, 0x56c0, 0x56c1, 0x56c2, 0x56c3, 0x56c4, 0x56c5, 0x56c6,
00211 0x56c7, 0x56c8, 0x56c9, 0x56cb, 0x56cc, 0x56cd, 0x56ce, 0x56cf,
00212 0x56d0, 0x56d1, 0x56d2, 0x56d3, 0x56d5, 0x56d6, 0x56d8, 0x56d9,
00213 0x56dc, 0x56e3, 0x56e5, 0x56e6, 0x56e7, 0x56e8, 0x56e9, 0x56ea,
00214 0x56ec, 0x56ee, 0x56ef, 0x56f2, 0x56f3, 0x56f6, 0x56ff, 0x56f8,
00215 0x56fb, 0x56fc, 0x5700, 0x5701, 0x5702, 0x5705, 0x5707, 0x570b,
00216 0x570c, 0x570d, 0x570e, 0x570f, 0x5710, 0x5711,
00217 /* 0x88 */
00218 0x5712, 0x5713, 0x5714, 0x5715, 0x5716, 0x5717, 0x5718, 0x5719,
00219 0x571a, 0x571b, 0x571d, 0x571e, 0x5720, 0x5721, 0x5722, 0x5724,
00220 0x5725, 0x5726, 0x5727, 0x572b, 0x5731, 0x5732, 0x5734, 0x5735,
00221 0x5736, 0x5737, 0x5738, 0x573c, 0x573d, 0x573f, 0x5741, 0x5743,
00222 0x5744, 0x5745, 0x5746, 0x5748, 0x5749, 0x574b, 0x5752, 0x5753,
00223 0x5754, 0x5755, 0x5756, 0x5758, 0x5759, 0x5762, 0x5763, 0x5765,
00224 0x5767, 0x576c, 0x576e, 0x5770, 0x5771, 0x5772, 0x5774, 0x5775,
00225 0x5778, 0x5779, 0x577a, 0x577d, 0x577e, 0x577f, 0x5780, 0x5781,
00226 0x5787, 0x5788, 0x5789, 0x578a, 0x578d, 0x578e, 0x578f, 0x5790,
00227 0x5791, 0x5794, 0x5795, 0x5796, 0x5797, 0x5798, 0x5799, 0x579a,
00228 0x579c, 0x579d, 0x579e, 0x579f, 0x57a5, 0x57a8, 0x57aa, 0x57ac,
00229 0x57af, 0x57b0, 0x57b1, 0x57b3, 0x57b5, 0x57b6, 0x57b7, 0x57b9,
00230 0x57ba, 0x57bb, 0x57bd, 0x57be, 0x57bf, 0x57c0, 0x57c1, 0x57c2,
00231 0x57c4, 0x57c5, 0x57c6, 0x57c7, 0x57c8, 0x57c9, 0x57ca, 0x57cc,
00232 0x57cd, 0x57d0, 0x57d1, 0x57d3, 0x57d6, 0x57d7, 0x57db, 0x57dc,
00233 0x57de, 0x57e1, 0x57e2, 0x57e3, 0x57e5, 0x57e6, 0x57e7, 0x57e8,
00234 0x57e9, 0x57ea, 0x57eb, 0x57ec, 0x57ee, 0x57f0, 0x57f1, 0x57f2,
00235 0x57f3, 0x57f5, 0x57f6, 0x57f7, 0x57fb, 0x57fc, 0x57fe, 0x57ff,
00236 0x5801, 0x5803, 0x5804, 0x5805, 0x5808, 0x5809, 0x580a, 0x580c,
00237 0x580e, 0x580f, 0x5810, 0x5812, 0x5813, 0x5814, 0x5816, 0x5817,
00238 0x5818, 0x581a, 0x581b, 0x581c, 0x581d, 0x581f, 0x5822, 0x5823,
00239 0x5825, 0x5826, 0x5827, 0x5828, 0x5829, 0x582b, 0x582c, 0x582d,
00240 0x582e, 0x582f, 0x5831, 0x5832, 0x5833, 0x5834, 0x5836, 0x5837,
00241 0x5838, 0x5839, 0x583a, 0x583b, 0x583c, 0x583d,
00242 /* 0x89 */
00243 0x583e, 0x583f, 0x5840, 0x5841, 0x5842, 0x5843, 0x5845, 0x5846,
00244 0x5847, 0x5848, 0x5849, 0x584a, 0x584b, 0x584e, 0x584f, 0x5850,
00245 0x5852, 0x5853, 0x5855, 0x5856, 0x5857, 0x5859, 0x585a, 0x585b,
00246 0x585c, 0x585d, 0x585f, 0x5860, 0x5861, 0x5862, 0x5863, 0x5864,
00247 0x5866, 0x5867, 0x5868, 0x5869, 0x586a, 0x586d, 0x586e, 0x586f,
00248 0x5870, 0x5871, 0x5872, 0x5873, 0x5874, 0x5875, 0x5876, 0x5877,
00249 0x5878, 0x5879, 0x587a, 0x587b, 0x587c, 0x587d, 0x587f, 0x5882,
00250 0x5884, 0x5886, 0x5887, 0x5888, 0x588a, 0x588b, 0x588c, 0x588d,
00251 0x588e, 0x588f, 0x5890, 0x5891, 0x5894, 0x5895, 0x5896, 0x5897,
00252 0x5898, 0x589b, 0x589c, 0x589d, 0x58a0, 0x58a1, 0x58a2, 0x58a3,
00253 0x58a4, 0x58a5, 0x58a6, 0x58a7, 0x58aa, 0x58ab, 0x58ac, 0x58ad,
00254 0x58ae, 0x58af, 0x58b0, 0x58b1, 0x58b2, 0x58b3, 0x58b4, 0x58b5,
00255 0x58b6, 0x58b7, 0x58b8, 0x58b9, 0x58ba, 0x58bb, 0x58bd, 0x58be,
00256 0x58bf, 0x58c0, 0x58c2, 0x58c3, 0x58c4, 0x58c6, 0x58c7, 0x58c8,
00257 0x58c9, 0x58ca, 0x58cb, 0x58cc, 0x58cd, 0x58ce, 0x58cf, 0x58d0,
00258 0x58d2, 0x58d3, 0x58d4, 0x58d6, 0x58d7, 0x58d8, 0x58d9, 0x58da,
00259 0x58db, 0x58dc, 0x58dd, 0x58de, 0x58df, 0x58e0, 0x58e1, 0x58e2,
00260 0x58e3, 0x58e5, 0x58e6, 0x58e7, 0x58e8, 0x58e9, 0x58ea, 0x58ed,
00261 0x58ef, 0x58f1, 0x58f2, 0x58f4, 0x58f5, 0x58f7, 0x58f8, 0x58fa,
00262 0x58fb, 0x58fc, 0x58fd, 0x58fe, 0x58ff, 0x5900, 0x5901, 0x5903,
00263 0x5905, 0x5906, 0x5908, 0x5909, 0x590a, 0x590b, 0x590c, 0x590e,
00264 0x5910, 0x5911, 0x5912, 0x5913, 0x5917, 0x5918, 0x591b, 0x591d,
00265 0x591e, 0x5920, 0x5921, 0x5922, 0x5923, 0x5926, 0x5928, 0x592c,
00266 0x5930, 0x5932, 0x5933, 0x5935, 0x5936, 0x593b,

```

```
00267  /* 0x8a */
00268  0x593d, 0x593e, 0x593f, 0x5940, 0x5943, 0x5945, 0x5946, 0x594a,
00269  0x594c, 0x594d, 0x594e, 0x5950, 0x5952, 0x5953, 0x5959, 0x595b, 0x595c,
00270  0x595d, 0x595e, 0x595f, 0x5961, 0x5963, 0x5964, 0x5966, 0x5967,
00271  0x5968, 0x5969, 0x596a, 0x596b, 0x596c, 0x596d, 0x596e, 0x596f,
00272  0x5970, 0x5971, 0x5972, 0x5975, 0x5977, 0x597a, 0x597b, 0x597c,
00273  0x597e, 0x597f, 0x5980, 0x5985, 0x5989, 0x598b, 0x598c, 0x598e,
00274  0x598f, 0x5990, 0x5991, 0x5994, 0x5995, 0x5998, 0x599a, 0x599b,
00275  0x599c, 0x599d, 0x599f, 0x59a0, 0x59a1, 0x59a2, 0x59a6, 0x59a7,
00276  0x59ac, 0x59ad, 0x59b0, 0x59b1, 0x59b3, 0x59b4, 0x59b5, 0x59b6,
00277  0x59b7, 0x59b8, 0x59ba, 0x59bc, 0x59bd, 0x59bf, 0x59c0, 0x59c1,
00278  0x59c2, 0x59c3, 0x59c4, 0x59c5, 0x59c7, 0x59c8, 0x59c9, 0x59cc,
00279  0x59cd, 0x59ce, 0x59cf, 0x59d5, 0x59d6, 0x59d9, 0x59db, 0x59de,
00280  0x59df, 0x59e0, 0x59e1, 0x59e2, 0x59e4, 0x59e6, 0x59e7, 0x59e9,
00281  0x59ea, 0x59eb, 0x59ed, 0x59ee, 0x59ef, 0x59f0, 0x59f1, 0x59f2,
00282  0x59f3, 0x59f4, 0x59f5, 0x59f6, 0x59f7, 0x59f8, 0x59fa, 0x59fc,
00283  0x59fd, 0x59fe, 0x5a00, 0x5a02, 0x5a0a, 0x5a0b, 0x5a0d, 0x5a0e,
00284  0x5a0f, 0x5a10, 0x5a12, 0x5a14, 0x5a15, 0x5a16, 0x5a17, 0x5a19,
00285  0x5a1a, 0x5a1b, 0x5a1d, 0x5a1e, 0x5a21, 0x5a22, 0x5a24, 0x5a26,
00286  0x5a27, 0x5a28, 0x5a2a, 0x5a2b, 0x5a2c, 0x5a2d, 0x5a2e, 0x5a2f,
00287  0x5a30, 0x5a32, 0x5a33, 0x5a35, 0x5a37, 0x5a38, 0x5a39, 0x5a3a, 0x5a3b,
00288  0x5a3d, 0x5a3e, 0x5a3f, 0x5a41, 0x5a42, 0x5a43, 0x5a44, 0x5a45,
00289  0x5a47, 0x5a48, 0x5a4b, 0x5a4c, 0x5a4d, 0x5a4e, 0x5a4f, 0x5a50,
00290  0x5a51, 0x5a52, 0x5a54, 0x5a55, 0x5a56, 0x5a57, 0x5a58, 0x5a59,
00291  0x5a5b, 0x5a5c, 0x5a5d, 0x5a5e, 0x5a5f, 0x5a60,
00292  /* 0x8b */
00293  0x5a61, 0x5a63, 0x5a64, 0x5a65, 0x5a66, 0x5a68, 0x5a69, 0x5a6b,
00294  0x5a6c, 0x5a6d, 0x5a6e, 0x5a6f, 0x5a70, 0x5a71, 0x5a72, 0x5a73,
00295  0x5a78, 0x5a79, 0x5a7b, 0x5a7c, 0x5a7d, 0x5a7e, 0x5a80, 0x5a81,
00296  0x5a82, 0x5a83, 0x5a84, 0x5a85, 0x5a86, 0x5a87, 0x5a88, 0x5a89,
00297  0x5a8a, 0x5a8b, 0x5a8c, 0x5a8d, 0x5a8e, 0x5a8f, 0x5a90, 0x5a91,
00298  0x5a93, 0x5a94, 0x5a95, 0x5a96, 0x5a97, 0x5a98, 0x5a99, 0x5a9c,
00299  0x5a9d, 0x5a9e, 0x5a9f, 0x5aa0, 0x5aa1, 0x5aa2, 0x5aa3, 0x5aa4,
00300  0x5aa5, 0x5aa6, 0x5aa7, 0x5aa8, 0x5aa9, 0x5aab, 0x5aac, 0x5aad,
00301  0x5aae, 0x5aaf, 0x5ab0, 0x5ab1, 0x5ab4, 0x5ab6, 0x5ab7, 0x5ab9,
00302  0x5aba, 0x5abb, 0x5abc, 0x5abd, 0x5abf, 0x5ac0, 0x5ac3, 0x5ac4,
00303  0x5ac5, 0x5ac6, 0x5ac7, 0x5ac8, 0x5aca, 0x5acb, 0x5acd, 0x5ace,
00304  0x5acf, 0x5ad0, 0x5ad1, 0x5ad3, 0x5ad5, 0x5ad7, 0x5ad9, 0x5ada,
00305  0x5adb, 0x5add, 0x5ade, 0x5adf, 0x5ae2, 0x5ae4, 0x5ae5, 0x5ae7,
00306  0x5ae8, 0x5aea, 0x5aec, 0x5aed, 0x5aee, 0x5aef, 0x5af0, 0x5af2,
00307  0x5af3, 0x5af4, 0x5af5, 0x5af6, 0x5af7, 0x5af8, 0x5af9, 0x5afa,
00308  0x5afb, 0x5afc, 0x5afd, 0x5afe, 0x5aff, 0x5b00, 0x5b01, 0x5b02,
00309  0x5b03, 0x5b04, 0x5b05, 0x5b06, 0x5b07, 0x5b08, 0x5b0a, 0x5b0b,
00310  0x5b0c, 0x5b0d, 0x5b0e, 0x5b0f, 0x5b10, 0x5b11, 0x5b12, 0x5b13,
00311  0x5b14, 0x5b15, 0x5b18, 0x5b19, 0x5b1a, 0x5b1b, 0x5b1c, 0x5b1d,
00312  0x5b1e, 0x5b1f, 0x5b20, 0x5b21, 0x5b22, 0x5b23, 0x5b24, 0x5b25,
00313  0x5b26, 0x5b27, 0x5b28, 0x5b29, 0x5b2a, 0x5b2b, 0x5b2c, 0x5b2d,
00314  0x5b2e, 0x5b2f, 0x5b30, 0x5b31, 0x5b33, 0x5b35, 0x5b36, 0x5b38,
00315  0x5b39, 0x5b3a, 0x5b3b, 0x5b3c, 0x5b3d, 0x5b3e, 0x5b3f, 0x5b41,
00316  0x5b42, 0x5b43, 0x5b44, 0x5b45, 0x5b46, 0x5b47,
00317  /* 0x8c */
00318  0x5b48, 0x5b49, 0x5b4a, 0x5b4b, 0x5b4c, 0x5b4d, 0x5b4e, 0x5b4f,
00319  0x5b52, 0x5b56, 0x5b5e, 0x5b60, 0x5b61, 0x5b67, 0x5b68, 0x5b6b,
00320  0x5b6d, 0x5b6e, 0x5b6f, 0x5b72, 0x5b74, 0x5b76, 0x5b77, 0x5b78,
00321  0x5b79, 0x5b7b, 0x5b7c, 0x5b7e, 0x5b7f, 0x5b82, 0x5b86, 0x5b8a,
00322  0x5b8d, 0x5b8e, 0x5b90, 0x5b91, 0x5b92, 0x5b94, 0x5b96, 0x5b9f,
00323  0x5ba7, 0x5ba8, 0x5ba9, 0x5bac, 0x5bad, 0x5bae, 0x5baf, 0x5bb1,
00324  0x5bb2, 0x5bb7, 0x5bba, 0x5bbb, 0x5bbc, 0x5bc0, 0x5bc1, 0x5bc3,
00325  0x5bc8, 0x5bc9, 0x5bca, 0x5bcb, 0x5bcd, 0x5bce, 0x5bcf, 0x5bd1,
00326  0x5bd4, 0x5bd5, 0x5bd6, 0x5bd7, 0x5bd8, 0x5bd9, 0x5bda, 0x5bdb,
00327  0x5bdc, 0x5be0, 0x5be2, 0x5be3, 0x5be6, 0x5be7, 0x5be9, 0x5bea,
00328  0x5beb, 0x5bec, 0x5bed, 0x5bef, 0x5bf1, 0x5bf2, 0x5bf3, 0x5bf4,
00329  0x5bf5, 0x5bf6, 0x5bf7, 0x5bfd, 0x5bfe, 0x5c00, 0x5c02, 0x5c03,
00330  0x5c05, 0x5c07, 0x5c08, 0x5c0b, 0x5c0c, 0x5c0d, 0x5c0e, 0x5c10,
00331  0x5c12, 0x5c13, 0x5c17, 0x5c19, 0x5c1b, 0x5c1e, 0x5c1f, 0x5c20,
00332  0x5c21, 0x5c23, 0x5c26, 0x5c28, 0x5c29, 0x5c2a, 0x5c2b, 0x5c2d,
00333  0x5c2e, 0x5c2f, 0x5c30, 0x5c32, 0x5c33, 0x5c35, 0x5c36, 0x5c37,
00334  0x5c43, 0x5c44, 0x5c46, 0x5c47, 0x5c4c, 0x5c4d, 0x5c52, 0x5c53,
00335  0x5c54, 0x5c56, 0x5c57, 0x5c58, 0x5c5a, 0x5c5b, 0x5c5c, 0x5c5d,
00336  0x5c5f, 0x5c62, 0x5c64, 0x5c67, 0x5c68, 0x5c69, 0x5c6a, 0x5c6b,
00337  0x5c6c, 0x5c6d, 0x5c70, 0x5c72, 0x5c73, 0x5c74, 0x5c75, 0x5c76,
00338  0x5c77, 0x5c78, 0x5c7b, 0x5c7c, 0x5c7d, 0x5c7e, 0x5c80, 0x5c83,
00339  0x5c84, 0x5c85, 0x5c86, 0x5c87, 0x5c89, 0x5c8a, 0x5c8b, 0x5c8e,
00340  0x5c8f, 0x5c92, 0x5c93, 0x5c95, 0x5c9d, 0x5c9e, 0x5c9f, 0x5ca0,
00341  0x5ca1, 0x5ca4, 0x5ca5, 0x5ca6, 0x5ca7, 0x5ca8,
00342  /* 0x8d */
00343  0x5caa, 0x5cae, 0x5caf, 0x5cb0, 0x5cb2, 0x5cb4, 0x5cb6, 0x5cb9,
00344  0x5cba, 0x5cbb, 0x5cbc, 0x5cbe, 0x5cc0, 0x5cc2, 0x5cc3, 0x5cc5,
00345  0x5cc6, 0x5cc7, 0x5cc8, 0x5cc9, 0x5cca, 0x5ccc, 0x5ccd, 0x5cce,
00346  0x5ccf, 0x5cd0, 0x5cd1, 0x5cd3, 0x5cd4, 0x5cd5, 0x5cd6, 0x5cd7,
00347  0x5cd8, 0x5cda, 0x5cdb, 0x5cdc, 0x5cdd, 0x5cde, 0x5cdf, 0x5ce0,
00348  0x5ce2, 0x5ce3, 0x5ce7, 0x5ce9, 0x5ceb, 0x5cec, 0x5cee, 0x5cef,
00349  0x5cf1, 0x5cf2, 0x5cf3, 0x5cf4, 0x5cf5, 0x5cf6, 0x5cf7, 0x5cf8,
00350  0x5cf9, 0x5cfa, 0x5cfc, 0x5cfd, 0x5cfe, 0x5cff, 0x5d00, 0x5d01,
00351  0x5d04, 0x5d05, 0x5d08, 0x5d09, 0x5d0a, 0x5d0b, 0x5d0c, 0x5d0d,
00352  0x5d0f, 0x5d10, 0x5d11, 0x5d12, 0x5d13, 0x5d15, 0x5d17, 0x5d18,
00353  0x5d19, 0x5d1a, 0x5d1c, 0x5d1d, 0x5d1f, 0x5d20, 0x5d21, 0x5d22,
```

```

00354 0x5d23, 0x5d25, 0x5d28, 0x5d2a, 0x5d2b, 0x5d2c, 0x5d2f, 0x5d30,
00355 0x5d31, 0x5d32, 0x5d33, 0x5d35, 0x5d36, 0x5d37, 0x5d38, 0x5d39,
00356 0x5d3a, 0x5d3b, 0x5d3c, 0x5d3f, 0x5d40, 0x5d41, 0x5d42, 0x5d43,
00357 0x5d44, 0x5d45, 0x5d46, 0x5d48, 0x5d49, 0x5d4d, 0x5d4e, 0x5d4f,
00358 0x5d50, 0x5d51, 0x5d52, 0x5d53, 0x5d54, 0x5d55, 0x5d56, 0x5d57,
00359 0x5d59, 0x5d5a, 0x5d5c, 0x5d5e, 0x5d5f, 0x5d60, 0x5d61, 0x5d62,
00360 0x5d63, 0x5d64, 0x5d65, 0x5d66, 0x5d67, 0x5d68, 0x5d6a, 0x5d6d,
00361 0x5d6e, 0x5d70, 0x5d71, 0x5d72, 0x5d73, 0x5d75, 0x5d76, 0x5d77,
00362 0x5d78, 0x5d79, 0x5d7a, 0x5d7b, 0x5d7c, 0x5d7d, 0x5d7e, 0x5d7f,
00363 0x5d80, 0x5d81, 0x5d83, 0x5d84, 0x5d85, 0x5d86, 0x5d87, 0x5d88,
00364 0x5d89, 0x5d8a, 0x5d8b, 0x5d8c, 0x5d8d, 0x5d8e, 0x5d8f, 0x5d90,
00365 0x5d91, 0x5d92, 0x5d93, 0x5d94, 0x5d95, 0x5d96, 0x5d97, 0x5d98,
00366 0x5d9a, 0x5d9b, 0x5d9c, 0x5d9e, 0x5d9f, 0x5da0,
00367 /* 0x8e */
00368 0x5da1, 0x5da2, 0x5da3, 0x5da4, 0x5da5, 0x5da6, 0x5da7, 0x5da8,
00369 0x5da9, 0x5daa, 0x5dab, 0x5dac, 0x5dad, 0x5dae, 0x5daf, 0x5db0,
00370 0x5db1, 0x5db2, 0x5db3, 0x5db4, 0x5db5, 0x5db6, 0x5dbb, 0x5db9,
00371 0x5dba, 0x5dbb, 0x5dbc, 0x5dbd, 0x5dbe, 0x5dbf, 0x5dc0, 0x5dc1,
00372 0x5dc2, 0x5dc3, 0x5dc4, 0x5dc6, 0x5dc7, 0x5dc8, 0x5dc9, 0x5dca,
00373 0x5dcb, 0x5dcc, 0x5dce, 0x5dcf, 0x5dd0, 0x5dd1, 0x5dd2, 0x5dd3,
00374 0x5dd4, 0x5dd5, 0x5dd6, 0x5dd7, 0x5dd8, 0x5dd9, 0x5dda, 0x5ddc,
00375 0x5ddf, 0x5de0, 0x5de3, 0x5de4, 0x5dea, 0x5dec, 0x5ded, 0x5df0,
00376 0x5df5, 0x5df6, 0x5df8, 0x5df9, 0x5dfa, 0x5dfb, 0x5dfc, 0x5dff,
00377 0x5e00, 0x5e04, 0x5e07, 0x5e09, 0x5e0a, 0x5e0b, 0x5e0d, 0x5e0e,
00378 0x5e12, 0x5e13, 0x5e17, 0x5e1e, 0x5e1f, 0x5e20, 0x5e21, 0x5e22,
00379 0x5e23, 0x5e24, 0x5e25, 0x5e28, 0x5e29, 0x5e2a, 0x5e2b, 0x5e2c,
00380 0x5e2f, 0x5e30, 0x5e32, 0x5e33, 0x5e34, 0x5e35, 0x5e36, 0x5e39,
00381 0x5e3a, 0x5e3e, 0x5e3f, 0x5e40, 0x5e41, 0x5e43, 0x5e46, 0x5e47,
00382 0x5e48, 0x5e49, 0x5e4a, 0x5e4b, 0x5e4d, 0x5e4e, 0x5e4f, 0x5e50,
00383 0x5e51, 0x5e52, 0x5e53, 0x5e56, 0x5e57, 0x5e58, 0x5e59, 0x5e5a,
00384 0x5e5c, 0x5e5d, 0x5e5f, 0x5e60, 0x5e63, 0x5e64, 0x5e65, 0x5e66,
00385 0x5e67, 0x5e68, 0x5e69, 0x5e6a, 0x5e6b, 0x5e6c, 0x5e6d, 0x5e6e,
00386 0x5e6f, 0x5e70, 0x5e71, 0x5e75, 0x5e77, 0x5e79, 0x5e7e, 0x5e81,
00387 0x5e82, 0x5e83, 0x5e85, 0x5e88, 0x5e89, 0x5e8c, 0x5e8d, 0x5e8e,
00388 0x5e92, 0x5e98, 0x5e9b, 0x5e9d, 0x5ea1, 0x5ea2, 0x5ea3, 0x5ea4,
00389 0x5ea8, 0x5ea9, 0x5eaa, 0x5eab, 0x5eac, 0x5eae, 0x5eaf, 0x5eb0,
00390 0x5eb1, 0x5eb2, 0x5eb4, 0x5eba, 0x5ebb, 0x5ebc, 0x5ebd, 0x5ebf,
00391 0x5ec0, 0x5ec1, 0x5ec2, 0x5ec3, 0x5ec4, 0x5ec5,
00392 /* 0x8f */
00393 0x5ec6, 0x5ec7, 0x5ec8, 0x5ecb, 0x5ecc, 0x5ecd, 0x5ece, 0x5ecf,
00394 0x5ed0, 0x5ed4, 0x5ed5, 0x5ed7, 0x5ed8, 0x5ed9, 0x5eda, 0x5edc,
00395 0x5edd, 0x5ede, 0x5edf, 0x5ee0, 0x5ee1, 0x5ee2, 0x5ee3, 0x5ee4,
00396 0x5ee5, 0x5ee6, 0x5ee7, 0x5ee9, 0x5eeb, 0x5eec, 0x5eed, 0x5eee,
00397 0x5eef, 0x5ef0, 0x5ef1, 0x5ef2, 0x5ef3, 0x5ef5, 0x5ef8, 0x5ef9,
00398 0x5efb, 0x5efc, 0x5efd, 0x5ef5, 0x5ef6, 0x5ef7, 0x5ef8, 0x5ef9,
00399 0x5f0d, 0x5f0e, 0x5f10, 0x5f12, 0x5f14, 0x5f16, 0x5f19, 0x5f1a,
00400 0x5f1c, 0x5f1d, 0x5f1e, 0x5f21, 0x5f22, 0x5f23, 0x5f24, 0x5f28,
00401 0x5f2b, 0x5f2c, 0x5f2e, 0x5f30, 0x5f32, 0x5f33, 0x5f34, 0x5f35,
00402 0x5f36, 0x5f37, 0x5f38, 0x5f3b, 0x5f3d, 0x5f3e, 0x5f3f, 0x5f41,
00403 0x5f42, 0x5f43, 0x5f44, 0x5f45, 0x5f46, 0x5f47, 0x5f48, 0x5f49,
00404 0x5f4a, 0x5f4b, 0x5f4c, 0x5f4d, 0x5f4e, 0x5f4f, 0x5f51, 0x5f54,
00405 0x5f59, 0x5f5a, 0x5f5b, 0x5f5c, 0x5f5e, 0x5f5f, 0x5f60, 0x5f63,
00406 0x5f65, 0x5f67, 0x5f68, 0x5f6b, 0x5f6e, 0x5f6f, 0x5f72, 0x5f74,
00407 0x5f75, 0x5f76, 0x5f78, 0x5f7a, 0x5f7d, 0x5f7e, 0x5f7f, 0x5f83,
00408 0x5f86, 0x5f8d, 0x5f8e, 0x5f8f, 0x5f91, 0x5f93, 0x5f94, 0x5f96,
00409 0x5f9a, 0x5f9b, 0x5f9d, 0x5f9e, 0x5f9f, 0x5fa0, 0x5fa2, 0x5fa3,
00410 0x5fa4, 0x5fa5, 0x5fa6, 0x5fa7, 0x5fa9, 0x5fab, 0x5fac, 0x5faf,
00411 0x5fb0, 0x5fb1, 0x5fb2, 0x5fb3, 0x5fb4, 0x5fb6, 0x5fb8, 0x5fb9,
00412 0x5fba, 0x5fbb, 0x5fbe, 0x5fbf, 0x5fc0, 0x5fc1, 0x5fc2, 0x5fc7,
00413 0x5fc8, 0x5fcb, 0x5fce, 0x5fcd, 0x5fd3, 0x5fd4, 0x5fd5, 0x5fda,
00414 0x5fdb, 0x5fde, 0x5fde, 0x5fdf, 0x5fe2, 0x5fe3, 0x5fe5, 0x5fe6,
00415 0x5fe8, 0x5fe9, 0x5fec, 0x5fef, 0x5ff0, 0x5ff2, 0x5ff3, 0x5ff4,
00416 0x5ff6, 0x5ff7, 0x5ff9, 0x5ffa, 0x5ffc, 0x6007,
00417 /* 0x90 */
00418 0x6008, 0x6009, 0x600b, 0x600c, 0x6010, 0x6011, 0x6013, 0x6017,
00419 0x6018, 0x601a, 0x601e, 0x601f, 0x6022, 0x6023, 0x6024, 0x602c,
00420 0x602d, 0x602e, 0x6030, 0x6031, 0x6032, 0x6033, 0x6034, 0x6036,
00421 0x6037, 0x6038, 0x6039, 0x603a, 0x603d, 0x603e, 0x6040, 0x6044,
00422 0x6045, 0x6046, 0x6047, 0x6048, 0x6049, 0x604a, 0x604c, 0x604e,
00423 0x604f, 0x6051, 0x6053, 0x6054, 0x6056, 0x6057, 0x6058, 0x605b,
00424 0x605c, 0x605e, 0x605f, 0x6060, 0x6061, 0x6065, 0x6066, 0x606e,
00425 0x6071, 0x6072, 0x6074, 0x6075, 0x6077, 0x607e, 0x6080, 0x6081,
00426 0x6082, 0x6085, 0x6086, 0x6087, 0x6088, 0x608a, 0x608b, 0x608e,
00427 0x608f, 0x6090, 0x6091, 0x6093, 0x6095, 0x6097, 0x6098, 0x6099,
00428 0x609c, 0x609e, 0x60a1, 0x60a2, 0x60a4, 0x60a5, 0x60a7, 0x60a9,
00429 0x60aa, 0x60ae, 0x60b0, 0x60b3, 0x60b5, 0x60b6, 0x60b7, 0x60b9,
00430 0x60ba, 0x60bd, 0x60be, 0x60bf, 0x60c0, 0x60c1, 0x60c2, 0x60c3,
00431 0x60c4, 0x60c7, 0x60c8, 0x60c9, 0x60cc, 0x60cd, 0x60ce, 0x60cf,
00432 0x60d0, 0x60d2, 0x60d3, 0x60d4, 0x60d6, 0x60d7, 0x60d9, 0x60db,
00433 0x60de, 0x60e1, 0x60e2, 0x60e3, 0x60e4, 0x60e5, 0x60ea, 0x60f1,
00434 0x60f2, 0x60f5, 0x60f7, 0x60f8, 0x60fb, 0x60fc, 0x60fd, 0x60fe,
00435 0x60ff, 0x6102, 0x6103, 0x6104, 0x6105, 0x6107, 0x610a, 0x610b,
00436 0x610c, 0x6110, 0x6111, 0x6112, 0x6113, 0x6114, 0x6116, 0x6117,
00437 0x6118, 0x6119, 0x611b, 0x611c, 0x611d, 0x611e, 0x6121, 0x6122,
00438 0x6125, 0x6128, 0x6129, 0x612a, 0x612c, 0x612d, 0x612e, 0x612f,
00439 0x6130, 0x6131, 0x6132, 0x6133, 0x6134, 0x6135, 0x6136, 0x6137,
00440 0x6138, 0x6139, 0x613a, 0x613b, 0x613c, 0x613d, 0x613e, 0x6140,

```

```
00441 0x6141, 0x6142, 0x6143, 0x6144, 0x6145, 0x6146,
00442 /* 0x91 */
00443 0x6147, 0x6149, 0x614b, 0x614d, 0x614f, 0x6150, 0x6152, 0x6153,
00444 0x6154, 0x6156, 0x6157, 0x6158, 0x6159, 0x615a, 0x615b, 0x615c,
00445 0x615e, 0x615f, 0x6160, 0x6161, 0x6163, 0x6164, 0x6165, 0x6166,
00446 0x6169, 0x616a, 0x616b, 0x616c, 0x616d, 0x616e, 0x616f, 0x6171,
00447 0x6172, 0x6173, 0x6174, 0x6176, 0x6178, 0x6179, 0x617a, 0x617b,
00448 0x617c, 0x617d, 0x617e, 0x617f, 0x6180, 0x6181, 0x6182, 0x6183,
00449 0x6184, 0x6185, 0x6186, 0x6187, 0x6188, 0x6189, 0x618a, 0x618c,
00450 0x618d, 0x618f, 0x6190, 0x6191, 0x6192, 0x6193, 0x6195, 0x6196,
00451 0x6197, 0x6198, 0x6199, 0x619a, 0x619b, 0x619c, 0x619e, 0x619f,
00452 0x61a0, 0x61a1, 0x61a2, 0x61a3, 0x61a4, 0x61a5, 0x61a6, 0x61aa,
00453 0x61ab, 0x61ad, 0x61ae, 0x61af, 0x61b0, 0x61b1, 0x61b2, 0x61b3,
00454 0x61b4, 0x61b5, 0x61b6, 0x61b8, 0x61b9, 0x61ba, 0x61bb, 0x61bc,
00455 0x61bd, 0x61bf, 0x61c0, 0x61c1, 0x61c3, 0x61c4, 0x61c5, 0x61c6,
00456 0x61c7, 0x61c9, 0x61cc, 0x61cd, 0x61ce, 0x61cf, 0x61d0, 0x61d3,
00457 0x61d5, 0x61d6, 0x61d7, 0x61d8, 0x61d9, 0x61da, 0x61db, 0x61dc,
00458 0x61dd, 0x61de, 0x61df, 0x61e0, 0x61e1, 0x61e2, 0x61e3, 0x61e4,
00459 0x61e5, 0x61e7, 0x61e8, 0x61e9, 0x61ea, 0x61eb, 0x61ec, 0x61ed,
00460 0x61ee, 0x61ef, 0x61f0, 0x61f1, 0x61f2, 0x61f3, 0x61f4, 0x61f6,
00461 0x61f7, 0x61f8, 0x61f9, 0x61fa, 0x61fb, 0x61fc, 0x61fd, 0x61fe,
00462 0x6200, 0x6201, 0x6202, 0x6203, 0x6204, 0x6205, 0x6207, 0x6209,
00463 0x6213, 0x6214, 0x6219, 0x621c, 0x621d, 0x621e, 0x6220, 0x6223,
00464 0x6226, 0x6227, 0x6228, 0x6229, 0x622b, 0x622d, 0x622f, 0x6230,
00465 0x6231, 0x6232, 0x6235, 0x6236, 0x6238, 0x6239, 0x623a, 0x623b,
00466 0x623c, 0x6242, 0x6244, 0x6245, 0x6246, 0x624a,
00467 /* 0x92 */
00468 0x624f, 0x6250, 0x6255, 0x6256, 0x6257, 0x6259, 0x625a, 0x625c,
00469 0x625d, 0x625e, 0x625f, 0x6260, 0x6261, 0x6262, 0x6264, 0x6265,
00470 0x6268, 0x6271, 0x6272, 0x6274, 0x6275, 0x6277, 0x6278, 0x627a,
00471 0x627b, 0x627d, 0x6281, 0x6282, 0x6283, 0x6285, 0x6286, 0x6287,
00472 0x6288, 0x628b, 0x628c, 0x628d, 0x628e, 0x628f, 0x6290, 0x6294,
00473 0x6299, 0x629c, 0x629d, 0x629e, 0x62a3, 0x62a6, 0x62a7, 0x62a9,
00474 0x62aa, 0x62ad, 0x62ae, 0x62af, 0x62b0, 0x62b2, 0x62b3, 0x62b4,
00475 0x62b6, 0x62b7, 0x62b8, 0x62ba, 0x62be, 0x62c0, 0x62c1, 0x62c3,
00476 0x62cb, 0x62cf, 0x62d1, 0x62d5, 0x62dd, 0x62de, 0x62e0, 0x62e1,
00477 0x62e4, 0x62ea, 0x62eb, 0x62f0, 0x62f2, 0x62f5, 0x62f8, 0x62f9,
00478 0x62fa, 0x62fb, 0x6300, 0x6303, 0x6304, 0x6305, 0x6306, 0x630a,
00479 0x630b, 0x630c, 0x630d, 0x630f, 0x6310, 0x6312, 0x6313, 0x6314,
00480 0x6315, 0x6317, 0x6318, 0x6319, 0x631c, 0x6326, 0x6327, 0x6329,
00481 0x632c, 0x632d, 0x632e, 0x6330, 0x6331, 0x6333, 0x6334, 0x6335,
00482 0x6336, 0x6337, 0x6338, 0x633b, 0x633c, 0x633e, 0x633f, 0x6340,
00483 0x6341, 0x6344, 0x6347, 0x6348, 0x634a, 0x6351, 0x6352, 0x6353,
00484 0x6354, 0x6356, 0x6357, 0x6358, 0x6359, 0x635a, 0x635b, 0x635c,
00485 0x635d, 0x6360, 0x6364, 0x6365, 0x6366, 0x6368, 0x636a, 0x636b,
00486 0x636c, 0x636f, 0x6370, 0x6372, 0x6373, 0x6374, 0x6375, 0x6378,
00487 0x6379, 0x637c, 0x637d, 0x637e, 0x637f, 0x6381, 0x6383, 0x6384,
00488 0x6385, 0x6386, 0x638b, 0x638d, 0x6391, 0x6393, 0x6394, 0x6395,
00489 0x6397, 0x6399, 0x639a, 0x639b, 0x639c, 0x639d, 0x639e, 0x639f,
00490 0x63a1, 0x63a4, 0x63a6, 0x63ab, 0x63af, 0x63b1, 0x63b2, 0x63b5,
00491 0x63b6, 0x63b9, 0x63bb, 0x63bd, 0x63bf, 0x63c0,
00492 /* 0x93 */
00493 0x63c1, 0x63c2, 0x63c3, 0x63c5, 0x63c7, 0x63c8, 0x63ca, 0x63cb,
00494 0x63cc, 0x63d1, 0x63d3, 0x63d4, 0x63d5, 0x63d7, 0x63d8, 0x63d9,
00495 0x63da, 0x63db, 0x63dc, 0x63dd, 0x63df, 0x63e2, 0x63e4, 0x63e5,
00496 0x63e6, 0x63e7, 0x63e8, 0x63eb, 0x63ec, 0x63ee, 0x63ef, 0x63f0,
00497 0x63f1, 0x63f3, 0x63f5, 0x63f7, 0x63f9, 0x63fa, 0x63fb, 0x63fc,
00498 0x63fe, 0x6403, 0x6404, 0x6406, 0x6407, 0x6408, 0x6409, 0x640a,
00499 0x640d, 0x640e, 0x6411, 0x6412, 0x6415, 0x6416, 0x6417, 0x6418,
00500 0x6419, 0x641a, 0x641d, 0x641f, 0x6422, 0x6423, 0x6424, 0x6425,
00501 0x6427, 0x6428, 0x6429, 0x642b, 0x642e, 0x642f, 0x6430, 0x6431,
00502 0x6432, 0x6433, 0x6435, 0x6436, 0x6437, 0x6438, 0x6439, 0x643b,
00503 0x643c, 0x643e, 0x6440, 0x6442, 0x6443, 0x6449, 0x644b, 0x644c,
00504 0x644d, 0x644e, 0x644f, 0x6450, 0x6451, 0x6453, 0x6455, 0x6456,
00505 0x6457, 0x6459, 0x645a, 0x645b, 0x645c, 0x645d, 0x645f, 0x6460,
00506 0x6461, 0x6462, 0x6463, 0x6464, 0x6465, 0x6466, 0x6468, 0x646a,
00507 0x646b, 0x646c, 0x646e, 0x646f, 0x6470, 0x6471, 0x6472, 0x6473,
00508 0x6474, 0x6475, 0x6476, 0x6477, 0x647b, 0x647c, 0x647d, 0x647e,
00509 0x647f, 0x6480, 0x6481, 0x6483, 0x6486, 0x6488, 0x6489, 0x648a,
00510 0x648b, 0x648c, 0x648d, 0x648e, 0x648f, 0x6490, 0x6493, 0x6494,
00511 0x6497, 0x6498, 0x649a, 0x649b, 0x649c, 0x649d, 0x649f, 0x64a0,
00512 0x64a1, 0x64a2, 0x64a3, 0x64a5, 0x64a6, 0x64a7, 0x64a8, 0x64aa,
00513 0x64ab, 0x64af, 0x64b1, 0x64b2, 0x64b3, 0x64b4, 0x64b6, 0x64b9,
00514 0x64bb, 0x64bd, 0x64be, 0x64bf, 0x64c1, 0x64c3, 0x64c4, 0x64c6,
00515 0x64c7, 0x64c8, 0x64c9, 0x64ca, 0x64cb, 0x64cc, 0x64cd,
00516 0x64d3, 0x64d4, 0x64d5, 0x64d6, 0x64d9, 0x64da,
00517 /* 0x94 */
00518 0x64db, 0x64dc, 0x64dd, 0x64df, 0x64e0, 0x64e1, 0x64e3, 0x64e5,
00519 0x64e7, 0x64e8, 0x64e9, 0x64ea, 0x64eb, 0x64ec, 0x64ed, 0x64ee,
00520 0x64ef, 0x64f0, 0x64f1, 0x64f2, 0x64f3, 0x64f4, 0x64f5, 0x64f6,
00521 0x64f7, 0x64f8, 0x64f9, 0x64fa, 0x64fb, 0x64fc, 0x64fd, 0x64fe,
00522 0x64ff, 0x6501, 0x6502, 0x6503, 0x6504, 0x6505, 0x6506, 0x6507,
00523 0x6508, 0x650a, 0x650b, 0x650c, 0x650d, 0x650e, 0x650f, 0x6510,
00524 0x6511, 0x6513, 0x6514, 0x6515, 0x6516, 0x6517, 0x6519, 0x651a,
00525 0x651b, 0x651c, 0x651d, 0x651e, 0x651f, 0x6520, 0x6521, 0x6522,
00526 0x6523, 0x6524, 0x6526, 0x6527, 0x6528, 0x6529, 0x652a, 0x652c,
00527 0x652d, 0x6530, 0x6531, 0x6532, 0x6533, 0x6537, 0x653a, 0x653c,
```

```
00528 0x653d, 0x6540, 0x6541, 0x6542, 0x6543, 0x6544, 0x6546, 0x6547,
00529 0x654a, 0x654b, 0x654d, 0x654e, 0x6550, 0x6552, 0x6553, 0x6554,
00530 0x6557, 0x6558, 0x655a, 0x655c, 0x655f, 0x6560, 0x6561, 0x6564,
00531 0x6565, 0x6567, 0x6568, 0x6569, 0x656a, 0x656d, 0x656e, 0x656f,
00532 0x6571, 0x6573, 0x6575, 0x6576, 0x6578, 0x6579, 0x657a, 0x657b,
00533 0x657c, 0x657d, 0x657e, 0x657f, 0x6580, 0x6581, 0x6582, 0x6583,
00534 0x6584, 0x6585, 0x6586, 0x6588, 0x6589, 0x658a, 0x658d, 0x658e,
00535 0x658f, 0x6592, 0x6594, 0x6595, 0x6596, 0x6598, 0x659a, 0x659d,
00536 0x659e, 0x65a0, 0x65a2, 0x65a3, 0x65a6, 0x65a8, 0x65aa, 0x65ac,
00537 0x65ae, 0x65b1, 0x65b2, 0x65b3, 0x65b4, 0x65b5, 0x65b6, 0x65b7,
00538 0x65b8, 0x65ba, 0x65bb, 0x65be, 0x65bf, 0x65c0, 0x65c2, 0x65c7,
00539 0x65c8, 0x65c9, 0x65ca, 0x65cd, 0x65d0, 0x65d1, 0x65d3, 0x65d4,
00540 0x65d5, 0x65d8, 0x65d9, 0x65da, 0x65db, 0x65dc, 0x65dd, 0x65de,
00541 0x65df, 0x65e1, 0x65e3, 0x65e4, 0x65ea, 0x65eb,
00542 /* 0x95 */
00543 0x65f2, 0x65f3, 0x65f4, 0x65f5, 0x65f8, 0x65f9, 0x65fb, 0x65fc,
00544 0x65fd, 0x65fe, 0x65ff, 0x6601, 0x6604, 0x6605, 0x6607, 0x6608,
00545 0x6609, 0x660b, 0x660d, 0x6610, 0x6611, 0x6612, 0x6616, 0x6617,
00546 0x6618, 0x661a, 0x661b, 0x661c, 0x661e, 0x6621, 0x6622, 0x6623,
00547 0x6624, 0x6626, 0x6629, 0x662a, 0x662b, 0x662c, 0x662e, 0x6630,
00548 0x6632, 0x6633, 0x6637, 0x6638, 0x6639, 0x663a, 0x663b, 0x663d,
00549 0x663f, 0x6640, 0x6642, 0x6644, 0x6645, 0x6646, 0x6647, 0x6648,
00550 0x6649, 0x664a, 0x664d, 0x664e, 0x6650, 0x6651, 0x6658, 0x6659,
00551 0x665b, 0x665c, 0x665d, 0x665e, 0x6660, 0x6662, 0x6663, 0x6665,
00552 0x6667, 0x6669, 0x666a, 0x666b, 0x666c, 0x666d, 0x6671, 0x6672,
00553 0x6673, 0x6675, 0x6678, 0x6679, 0x667b, 0x667c, 0x667d, 0x667f,
00554 0x6680, 0x6681, 0x6683, 0x6685, 0x6686, 0x6688, 0x6689, 0x668a,
00555 0x668b, 0x668d, 0x668e, 0x668f, 0x6690, 0x6692, 0x6693, 0x6694,
00556 0x6695, 0x6698, 0x6699, 0x669a, 0x669b, 0x669c, 0x669e, 0x669f,
00557 0x66a0, 0x66a1, 0x66a2, 0x66a3, 0x66a4, 0x66a5, 0x66a6, 0x66a9,
00558 0x66aa, 0x66ab, 0x66ac, 0x66ad, 0x66af, 0x66b0, 0x66b1, 0x66b2,
00559 0x66b3, 0x66b5, 0x66b6, 0x66b7, 0x66b8, 0x66ba, 0x66bb, 0x66bc,
00560 0x66bd, 0x66bf, 0x66c0, 0x66c1, 0x66c2, 0x66c3, 0x66c4, 0x66c5,
00561 0x66c6, 0x66c7, 0x66c8, 0x66c9, 0x66ca, 0x66cb, 0x66cc, 0x66cd,
00562 0x66ce, 0x66cf, 0x66d0, 0x66d1, 0x66d2, 0x66d3, 0x66d4, 0x66d5,
00563 0x66d6, 0x66d7, 0x66d8, 0x66da, 0x66de, 0x66df, 0x66e0, 0x66e1,
00564 0x66e2, 0x66e3, 0x66e4, 0x66e5, 0x66e7, 0x66e8, 0x66ea, 0x66eb,
00565 0x66ec, 0x66ed, 0x66ee, 0x66ef, 0x66f1, 0x66f5, 0x66f6, 0x66f8,
00566 0x66fa, 0x66fb, 0x66fd, 0x6701, 0x6702, 0x6703,
00567 /* 0x96 */
00568 0x6704, 0x6705, 0x6706, 0x6707, 0x670c, 0x670e, 0x670f, 0x6711,
00569 0x6712, 0x6713, 0x6716, 0x6718, 0x6719, 0x671a, 0x671c, 0x671e,
00570 0x6720, 0x6721, 0x6722, 0x6723, 0x6724, 0x6725, 0x6727, 0x6729,
00571 0x672e, 0x6730, 0x6732, 0x6733, 0x6736, 0x6737, 0x6738, 0x6739,
00572 0x673b, 0x673c, 0x673e, 0x673f, 0x6741, 0x6744, 0x6745, 0x6747,
00573 0x674a, 0x674b, 0x674d, 0x6752, 0x6754, 0x6755, 0x6757, 0x6758,
00574 0x6759, 0x675a, 0x675b, 0x675d, 0x6762, 0x6763, 0x6764, 0x6766,
00575 0x6767, 0x676b, 0x676c, 0x676e, 0x6771, 0x6774, 0x6776, 0x6778,
00576 0x6779, 0x677a, 0x677b, 0x677d, 0x6780, 0x6782, 0x6783, 0x6785,
00577 0x6786, 0x6788, 0x678a, 0x678c, 0x678d, 0x678e, 0x678f, 0x6791,
00578 0x6792, 0x6793, 0x6796, 0x6799, 0x679b, 0x679e, 0x67a0,
00579 0x67a1, 0x67a4, 0x67a6, 0x67a9, 0x67ac, 0x67ae, 0x67b1, 0x67b2,
00580 0x67b4, 0x67b9, 0x67ba, 0x67bb, 0x67bc, 0x67bd, 0x67be, 0x67bf,
00581 0x67c0, 0x67c2, 0x67c5, 0x67c6, 0x67c7, 0x67c8, 0x67c9, 0x67ca,
00582 0x67cb, 0x67cc, 0x67cd, 0x67ce, 0x67d5, 0x67d6, 0x67d7, 0x67db,
00583 0x67df, 0x67e1, 0x67e3, 0x67e4, 0x67e6, 0x67e7, 0x67e8, 0x67ea,
00584 0x67eb, 0x67ed, 0x67ee, 0x67f2, 0x67f5, 0x67f6, 0x67f7, 0x67f8,
00585 0x67f9, 0x67fa, 0x67fb, 0x67fc, 0x67fe, 0x6801, 0x6802, 0x6803,
00586 0x6804, 0x6806, 0x680d, 0x6810, 0x6812, 0x6814, 0x6815, 0x6818,
00587 0x6819, 0x681a, 0x681b, 0x681c, 0x681e, 0x681f, 0x6820, 0x6822,
00588 0x6823, 0x6824, 0x6825, 0x6826, 0x6827, 0x6828, 0x682b, 0x682c,
00589 0x682d, 0x682e, 0x682f, 0x6830, 0x6831, 0x6834, 0x6835, 0x6836,
00590 0x683a, 0x683b, 0x683f, 0x6847, 0x684b, 0x684d, 0x684f, 0x6852,
00591 0x6856, 0x6857, 0x6858, 0x6859, 0x685a, 0x685b,
00592 /* 0x97 */
00593 0x685c, 0x685d, 0x685e, 0x685f, 0x686a, 0x686c, 0x686d, 0x686e,
00594 0x686f, 0x6870, 0x6871, 0x6872, 0x6873, 0x6875, 0x6878, 0x6879,
00595 0x687a, 0x687b, 0x687c, 0x687d, 0x687e, 0x687f, 0x6880, 0x6882,
00596 0x6884, 0x6887, 0x6888, 0x6889, 0x688a, 0x688b, 0x688c, 0x688d,
00597 0x688e, 0x6890, 0x6891, 0x6892, 0x6894, 0x6895, 0x6896, 0x6898,
00598 0x6899, 0x689a, 0x689b, 0x689c, 0x689d, 0x689e, 0x689f, 0x68a0,
00599 0x68a1, 0x68a3, 0x68a4, 0x68a5, 0x68a9, 0x68aa, 0x68ab, 0x68ac,
00600 0x68ae, 0x68b1, 0x68b2, 0x68b4, 0x68b6, 0x68b7, 0x68b8, 0x68b9,
00601 0x68ba, 0x68bb, 0x68bc, 0x68bd, 0x68be, 0x68bf, 0x68c1, 0x68c3,
00602 0x68c4, 0x68c5, 0x68c6, 0x68c7, 0x68c8, 0x68ca, 0x68cc, 0x68ce,
00603 0x68cf, 0x68d0, 0x68d1, 0x68d3, 0x68d4, 0x68d6, 0x68d7, 0x68d9,
00604 0x68db, 0x68dc, 0x68dd, 0x68de, 0x68df, 0x68e1, 0x68e2, 0x68e4,
00605 0x68e5, 0x68e6, 0x68e7, 0x68e8, 0x68e9, 0x68ea, 0x68eb, 0x68ec,
00606 0x68ed, 0x68ef, 0x68f2, 0x68f3, 0x68f4, 0x68f6, 0x68f7, 0x68f8,
00607 0x68fb, 0x68fd, 0x68fe, 0x68ff, 0x6900, 0x6902, 0x6903, 0x6904,
00608 0x6906, 0x6907, 0x6908, 0x6909, 0x690a, 0x690c, 0x690f, 0x6911,
00609 0x6913, 0x6914, 0x6915, 0x6916, 0x6917, 0x6918, 0x6919, 0x691a,
00610 0x691b, 0x691c, 0x691d, 0x691e, 0x6921, 0x6922, 0x6923, 0x6925,
00611 0x6926, 0x6927, 0x6928, 0x6929, 0x692a, 0x692b, 0x692c, 0x692e,
00612 0x692f, 0x6931, 0x6932, 0x6933, 0x6935, 0x6936, 0x6937, 0x6938,
00613 0x693a, 0x693b, 0x693c, 0x693e, 0x6940, 0x6941, 0x6943, 0x6944,
00614 0x6945, 0x6946, 0x6947, 0x6948, 0x6949, 0x694a, 0x694b, 0x694c,
```

```
00615 0x694d, 0x694e, 0x694f, 0x6950, 0x6951, 0x6952, 0x6953, 0x6955,
00616 0x6956, 0x6958, 0x6959, 0x695b, 0x695c, 0x695f,
00617 /* 0x98 */
00618 0x6961, 0x6962, 0x6964, 0x6965, 0x6967, 0x6968, 0x6969, 0x696a,
00619 0x696c, 0x696d, 0x696f, 0x6970, 0x6972, 0x6973, 0x6974, 0x6975,
00620 0x6976, 0x697a, 0x697b, 0x697d, 0x697e, 0x697f, 0x6981, 0x6983,
00621 0x6985, 0x698a, 0x698b, 0x698c, 0x698e, 0x698f, 0x6990, 0x6991,
00622 0x6992, 0x6993, 0x6996, 0x6997, 0x6999, 0x699a, 0x699d, 0x699e,
00623 0x699f, 0x69a0, 0x69a1, 0x69a2, 0x69a3, 0x69a4, 0x69a5, 0x69a6,
00624 0x69a9, 0x69aa, 0x69ac, 0x69ae, 0x69af, 0x69b0, 0x69b2, 0x69b3,
00625 0x69b5, 0x69b6, 0x69b8, 0x69b9, 0x69ba, 0x69bc, 0x69bd, 0x69be,
00626 0x69bf, 0x69c0, 0x69c2, 0x69c3, 0x69c4, 0x69c5, 0x69c6, 0x69c7,
00627 0x69c8, 0x69c9, 0x69cb, 0x69cd, 0x69cf, 0x69d1, 0x69d2, 0x69d3,
00628 0x69d5, 0x69d6, 0x69d7, 0x69d8, 0x69d9, 0x69da, 0x69dc, 0x69dd,
00629 0x69de, 0x69e1, 0x69e2, 0x69e3, 0x69e4, 0x69e5, 0x69e6, 0x69e7,
00630 0x69e8, 0x69e9, 0x69ea, 0x69eb, 0x69ec, 0x69ee, 0x69ef, 0x69f0,
00631 0x69f1, 0x69f3, 0x69f4, 0x69f5, 0x69f6, 0x69f7, 0x69f8, 0x69f9,
00632 0x69fa, 0x69fb, 0x69fc, 0x69fe, 0x6a00, 0x6a01, 0x6a02, 0x6a03,
00633 0x6a04, 0x6a05, 0x6a06, 0x6a07, 0x6a08, 0x6a09, 0x6a0b, 0x6a0c,
00634 0x6a0d, 0x6a0e, 0x6a0f, 0x6a10, 0x6a11, 0x6a12, 0x6a13, 0x6a14,
00635 0x6a15, 0x6a16, 0x6a19, 0x6a1a, 0x6a1b, 0x6a1c, 0x6a1d, 0x6a1e,
00636 0x6a20, 0x6a22, 0x6a23, 0x6a24, 0x6a25, 0x6a26, 0x6a27, 0x6a29,
00637 0x6a2b, 0x6a2c, 0x6a2d, 0x6a2e, 0x6a30, 0x6a32, 0x6a33, 0x6a34,
00638 0x6a36, 0x6a37, 0x6a38, 0x6a39, 0x6a3a, 0x6a3b, 0x6a3c, 0x6a3f,
00639 0x6a40, 0x6a41, 0x6a42, 0x6a43, 0x6a45, 0x6a46, 0x6a48, 0x6a49,
00640 0x6a4a, 0x6a4b, 0x6a4c, 0x6a4d, 0x6a4e, 0x6a4f, 0x6a51, 0x6a52,
00641 0x6a53, 0x6a54, 0x6a55, 0x6a56, 0x6a57, 0x6a5a,
00642 /* 0x99 */
00643 0x6a5c, 0x6a5d, 0x6a5e, 0x6a5f, 0x6a60, 0x6a62, 0x6a63, 0x6a64,
00644 0x6a66, 0x6a67, 0x6a68, 0x6a69, 0x6a6a, 0x6a6b, 0x6a6c, 0x6a6d,
00645 0x6a6e, 0x6a6f, 0x6a70, 0x6a72, 0x6a73, 0x6a74, 0x6a75, 0x6a76,
00646 0x6a77, 0x6a78, 0x6a7a, 0x6a7b, 0x6a7d, 0x6a7e, 0x6a7f, 0x6a81,
00647 0x6a82, 0x6a83, 0x6a85, 0x6a86, 0x6a87, 0x6a88, 0x6a89, 0x6a8a,
00648 0x6a8b, 0x6a8c, 0x6a8d, 0x6a8f, 0x6a92, 0x6a93, 0x6a94, 0x6a95,
00649 0x6a96, 0x6a98, 0x6a99, 0x6a9a, 0x6a9b, 0x6a9c, 0x6a9d, 0x6a9e,
00650 0x6a9f, 0x6aa1, 0x6aa2, 0x6aa3, 0x6aa4, 0x6aa5, 0x6aa6, 0x6aa7,
00651 0x6aa8, 0x6aaa, 0x6aad, 0x6aae, 0x6aaf, 0x6ab0, 0x6ab1, 0x6ab2,
00652 0x6ab3, 0x6ab4, 0x6ab5, 0x6ab6, 0x6ab7, 0x6ab8, 0x6ab9, 0x6aba,
00653 0x6abb, 0x6abc, 0x6abd, 0x6abe, 0x6abf, 0x6ac0, 0x6ac1, 0x6ac2,
00654 0x6ac3, 0x6ac4, 0x6ac5, 0x6ac6, 0x6ac7, 0x6ac8, 0x6ac9, 0x6aca,
00655 0x6acb, 0x6acc, 0x6acd, 0x6ace, 0x6acf, 0x6ad0, 0x6ad1, 0x6ad2,
00656 0x6ad3, 0x6ad4, 0x6ad5, 0x6ad6, 0x6ad7, 0x6ad8, 0x6ad9, 0x6ada,
00657 0x6adb, 0x6adc, 0x6add, 0x6ade, 0x6adf, 0x6ae0, 0x6ae1, 0x6ae2,
00658 0x6ae3, 0x6ae4, 0x6ae5, 0x6ae6, 0x6ae7, 0x6ae8, 0x6ae9, 0x6aea,
00659 0x6aeb, 0x6aec, 0x6aed, 0x6aee, 0x6aef, 0x6af0, 0x6af1, 0x6af2,
00660 0x6af3, 0x6af4, 0x6af5, 0x6af6, 0x6af7, 0x6af8, 0x6af9, 0x6afa,
00661 0x6afb, 0x6afc, 0x6afd, 0x6afe, 0x6aff, 0x6b00, 0x6b01, 0x6b02,
00662 0x6b03, 0x6b04, 0x6b05, 0x6b06, 0x6b07, 0x6b08, 0x6b09, 0x6b0a,
00663 0x6b0b, 0x6b0c, 0x6b0d, 0x6b0e, 0x6b0f, 0x6b10, 0x6b11, 0x6b12,
00664 0x6b13, 0x6b14, 0x6b15, 0x6b16, 0x6b17, 0x6b18, 0x6b19, 0x6b1a,
00665 0x6b1b, 0x6b1c, 0x6b1d, 0x6b1e, 0x6b1f, 0x6b25, 0x6b26, 0x6b28,
00666 0x6b29, 0x6b2a, 0x6b2b, 0x6b2c, 0x6b2d, 0x6b2e,
00667 /* 0x9a */
00668 0x6b2f, 0x6b30, 0x6b31, 0x6b33, 0x6b34, 0x6b35, 0x6b36, 0x6b38,
00669 0x6b3b, 0x6b3c, 0x6b3d, 0x6b3f, 0x6b40, 0x6b41, 0x6b42, 0x6b44,
00670 0x6b45, 0x6b48, 0x6b4a, 0x6b4b, 0x6b4d, 0x6b4e, 0x6b4f, 0x6b50,
00671 0x6b51, 0x6b52, 0x6b53, 0x6b54, 0x6b55, 0x6b56, 0x6b57, 0x6b58,
00672 0x6b5a, 0x6b5b, 0x6b5c, 0x6b5d, 0x6b5e, 0x6b5f, 0x6b60, 0x6b61,
00673 0x6b68, 0x6b69, 0x6b6b, 0x6b6c, 0x6b6d, 0x6b6e, 0x6b6f, 0x6b70,
00674 0x6b71, 0x6b72, 0x6b73, 0x6b74, 0x6b75, 0x6b76, 0x6b77, 0x6b78,
00675 0x6b7a, 0x6b7d, 0x6b7e, 0x6b7f, 0x6b80, 0x6b85, 0x6b88, 0x6b8c,
00676 0x6b8e, 0x6b8f, 0x6b90, 0x6b91, 0x6b94, 0x6b95, 0x6b97, 0x6b98,
00677 0x6b99, 0x6b9c, 0x6b9d, 0x6b9e, 0x6b9f, 0x6ba0, 0x6ba2, 0x6ba3,
00678 0x6ba4, 0x6ba5, 0x6ba6, 0x6ba7, 0x6ba8, 0x6ba9, 0x6bab, 0x6bac,
00679 0x6bad, 0x6bae, 0x6baf, 0x6bb0, 0x6bb1, 0x6bb2, 0x6bb6, 0x6bb8,
00680 0x6bb9, 0x6bba, 0x6bbb, 0x6bbc, 0x6bbd, 0x6bbe, 0x6bc0, 0x6bc3,
00681 0x6bc4, 0x6bc6, 0x6bc7, 0x6bc8, 0x6bc9, 0x6bca, 0x6bcc, 0x6bce,
00682 0x6bd0, 0x6bd1, 0x6bd8, 0x6bda, 0x6bdc, 0x6bdd, 0x6bde, 0x6bdf,
00683 0x6be0, 0x6be2, 0x6be3, 0x6be4, 0x6be5, 0x6be6, 0x6be7, 0x6be8,
00684 0x6be9, 0x6bec, 0x6bed, 0x6bee, 0x6bef0, 0x6bfb, 0x6bfc, 0x6bfd,
00685 0x6bfe, 0x6bf7, 0x6bf8, 0x6bfa, 0x6bfb, 0x6bfc, 0x6bfe, 0x6bff,
00686 0x6c00, 0x6c01, 0x6c02, 0x6c03, 0x6c04, 0x6c08, 0x6c09, 0x6c0a,
00687 0x6c0b, 0x6c0c, 0x6c0e, 0x6c12, 0x6c17, 0x6c1c, 0x6c1d, 0x6c1e,
00688 0x6c20, 0x6c23, 0x6c25, 0x6c2b, 0x6c2c, 0x6c2d, 0x6c31, 0x6c33,
00689 0x6c36, 0x6c37, 0x6c39, 0x6c3a, 0x6c3b, 0x6c3c, 0x6c3e, 0x6c3f,
00690 0x6c43, 0x6c44, 0x6c45, 0x6c48, 0x6c4b, 0x6c4c, 0x6c4d, 0x6c4e,
00691 0x6c4f, 0x6c51, 0x6c52, 0x6c53, 0x6c56, 0x6c58,
00692 /* 0x9b */
00693 0x6c59, 0x6c5a, 0x6c62, 0x6c63, 0x6c65, 0x6c66, 0x6c67, 0x6c6b,
00694 0x6c6c, 0x6c6d, 0x6c6e, 0x6c6f, 0x6c71, 0x6c73, 0x6c75, 0x6c77,
00695 0x6c78, 0x6c7a, 0x6c7b, 0x6c7c, 0x6c7f, 0x6c80, 0x6c84, 0x6c87,
00696 0x6c8a, 0x6c8b, 0x6c8d, 0x6c8e, 0x6c91, 0x6c92, 0x6c95, 0x6c96,
00697 0x6c97, 0x6c98, 0x6c9a, 0x6c9c, 0x6c9d, 0x6c9e, 0x6ca0, 0x6ca2,
00698 0x6ca8, 0x6cac, 0x6caf, 0x6cb0, 0x6cb4, 0x6cb5, 0x6cb6, 0x6cb7,
00699 0x6cba, 0x6ccb, 0x6ccc, 0x6ccc, 0x6ccc, 0x6ccc, 0x6ccc, 0x6ccc,
00700 0x6ccd, 0x6ccd, 0x6cce, 0x6ccf, 0x6cd1, 0x6cd2, 0x6cd8, 0x6cd9,
00701 0x6cda, 0x6cdc, 0x6cdd, 0x6cdf, 0x6ce4, 0x6ce6, 0x6ce7, 0x6ce9,
```



```

00702 0x6cec, 0x6ced, 0x6cf2, 0x6cf4, 0x6cf9, 0x6cff, 0x6d00, 0x6d02,
00703 0x6d03, 0x6d05, 0x6d06, 0x6d08, 0x6d09, 0x6d0a, 0x6d0d, 0x6d0f,
00704 0x6d10, 0x6d11, 0x6d13, 0x6d14, 0x6d15, 0x6d16, 0x6d18, 0x6d1c,
00705 0x6d1d, 0x6d1f, 0x6d20, 0x6d21, 0x6d22, 0x6d23, 0x6d24, 0x6d26,
00706 0x6d28, 0x6d29, 0x6d2c, 0x6d2d, 0x6d2f, 0x6d30, 0x6d34, 0x6d36,
00707 0x6d37, 0x6d38, 0x6d3a, 0x6d3f, 0x6d40, 0x6d42, 0x6d44, 0x6d49,
00708 0x6d4c, 0x6d50, 0x6d55, 0x6d56, 0x6d57, 0x6d58, 0x6d5b, 0x6d5d,
00709 0x6d5f, 0x6d61, 0x6d62, 0x6d64, 0x6d65, 0x6d67, 0x6d68, 0x6d6b,
00710 0x6d6c, 0x6d6d, 0x6d70, 0x6d71, 0x6d72, 0x6d73, 0x6d75, 0x6d76,
00711 0x6d79, 0x6d7a, 0x6d7b, 0x6d7d, 0x6d7e, 0x6d7f, 0x6d80, 0x6d81,
00712 0x6d83, 0x6d84, 0x6d86, 0x6d87, 0x6d8a, 0x6d8b, 0x6d8d, 0x6d8f,
00713 0x6d90, 0x6d92, 0x6d96, 0x6d97, 0x6d98, 0x6d99, 0x6d9a, 0x6d9c,
00714 0x6da2, 0x6da5, 0x6dac, 0x6dad, 0x6db0, 0x6db1, 0x6db3, 0x6db4,
00715 0x6db6, 0x6db7, 0x6db9, 0x6dba, 0x6dbb, 0x6dbc, 0x6dbd, 0x6dbe,
00716 0x6dc1, 0x6dc2, 0x6dc3, 0x6dc8, 0x6dc9, 0x6dca,
00717 /* 0x9c */
00718 0x6dcd, 0x6dce, 0x6dcf, 0x6dd0, 0x6dd2, 0x6dd3, 0x6dd4, 0x6dd5,
00719 0x6dd7, 0x6dda, 0x6ddb, 0x6ddc, 0x6ddf, 0x6de2, 0x6de3, 0x6de5,
00720 0x6de7, 0x6de8, 0x6de9, 0x6dea, 0x6ded, 0x6def, 0x6df0, 0x6df2,
00721 0x6df4, 0x6df5, 0x6df6, 0x6df8, 0x6dfa, 0x6dfd, 0x6dfe, 0x6dff,
00722 0x6e00, 0x6e01, 0x6e02, 0x6e03, 0x6e04, 0x6e06, 0x6e07, 0x6e08,
00723 0x6e09, 0x6e0b, 0x6e0f, 0x6e12, 0x6e13, 0x6e15, 0x6e18, 0x6e19,
00724 0x6e1b, 0x6e1c, 0x6e1e, 0x6e1f, 0x6e22, 0x6e26, 0x6e27, 0x6e28,
00725 0x6e2a, 0x6e2c, 0x6e2e, 0x6e30, 0x6e31, 0x6e33, 0x6e35, 0x6e36,
00726 0x6e37, 0x6e39, 0x6e3b, 0x6e3c, 0x6e3d, 0x6e3e, 0x6e3f, 0x6e40,
00727 0x6e41, 0x6e42, 0x6e45, 0x6e46, 0x6e47, 0x6e48, 0x6e49, 0x6e4a,
00728 0x6e4b, 0x6e4c, 0x6e4f, 0x6e50, 0x6e51, 0x6e52, 0x6e55, 0x6e57,
00729 0x6e59, 0x6e5a, 0x6e5c, 0x6e5d, 0x6e5e, 0x6e60, 0x6e61, 0x6e62,
00730 0x6e63, 0x6e64, 0x6e65, 0x6e66, 0x6e67, 0x6e68, 0x6e69, 0x6e6a,
00731 0x6e6c, 0x6e6d, 0x6e6f, 0x6e70, 0x6e71, 0x6e72, 0x6e73, 0x6e74,
00732 0x6e75, 0x6e76, 0x6e77, 0x6e78, 0x6e79, 0x6e7a, 0x6e7b, 0x6e7c,
00733 0x6e7d, 0x6e80, 0x6e81, 0x6e82, 0x6e84, 0x6e87, 0x6e88, 0x6e8a,
00734 0x6e8b, 0x6e8c, 0x6e8d, 0x6e8e, 0x6e91, 0x6e92, 0x6e93, 0x6e94,
00735 0x6e95, 0x6e96, 0x6e97, 0x6e99, 0x6e9a, 0x6e9b, 0x6e9d, 0x6e9e,
00736 0x6ea0, 0x6ea1, 0x6ea3, 0x6ea4, 0x6ea6, 0x6ea8, 0x6ea9, 0x6eab,
00737 0x6eac, 0x6ead, 0x6eae, 0x6eb0, 0x6eb3, 0x6eb5, 0x6eb8, 0x6eb9,
00738 0x6ebc, 0x6ebe, 0x6ebf, 0x6ec0, 0x6ec3, 0x6ec4, 0x6ec5, 0x6ec6,
00739 0x6ec8, 0x6ec9, 0x6eca, 0x6ecc, 0x6ecd, 0x6ece, 0x6ed0, 0x6ed2,
00740 0x6ed6, 0x6ed8, 0x6ed9, 0x6edb, 0x6edc, 0x6edd, 0x6ee3, 0x6ee7,
00741 0x6eea, 0x6eeb, 0x6eec, 0x6eed, 0x6eee, 0x6eef,
00742 /* 0x9d */
00743 0x6ef0, 0x6ef1, 0x6ef2, 0x6ef3, 0x6ef5, 0x6ef6, 0x6ef7, 0x6ef8,
00744 0x6efa, 0x6efb, 0x6efc, 0x6efd, 0x6efe, 0x6eff, 0x6f00, 0x6f01,
00745 0x6f03, 0x6f04, 0x6f05, 0x6f07, 0x6f08, 0x6f0a, 0x6f0b, 0x6f0c,
00746 0x6f0d, 0x6f0e, 0x6f10, 0x6f11, 0x6f12, 0x6f16, 0x6f17, 0x6f18,
00747 0x6f19, 0x6f1a, 0x6f1b, 0x6f1c, 0x6f1d, 0x6f1e, 0x6f1f, 0x6f21,
00748 0x6f22, 0x6f23, 0x6f25, 0x6f26, 0x6f27, 0x6f28, 0x6f2c, 0x6f2e,
00749 0x6f30, 0x6f32, 0x6f34, 0x6f35, 0x6f37, 0x6f38, 0x6f39, 0x6f3a,
00750 0x6f3b, 0x6f3c, 0x6f3d, 0x6f3f, 0x6f40, 0x6f41, 0x6f42, 0x6f43,
00751 0x6f44, 0x6f45, 0x6f48, 0x6f49, 0x6f4a, 0x6f4c, 0x6f4e, 0x6f4f,
00752 0x6f50, 0x6f51, 0x6f52, 0x6f53, 0x6f54, 0x6f55, 0x6f56, 0x6f57,
00753 0x6f59, 0x6f5a, 0x6f5b, 0x6f5d, 0x6f5f, 0x6f60, 0x6f61, 0x6f63,
00754 0x6f64, 0x6f65, 0x6f67, 0x6f68, 0x6f69, 0x6f6a, 0x6f6b, 0x6f6c,
00755 0x6f6f, 0x6f70, 0x6f71, 0x6f73, 0x6f75, 0x6f76, 0x6f77, 0x6f79,
00756 0x6f7b, 0x6f7d, 0x6f7e, 0x6f7f, 0x6f80, 0x6f81, 0x6f82, 0x6f83,
00757 0x6f85, 0x6f86, 0x6f87, 0x6f8a, 0x6f8b, 0x6f8f, 0x6f90, 0x6f91,
00758 0x6f92, 0x6f93, 0x6f94, 0x6f95, 0x6f96, 0x6f97, 0x6f98, 0x6f99,
00759 0x6f9a, 0x6f9b, 0x6f9d, 0x6f9e, 0x6f9f, 0x6fa0, 0x6fa2, 0x6fa3,
00760 0x6fa4, 0x6fa5, 0x6fa6, 0x6fa8, 0x6fa9, 0x6faa, 0x6fab, 0x6fac,
00761 0x6fad, 0x6fae, 0x6faf, 0x6fb0, 0x6fb1, 0x6fb2, 0x6fb4, 0x6fb5,
00762 0x6fb7, 0x6fb8, 0x6fba, 0x6fbb, 0x6fbc, 0x6fbd, 0x6fbe, 0x6fbf,
00763 0x6fc1, 0x6fc3, 0x6fc4, 0x6fc5, 0x6fc6, 0x6fc7, 0x6fc8, 0x6fca,
00764 0x6fcb, 0x6fcc, 0x6fcd, 0x6fce, 0x6fcf, 0x6fd0, 0x6fd3, 0x6fd4,
00765 0x6fd5, 0x6fd6, 0x6fd7, 0x6fd8, 0x6fd9, 0x6fda, 0x6fdb, 0x6fdc,
00766 0x6fdd, 0x6fdf, 0x6fe2, 0x6fe3, 0x6fe4, 0x6fe5,
00767 /* 0x9e */
00768 0x6fe6, 0x6fe7, 0x6fe8, 0x6fe9, 0x6fea, 0x6feb, 0x6fec, 0x6fed,
00769 0x6ff0, 0x6ff1, 0x6ff2, 0x6ff3, 0x6ff4, 0x6ff5, 0x6ff6, 0x6ff7,
00770 0x6ff8, 0x6ff9, 0x6ffa, 0x6ffb, 0x6ffc, 0x6ffd, 0x6ffe, 0x6fff,
00771 0x7000, 0x7001, 0x7002, 0x7003, 0x7004, 0x7005, 0x7006, 0x7007,
00772 0x7008, 0x7009, 0x700a, 0x700b, 0x700c, 0x700d, 0x700e, 0x700f,
00773 0x7010, 0x7012, 0x7013, 0x7014, 0x7015, 0x7016, 0x7017, 0x7018,
00774 0x7019, 0x701c, 0x701d, 0x701e, 0x701f, 0x7020, 0x7021, 0x7022,
00775 0x7024, 0x7025, 0x7026, 0x7027, 0x7028, 0x7029, 0x702a, 0x702b,
00776 0x702c, 0x702d, 0x702e, 0x702f, 0x7030, 0x7031, 0x7032, 0x7033,
00777 0x7034, 0x7036, 0x7037, 0x7038, 0x703a, 0x703b, 0x703c, 0x703d,
00778 0x703e, 0x703f, 0x7040, 0x7041, 0x7042, 0x7043, 0x7044, 0x7045,
00779 0x7046, 0x7047, 0x7048, 0x7049, 0x704a, 0x704b, 0x704d, 0x704e,
00780 0x7050, 0x7051, 0x7052, 0x7053, 0x7054, 0x7055, 0x7056, 0x7057,
00781 0x7058, 0x7059, 0x705a, 0x705b, 0x705c, 0x705d, 0x705f, 0x7060,
00782 0x7061, 0x7062, 0x7063, 0x7064, 0x7065, 0x7066, 0x7067, 0x7068,
00783 0x7069, 0x706a, 0x706e, 0x7071, 0x7072, 0x7073, 0x7074, 0x7077,
00784 0x7079, 0x707a, 0x707b, 0x707d, 0x7081, 0x7082, 0x7083, 0x7084,
00785 0x7086, 0x7087, 0x7088, 0x708b, 0x708c, 0x708d, 0x708f, 0x7090,
00786 0x7091, 0x7093, 0x7097, 0x7098, 0x709a, 0x709b, 0x709e, 0x709f,
00787 0x70a0, 0x70a1, 0x70a2, 0x70a3, 0x70a4, 0x70a5, 0x70a6, 0x70a7,
00788 0x70a8, 0x70a9, 0x70aa, 0x70ab, 0x70b0, 0x70b2, 0x70b4, 0x70b5, 0x70b6,

```



```
00789 0x70ba, 0x70be, 0x70bf, 0x70c4, 0x70c5, 0x70c6, 0x70c7, 0x70c9,
00790 0x70cb, 0x70cc, 0x70cd, 0x70ce, 0x70cf, 0x70d0, 0x70d1, 0x70d2,
00791 0x70d3, 0x70d4, 0x70d5, 0x70d6, 0x70d7, 0x70da,
00792 /* 0x9f */
00793 0x70dc, 0x70dd, 0x70de, 0x70e0, 0x70e1, 0x70e2, 0x70e3, 0x70e5,
00794 0x70ea, 0x70ee, 0x70f0, 0x70f1, 0x70f2, 0x70f3, 0x70f4, 0x70f5,
00795 0x70f6, 0x70f8, 0x70fa, 0x70fb, 0x70fc, 0x70fe, 0x70ff, 0x7100,
00796 0x7101, 0x7102, 0x7103, 0x7104, 0x7105, 0x7106, 0x7107, 0x7108,
00797 0x710b, 0x710c, 0x710d, 0x710e, 0x710f, 0x7111, 0x7112, 0x7114,
00798 0x7117, 0x711b, 0x711c, 0x711d, 0x711e, 0x711f, 0x7120, 0x7121,
00799 0x7122, 0x7123, 0x7124, 0x7125, 0x7127, 0x7128, 0x7129, 0x712a,
08000 0x712b, 0x712c, 0x712d, 0x712e, 0x7132, 0x7133, 0x7134, 0x7135,
08001 0x7137, 0x7138, 0x7139, 0x713a, 0x713b, 0x713c, 0x713d, 0x713e,
08002 0x713f, 0x7140, 0x7141, 0x7142, 0x7143, 0x7144, 0x7146, 0x7147,
08003 0x7148, 0x7149, 0x714a, 0x714b, 0x714d, 0x714f, 0x7150, 0x7151, 0x7152,
08004 0x7153, 0x7154, 0x7155, 0x7156, 0x7157, 0x7158, 0x7159, 0x715a,
08005 0x715b, 0x715d, 0x715f, 0x7160, 0x7161, 0x7162, 0x7163, 0x7165,
08006 0x7169, 0x716a, 0x716b, 0x716c, 0x716d, 0x716f, 0x7170, 0x7171,
08007 0x7174, 0x7175, 0x7176, 0x7177, 0x7179, 0x717b, 0x717c, 0x717e,
08008 0x717f, 0x7180, 0x7181, 0x7182, 0x7183, 0x7185, 0x7186, 0x7187,
08009 0x7188, 0x7189, 0x718b, 0x718c, 0x718d, 0x718e, 0x7190, 0x7191,
08010 0x7192, 0x7193, 0x7195, 0x7196, 0x7197, 0x719a, 0x719b, 0x719c,
08011 0x719d, 0x719e, 0x71a1, 0x71a2, 0x71a3, 0x71a4, 0x71a5, 0x71a6,
08012 0x71a7, 0x71a9, 0x71aa, 0x71ab, 0x71ad, 0x71ae, 0x71af, 0x71b0,
08013 0x71b1, 0x71b2, 0x71b4, 0x71b6, 0x71b7, 0x71b8, 0x71ba, 0x71bb,
08014 0x71bc, 0x71bd, 0x71be, 0x71bf, 0x71c0, 0x71c1, 0x71c2, 0x71c4,
08015 0x71c5, 0x71c6, 0x71c7, 0x71c8, 0x71c9, 0x71ca, 0x71cb, 0x71cc,
08016 0x71cd, 0x71cf, 0x71d0, 0x71d1, 0x71d2, 0x71d3,
08017 /* 0xa0 */
08018 0x71d6, 0x71d7, 0x71d8, 0x71d9, 0x71da, 0x71db, 0x71dc, 0x71dd,
08019 0x71de, 0x71df, 0x71e1, 0x71e2, 0x71e3, 0x71e4, 0x71e6, 0x71e8,
08020 0x71e9, 0x71ea, 0x71eb, 0x71ec, 0x71ed, 0x71ef, 0x71f0, 0x71f1,
08021 0x71f2, 0x71f3, 0x71f4, 0x71f5, 0x71f6, 0x71f7, 0x71f8, 0x71fa,
08022 0x71fb, 0x71fc, 0x71fd, 0x71fe, 0x71ff, 0x7200, 0x7201, 0x7202,
08023 0x7203, 0x7204, 0x7205, 0x7207, 0x7208, 0x7209, 0x720a, 0x720b,
08024 0x720c, 0x720d, 0x720e, 0x720f, 0x7210, 0x7211, 0x7212, 0x7213,
08025 0x7214, 0x7215, 0x7216, 0x7217, 0x7218, 0x7219, 0x721a, 0x721b,
08026 0x721c, 0x721e, 0x721f, 0x7220, 0x7221, 0x7222, 0x7223, 0x7224,
08027 0x7225, 0x7226, 0x7227, 0x7229, 0x722b, 0x722d, 0x722e, 0x722f,
08028 0x7232, 0x7233, 0x7234, 0x723a, 0x723c, 0x723e, 0x7240, 0x7241,
08029 0x7242, 0x7243, 0x7244, 0x7245, 0x7246, 0x7249, 0x724a, 0x724b,
08030 0x724e, 0x724f, 0x7250, 0x7251, 0x7253, 0x7254, 0x7255, 0x7257,
08031 0x7258, 0x725a, 0x725c, 0x725e, 0x7260, 0x7263, 0x7264, 0x7265,
08032 0x7268, 0x726a, 0x726b, 0x726c, 0x726d, 0x7270, 0x7271, 0x7273,
08033 0x7274, 0x7276, 0x7278, 0x727b, 0x727c, 0x727d, 0x7282,
08034 0x7283, 0x7285, 0x7286, 0x7287, 0x7288, 0x7289, 0x728c, 0x728e,
08035 0x7290, 0x7291, 0x7293, 0x7294, 0x7295, 0x7296, 0x7297, 0x7298,
08036 0x7299, 0x729a, 0x729b, 0x729c, 0x729d, 0x729e, 0x72a0, 0x72a1,
08037 0x72a2, 0x72a3, 0x72a4, 0x72a5, 0x72a6, 0x72a7, 0x72a8, 0x72a9,
08038 0x72aa, 0x72ab, 0x72ae, 0x72b1, 0x72b2, 0x72b3, 0x72b5, 0x72ba,
08039 0x72bb, 0x72bd, 0x72be, 0x72bf, 0x72c0, 0x72c5, 0x72c6,
08040 0x72c7, 0x72c9, 0x72ca, 0x72cb, 0x72cc, 0x72cf, 0x72d1, 0x72d3,
08041 0x72d4, 0x72d5, 0x72d6, 0x72d8, 0x72da, 0x72db,
08042 /* 0xa1 */
08043 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08044 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08045 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08046 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08047 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08048 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08049 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08050 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08051 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08052 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08053 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08054 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08055 0x3000, 0x3001, 0x3002, 0x00b7, 0x02c9, 0x02c7, 0x00a8, 0x3003,
08056 0x3005, 0x2014, 0xff5e, 0x2016, 0x2026, 0x2018, 0x2019, 0x201c,
08057 0x201d, 0x3014, 0x3015, 0x3008, 0x3009, 0x300a, 0x300b, 0x300c,
08058 0x300d, 0x300e, 0x300f, 0x3016, 0x3017, 0x3010, 0x3011, 0x00b1,
08059 0x00d7, 0x00f7, 0x2236, 0x2227, 0x2211, 0x220f, 0x222a,
08060 0x2229, 0x2208, 0x2237, 0x221a, 0x22a5, 0x2225, 0x2220, 0x2312,
08061 0x2299, 0x222b, 0x222e, 0x2261, 0x224c, 0x2248, 0x223d, 0x221d,
08062 0x2260, 0x226e, 0x226f, 0x2264, 0x2265, 0x221e, 0x2235, 0x2234,
08063 0x2642, 0x2640, 0x00b0, 0x2032, 0x2033, 0x2103, 0xff04, 0x00a4,
08064 0xffe0, 0xffe1, 0x2030, 0x00a7, 0x2116, 0x2606, 0x2605, 0x25cb,
08065 0x25cf, 0x25ce, 0x25c7, 0x25c6, 0x25a1, 0x25a0, 0x25b3, 0x25b2,
08066 0x203b, 0x2192, 0x2190, 0x2191, 0x2193, 0x3013,
08067 /* 0xa2 */
08068 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08069 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08070 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08071 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08072 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08073 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08074 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
08075 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
```

```

00876 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00877 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00878 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00879 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00880 0x2170, 0x2171, 0x2172, 0x2173, 0x2174, 0x2175, 0x2176, 0x2177,
00881 0x2178, 0x2179, 0x2179, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00882 0x2488, 0x2489, 0x248a, 0x248b, 0x248c, 0x248d, 0x248e, 0x248f,
00883 0x2490, 0x2491, 0x2492, 0x2493, 0x2494, 0x2495, 0x2496, 0x2497,
00884 0x2498, 0x2499, 0x249a, 0x249b, 0x2474, 0x2475, 0x2476, 0x2477,
00885 0x2478, 0x2479, 0x247a, 0x247b, 0x247c, 0x247d, 0x247e, 0x247f,
00886 0x2480, 0x2481, 0x2482, 0x2483, 0x2484, 0x2485, 0x2486, 0x2487,
00887 0x2460, 0x2461, 0x2462, 0x2463, 0x2464, 0x2465, 0x2466, 0x2467,
00888 0x2468, 0x2469, 0xffffd, 0xffffd, 0x3220, 0x3221, 0x3222, 0x3223,
00889 0x3224, 0x3225, 0x3226, 0x3227, 0x3228, 0x3229, 0xffffd, 0xffffd,
00890 0x2160, 0x2161, 0x2162, 0x2163, 0x2164, 0x2165, 0x2166, 0x2167,
00891 0x2168, 0x2169, 0x216a, 0x216b, 0xffffd, 0xffffd,
00892 /* 0xa3 */
00893 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00894 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00895 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00896 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00897 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00898 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00899 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00900 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00901 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00902 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00903 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00904 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00905 0xff01, 0xff02, 0xff03, 0xff05, 0xff05, 0xff06, 0xff07, 0xff08,
00906 0xff09, 0xff0a, 0xff0b, 0xff0c, 0xff0d, 0xff0e, 0xff0f, 0xff10,
00907 0xff11, 0xff12, 0xff13, 0xff14, 0xff15, 0xff16, 0xff17, 0xff18,
00908 0xff19, 0xff1a, 0xff1b, 0xff1c, 0xff1d, 0xff1e, 0xff1f, 0xff20,
00909 0xff21, 0xff22, 0xff23, 0xff24, 0xff25, 0xff26, 0xff27, 0xff28,
00910 0xff29, 0xff2a, 0xff2b, 0xff2c, 0xff2d, 0xff2e, 0xff2f, 0xff30,
00911 0xff31, 0xff32, 0xff33, 0xff34, 0xff35, 0xff36, 0xff37, 0xff38,
00912 0xff39, 0xff3a, 0xff3b, 0xff3c, 0xff3d, 0xff3e, 0xff3f, 0xff40,
00913 0xff41, 0xff42, 0xff43, 0xff44, 0xff45, 0xff46, 0xff47, 0xff48,
00914 0xff49, 0xff4a, 0xff4b, 0xff4c, 0xff4d, 0xff4e, 0xff4f, 0xff50,
00915 0xff51, 0xff52, 0xff53, 0xff54, 0xff55, 0xff56, 0xff57, 0xff58,
00916 0xff59, 0xff5a, 0xff5b, 0xff5c, 0xff5d, 0xff5e,
00917 /* 0xa4 */
00918 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00919 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00920 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00921 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00922 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00923 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00924 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00925 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00926 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00927 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00928 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00929 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00930 0x3041, 0x3042, 0x3043, 0x3044, 0x3045, 0x3046, 0x3047, 0x3048,
00931 0x3049, 0x304a, 0x304b, 0x304c, 0x304d, 0x304e, 0x304f, 0x3050,
00932 0x3051, 0x3052, 0x3053, 0x3054, 0x3055, 0x3056, 0x3057, 0x3058,
00933 0x3059, 0x305a, 0x305b, 0x305c, 0x305d, 0x305e, 0x305f, 0x3060,
00934 0x3061, 0x3062, 0x3063, 0x3064, 0x3065, 0x3066, 0x3067, 0x3068,
00935 0x3069, 0x306a, 0x306b, 0x306c, 0x306d, 0x306e, 0x306f, 0x3070,
00936 0x3071, 0x3072, 0x3073, 0x3074, 0x3075, 0x3076, 0x3077, 0x3078,
00937 0x3079, 0x307a, 0x307b, 0x307c, 0x307d, 0x307e, 0x307f, 0x3080,
00938 0x3081, 0x3082, 0x3083, 0x3084, 0x3085, 0x3086, 0x3087, 0x3088,
00939 0x3089, 0x308a, 0x308b, 0x308c, 0x308d, 0x308e, 0x308f, 0x3090,
00940 0x3091, 0x3092, 0x3093, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00941 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00942 /* 0xa5 */
00943 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00944 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00945 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00946 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00947 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00948 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00949 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00950 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00951 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00952 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00953 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00954 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00955 0x30a1, 0x30a2, 0x30a3, 0x30a4, 0x30a5, 0x30a6, 0x30a7, 0x30a8,
00956 0x30a9, 0x30aa, 0x30ab, 0x30ac, 0x30ad, 0x30ae, 0x30af, 0x30b0,
00957 0x30b1, 0x30b2, 0x30b3, 0x30b4, 0x30b5, 0x30b6, 0x30b7, 0x30b8,
00958 0x30b9, 0x30ba, 0x30bb, 0x30bc, 0x30bd, 0x30be, 0x30bf, 0x30c0,
00959 0x30c1, 0x30c2, 0x30c3, 0x30c4, 0x30c5, 0x30c6, 0x30c7, 0x30c8,
00960 0x30c9, 0x30ca, 0x30cb, 0x30cc, 0x30cd, 0x30ce, 0x30cf, 0x30d0,
00961 0x30d1, 0x30d2, 0x30d3, 0x30d4, 0x30d5, 0x30d6, 0x30d7, 0x30d8,
00962 0x30d9, 0x30da, 0x30db, 0x30dc, 0x30dd, 0x30de, 0x30df, 0x30e0,

```

```
00963 0x30e1, 0x30e2, 0x30e3, 0x30e4, 0x30e5, 0x30e6, 0x30e7, 0x30e8,
00964 0x30e9, 0x30ea, 0x30eb, 0x30ec, 0x30ed, 0x30ee, 0x30ef, 0x30f0,
00965 0x30f1, 0x30f2, 0x30f3, 0x30f4, 0x30f5, 0x30f6, 0xffff, 0xffff,
00966 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00967 /* 0xa6 */
00968 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00969 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00970 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00971 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00972 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00973 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00974 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00975 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00976 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00977 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00978 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00979 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00980 0x0391, 0x0392, 0x0393, 0x0394, 0x0395, 0x0396, 0x0397, 0x0398,
00981 0x0399, 0x039a, 0x039b, 0x039c, 0x039d, 0x039e, 0x039f, 0x03a0,
00982 0x03a1, 0x03a2, 0x03a3, 0x03a4, 0x03a5, 0x03a6, 0x03a7, 0x03a8, 0x03a9,
00983 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00984 0x03b1, 0x03b2, 0x03b3, 0x03b4, 0x03b5, 0x03b6, 0x03b7, 0x03b8,
00985 0x03b9, 0x03ba, 0x03bb, 0x03bc, 0x03bd, 0x03be, 0x03bf, 0x03c0,
00986 0x03c1, 0x03c2, 0x03c3, 0x03c4, 0x03c5, 0x03c6, 0x03c7, 0x03c8,
00987 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00988 0xfe36, 0xfe37, 0xfe38, 0xfe39, 0xfe3a, 0xfe3b, 0xfe3c, 0xfe3d,
00989 0xfe3e, 0xfe3f, 0xfe40, 0xfe41, 0xfe42, 0xfe43, 0xfe44, 0xfe45,
00990 0xfe46, 0xfe47, 0xfe48, 0xfe49, 0xfe4a, 0xfe4b, 0xfe4c, 0xfe4d,
00991 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00992 /* 0xa7 */
00993 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00994 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00995 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00996 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00997 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00998 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00999 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01000 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01001 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01002 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01003 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01004 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01005 0x0410, 0x0411, 0x0412, 0x0413, 0x0414, 0x0415, 0x0416, 0x0417,
01006 0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e, 0x041f,
01007 0x0420, 0x0421, 0x0422, 0x0423, 0x0424, 0x0425, 0x0426, 0x0427,
01008 0x0428, 0x0429, 0x042a, 0x042b, 0x042c, 0x042d, 0x042e, 0x042f,
01009 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01010 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01011 0x0430, 0x0431, 0x0432, 0x0433, 0x0434, 0x0435, 0x0436, 0x0437,
01012 0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e, 0x043f,
01013 0x0440, 0x0441, 0x0442, 0x0443, 0x0444, 0x0445, 0x0446, 0x0447,
01014 0x0448, 0x0449, 0x044a, 0x044b, 0x044c, 0x044d, 0x044e, 0x044f,
01015 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01016 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01017 /* 0xa8 */
01018 0x02ca, 0x02cb, 0x02cc, 0x02cd, 0x02ce, 0x02cf, 0x02d0, 0x02d1,
01019 0x02d2, 0x02d3, 0x02d4, 0x02d5, 0x02d6, 0x02d7, 0x02d8, 0x02d9,
01020 0x02da, 0x02db, 0x02dc, 0x02dd, 0x02de, 0x02df, 0x02e0, 0x02e1,
01021 0x02e2, 0x02e3, 0x02e4, 0x02e5, 0x02e6, 0x02e7, 0x02e8, 0x02e9,
01022 0x02ea, 0x02eb, 0x02ec, 0x02ed, 0x02ee, 0x02ef, 0x02f0, 0x02f1,
01023 0x02f2, 0x02f3, 0x02f4, 0x02f5, 0x02f6, 0x02f7, 0x02f8, 0x02f9,
01024 0x02fa, 0x02fb, 0x02fc, 0x02fd, 0x02fe, 0x02ff, 0x0300, 0x0301,
01025 0x0302, 0x0303, 0x0304, 0x0305, 0x0306, 0x0307, 0x0308, 0x0309,
01026 0x030a, 0x030b, 0x030c, 0x030d, 0x030e, 0x030f, 0x0310, 0x0311,
01027 0x0312, 0x0313, 0x0314, 0x0315, 0x0316, 0x0317, 0x0318, 0x0319,
01028 0x031a, 0x031b, 0x031c, 0x031d, 0x031e, 0x031f, 0x0320, 0x0321,
01029 0x0322, 0x0323, 0x0324, 0x0325, 0x0326, 0x0327, 0x0328, 0x0329,
01030 0x032a, 0x032b, 0x032c, 0x032d, 0x032e, 0x032f, 0x0330, 0x0331,
01031 0x0332, 0x0333, 0x0334, 0x0335, 0x0336, 0x0337, 0x0338, 0x0339,
01032 0x033a, 0x033b, 0x033c, 0x033d, 0x033e, 0x033f, 0x0340, 0x0341,
01033 0x0342, 0x0343, 0x0344, 0x0345, 0x0346, 0x0347, 0x0348, 0x0349,
01034 0x034a, 0x034b, 0x034c, 0x034d, 0x034e, 0x034f, 0x0350, 0x0351,
01035 0x0352, 0x0353, 0x0354, 0x0355, 0x0356, 0x0357, 0x0358, 0x0359,
01036 0x035a, 0x035b, 0x035c, 0x035d, 0x035e, 0x035f, 0x0360, 0x0361,
01037 0x0362, 0x0363, 0x0364, 0x0365, 0x0366, 0x0367, 0x0368, 0x0369,
01038 0x036a, 0x036b, 0x036c, 0x036d, 0x036e, 0x036f, 0x0370, 0x0371,
01039 0x0372, 0x0373, 0x0374, 0x0375, 0x0376, 0x0377, 0x0378, 0x0379,
01040 0x037a, 0x037b, 0x037c, 0x037d, 0x037e, 0x037f, 0x0380, 0x0381,
01041 0x0382, 0x0383, 0x0384, 0x0385, 0x0386, 0x0387, 0x0388, 0x0389,
01042 /* 0xa9 */
01043 0x0301, 0x0302, 0x0303, 0x0304, 0x0305, 0x0306, 0x0307, 0x0308,
01044 0x0309, 0x030a, 0x030b, 0x030c, 0x030d, 0x030e, 0x030f, 0x0310,
01045 0x0311, 0x0312, 0x0313, 0x0314, 0x0315, 0x0316, 0x0317, 0x0318,
01046 0x0319, 0x031a, 0x031b, 0x031c, 0x031d, 0x031e, 0x031f, 0x0320,
01047 0x0321, 0x0322, 0x0323, 0x0324, 0x0325, 0x0326, 0x0327, 0x0328,
01048 0x0329, 0x032a, 0x032b, 0x032c, 0x032d, 0x032e, 0x032f, 0x0330,
01049 0x0331, 0x0332, 0x0333, 0x0334, 0x0335, 0x0336, 0x0337, 0x0338,
```

```

01050 0xfe5b, 0xfe5c, 0xfe5d, 0xfe5e, 0xfe5f, 0xfe60, 0xfe61, 0xfe62,
01051 0xfe63, 0xfe64, 0xfe65, 0xfe66, 0xfe68, 0xfe69, 0xfe6a, 0xfe6b,
01052 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01053 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x3007, 0xffff, 0xffff,
01054 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01055 0xffff, 0xffff, 0xffff, 0xffff, 0x2500, 0x2501, 0x2502, 0x2503, 0x2504,
01056 0x2505, 0x2506, 0x2507, 0x2508, 0x2509, 0x250a, 0x250b, 0x250c,
01057 0x250d, 0x250e, 0x250f, 0x2510, 0x2511, 0x2512, 0x2513, 0x2514,
01058 0x2515, 0x2516, 0x2517, 0x2518, 0x2519, 0x251a, 0x251b, 0x251c,
01059 0x251d, 0x251e, 0x251f, 0x2520, 0x2521, 0x2522, 0x2523, 0x2524,
01060 0x2525, 0x2526, 0x2527, 0x2528, 0x2529, 0x252a, 0x252b, 0x252c,
01061 0x252d, 0x252e, 0x252f, 0x2530, 0x2531, 0x2532, 0x2533, 0x2534,
01062 0x2535, 0x2536, 0x2537, 0x2538, 0x2539, 0x253a, 0x253b, 0x253c,
01063 0x253d, 0x253e, 0x253f, 0x2540, 0x2541, 0x2542, 0x2543, 0x2544,
01064 0x2545, 0x2546, 0x2547, 0x2548, 0x2549, 0x254a, 0x254b, 0xffff,
01065 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01066 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01067 /* 0xaa */
01068 0x72dc, 0x72dd, 0x72de, 0x72e2, 0x72e3, 0x72e4, 0x72e5, 0x72e6,
01069 0x72e7, 0x72e8, 0x72eb, 0x72ef, 0x72f6, 0x72f9, 0x72fd, 0x72fe,
01070 0x72ff, 0x7300, 0x7302, 0x7304, 0x7305, 0x7306, 0x7307, 0x7308,
01071 0x7309, 0x730b, 0x730c, 0x730d, 0x730f, 0x7310, 0x7311, 0x7312,
01072 0x7314, 0x7318, 0x7319, 0x731a, 0x731f, 0x7320, 0x7323, 0x7324,
01073 0x7326, 0x7327, 0x7328, 0x732d, 0x732f, 0x7330, 0x7332, 0x7333,
01074 0x7335, 0x7336, 0x733a, 0x733b, 0x733c, 0x733d, 0x7340, 0x7341,
01075 0x7342, 0x7343, 0x7344, 0x7345, 0x7346, 0x7347, 0x7348, 0x7349,
01076 0x734a, 0x734b, 0x734c, 0x734e, 0x734f, 0x7351, 0x7353, 0x7354,
01077 0x7355, 0x7356, 0x7358, 0x7359, 0x735a, 0x735b, 0x735c, 0x735d,
01078 0x735e, 0x735f, 0x7361, 0x7362, 0x7363, 0x7364, 0x7365, 0x7366,
01079 0x7367, 0x7368, 0x7369, 0x736a, 0x736b, 0x736e, 0x7370, 0x7371,
01080 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01081 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01082 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01083 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01084 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01085 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01086 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01087 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01088 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01089 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01090 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01091 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01092 /* 0xab */
01093 0x7372, 0x7373, 0x7374, 0x7375, 0x7376, 0x7377, 0x7378, 0x7379,
01094 0x737a, 0x737b, 0x737c, 0x737d, 0x737e, 0x7380, 0x7381, 0x7382,
01095 0x7383, 0x7385, 0x7386, 0x7388, 0x738a, 0x738c, 0x738d, 0x738f,
01096 0x7390, 0x7392, 0x7393, 0x7394, 0x7395, 0x7397, 0x7398, 0x7399,
01097 0x739a, 0x739c, 0x739d, 0x739e, 0x73a0, 0x73a1, 0x73a3, 0x73a4,
01098 0x73a5, 0x73a6, 0x73a7, 0x73a8, 0x73aa, 0x73ac, 0x73ad, 0x73b1,
01099 0x73b4, 0x73b5, 0x73b6, 0x73b8, 0x73b9, 0x73bc, 0x73bd, 0x73be,
01100 0x73bf, 0x73c1, 0x73c3, 0x73c4, 0x73c5, 0x73c6, 0x73c7, 0x73cb,
01101 0x73cc, 0x73ce, 0x73d2, 0x73d3, 0x73d4, 0x73d5, 0x73d6, 0x73d7,
01102 0x73d8, 0x73da, 0x73db, 0x73dc, 0x73dd, 0x73df, 0x73e1, 0x73e2,
01103 0x73e3, 0x73e4, 0x73e6, 0x73e8, 0x73ea, 0x73eb, 0x73ec, 0x73ee,
01104 0x73ef, 0x73f0, 0x73f1, 0x73f3, 0x73f4, 0x73f5, 0x73f6, 0x73f7,
01105 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01106 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01107 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01108 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01109 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01110 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01111 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01112 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01113 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01114 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01115 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01116 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01117 /* 0xac */
01118 0x73f8, 0x73f9, 0x73fa, 0x73fb, 0x73fc, 0x73fd, 0x73fe, 0x73ff,
01119 0x7400, 0x7401, 0x7402, 0x7404, 0x7407, 0x7408, 0x740b, 0x740c,
01120 0x740d, 0x740e, 0x7411, 0x7412, 0x7413, 0x7414, 0x7415, 0x7416,
01121 0x7417, 0x7418, 0x7419, 0x741c, 0x741d, 0x741e, 0x741f, 0x7420,
01122 0x7421, 0x7423, 0x7424, 0x7427, 0x7429, 0x742b, 0x742d, 0x742f,
01123 0x7431, 0x7432, 0x7437, 0x7438, 0x7439, 0x743a, 0x743b, 0x743d,
01124 0x743e, 0x743f, 0x7440, 0x7442, 0x7443, 0x7444, 0x7445, 0x7446,
01125 0x7447, 0x7448, 0x7449, 0x744a, 0x744b, 0x744c, 0x744d, 0x744e,
01126 0x744f, 0x7450, 0x7451, 0x7452, 0x7453, 0x7454, 0x7456, 0x7458,
01127 0x745d, 0x7460, 0x7461, 0x7462, 0x7463, 0x7464, 0x7465, 0x7466,
01128 0x7467, 0x7468, 0x7469, 0x746a, 0x746b, 0x746c, 0x746e, 0x746f,
01129 0x7471, 0x7472, 0x7473, 0x7474, 0x7475, 0x7478, 0x7479, 0x747a,
01130 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01131 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01132 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01133 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01134 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01135 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
01136 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,

```

```
01137 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01138 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01139 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01140 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01141 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01142 /* 0xad */
01143 0x747b, 0x747c, 0x747d, 0x747f, 0x7482, 0x7484, 0x7485, 0x7486,
01144 0x7488, 0x7489, 0x748a, 0x748c, 0x748d, 0x748f, 0x7491, 0x7492,
01145 0x7493, 0x7494, 0x7495, 0x7496, 0x7497, 0x7498, 0x7499, 0x749a,
01146 0x749b, 0x749d, 0x749f, 0x74a0, 0x74a1, 0x74a2, 0x74a3, 0x74a4,
01147 0x74a5, 0x74a6, 0x74aa, 0x74ab, 0x74ac, 0x74ad, 0x74ae, 0x74af,
01148 0x74b0, 0x74b1, 0x74b2, 0x74b3, 0x74b4, 0x74b5, 0x74b6, 0x74b7,
01149 0x74b8, 0x74b9, 0x74bb, 0x74bc, 0x74bd, 0x74be, 0x74bf, 0x74c0,
01150 0x74c1, 0x74c2, 0x74c3, 0x74c4, 0x74c5, 0x74c6, 0x74c7, 0x74c8,
01151 0x74c9, 0x74ca, 0x74cb, 0x74cc, 0x74cd, 0x74ce, 0x74cf, 0x74d0,
01152 0x74d1, 0x74d3, 0x74d4, 0x74d5, 0x74d6, 0x74d7, 0x74d8, 0x74d9,
01153 0x74da, 0x74db, 0x74dd, 0x74df, 0x74e1, 0x74e5, 0x74e7, 0x74e8,
01154 0x74e9, 0x74ea, 0x74eb, 0x74ec, 0x74ed, 0x74f0, 0x74f1, 0x74f2,
01155 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01156 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01157 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01158 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01159 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01160 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01161 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01162 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01163 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01164 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01165 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01166 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01167 /* 0xae */
01168 0x74f3, 0x74f5, 0x74f8, 0x74f9, 0x74fa, 0x74fb, 0x74fc, 0x74fd,
01169 0x74fe, 0x7500, 0x7501, 0x7502, 0x7503, 0x7505, 0x7506, 0x7507,
01170 0x7508, 0x7509, 0x750a, 0x750b, 0x750c, 0x750e, 0x7510, 0x7512,
01171 0x7514, 0x7515, 0x7516, 0x7517, 0x751b, 0x751d, 0x751e, 0x7520,
01172 0x7521, 0x7522, 0x7523, 0x7524, 0x7526, 0x7527, 0x752a, 0x752e,
01173 0x7534, 0x7536, 0x7539, 0x753c, 0x753d, 0x753f, 0x7541, 0x7542,
01174 0x7543, 0x7544, 0x7546, 0x7547, 0x7549, 0x754a, 0x754d, 0x7550,
01175 0x7551, 0x7552, 0x7553, 0x7555, 0x7556, 0x7557, 0x7558, 0x755d,
01176 0x755e, 0x755f, 0x7560, 0x7561, 0x7562, 0x7563, 0x7564, 0x7567,
01177 0x7568, 0x7569, 0x756b, 0x756c, 0x756d, 0x756e, 0x756f, 0x7570,
01178 0x7571, 0x7573, 0x7575, 0x7576, 0x7577, 0x757a, 0x757b, 0x757c,
01179 0x757d, 0x757e, 0x7580, 0x7581, 0x7582, 0x7584, 0x7585, 0x7587,
01180 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01181 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01182 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01183 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01184 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01185 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01186 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01187 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01188 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01189 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01190 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01191 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01192 /* 0xaf */
01193 0x7588, 0x7589, 0x758a, 0x758c, 0x758d, 0x758e, 0x7590, 0x7593,
01194 0x7595, 0x7598, 0x759b, 0x759c, 0x759e, 0x75a2, 0x75a6, 0x75a7,
01195 0x75a8, 0x75a9, 0x75aa, 0x75ad, 0x75b6, 0x75b7, 0x75ba, 0x75bb,
01196 0x75bf, 0x75c0, 0x75c1, 0x75c6, 0x75cb, 0x75cc, 0x75ce, 0x75cf,
01197 0x75d0, 0x75d1, 0x75d3, 0x75d7, 0x75d9, 0x75da, 0x75dc, 0x75dd,
01198 0x75df, 0x75e0, 0x75e1, 0x75e5, 0x75e9, 0x75ec, 0x75ed, 0x75ee,
01199 0x75ef, 0x75f2, 0x75f3, 0x75f5, 0x75f6, 0x75f7, 0x75f8, 0x75fa,
01200 0x75fb, 0x75fd, 0x75fe, 0x7602, 0x7604, 0x7606, 0x7607, 0x7608,
01201 0x7609, 0x760b, 0x760d, 0x760e, 0x760f, 0x7611, 0x7612, 0x7613,
01202 0x7614, 0x7616, 0x761a, 0x761c, 0x761d, 0x761e, 0x7621, 0x7623,
01203 0x7627, 0x7628, 0x762c, 0x762e, 0x762f, 0x7631, 0x7632, 0x7636,
01204 0x7637, 0x7639, 0x763a, 0x763b, 0x763d, 0x7641, 0x7642, 0x7644,
01205 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01206 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01207 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01208 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01209 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01210 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01211 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01212 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01213 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01214 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01215 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01216 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
01217 /* 0xb0 */
01218 0x7645, 0x7646, 0x7647, 0x7648, 0x7649, 0x764a, 0x764b, 0x764e,
01219 0x764f, 0x7650, 0x7651, 0x7652, 0x7653, 0x7655, 0x7657, 0x7658,
01220 0x7659, 0x765a, 0x765b, 0x765d, 0x765f, 0x7660, 0x7661, 0x7662,
01221 0x7664, 0x7665, 0x7666, 0x7667, 0x7668, 0x7669, 0x766a, 0x766c,
01222 0x766d, 0x766e, 0x7670, 0x7671, 0x7672, 0x7673, 0x7674, 0x7675,
01223 0x7676, 0x7677, 0x7679, 0x767a, 0x767c, 0x767f, 0x7680, 0x7681,
```

```

01224 0x7683, 0x7685, 0x7689, 0x768a, 0x768c, 0x768d, 0x768f, 0x7690,
01225 0x7692, 0x7694, 0x7695, 0x7697, 0x7698, 0x769a, 0x769b, 0x769c,
01226 0x769d, 0x769e, 0x769f, 0x76a0, 0x76a1, 0x76a2, 0x76a3, 0x76a5,
01227 0x76a6, 0x76a7, 0x76a8, 0x76a9, 0x76aa, 0x76ab, 0x76ac, 0x76ad,
01228 0x76af, 0x76b0, 0x76b3, 0x76b5, 0x76b6, 0x76b7, 0x76b8, 0x76b9,
01229 0x76ba, 0x76bb, 0x76bc, 0x76bd, 0x76be, 0x76c0, 0x76c1, 0x76c3,
01230 0x554a, 0x963f, 0x57c3, 0x6328, 0x54ce, 0x5509, 0x54c0, 0x7691,
01231 0x764c, 0x853c, 0x77ee, 0x827e, 0x788d, 0x7231, 0x9698, 0x978d,
01232 0x6c28, 0x5b89, 0x4ffa, 0x6309, 0x6697, 0x5cb8, 0x80fa, 0x6848,
01233 0x80ae, 0x6602, 0x76ce, 0x51f9, 0x6556, 0x71ac, 0x7ff1, 0x8884,
01234 0x50b2, 0x5965, 0x61ca, 0x6fb3, 0x82ad, 0x634c, 0x6252, 0x53ed,
01235 0x5427, 0x7b06, 0x516b, 0x75a4, 0x5df4, 0x62d4, 0x8dcb, 0x9776,
01236 0x628a, 0x8019, 0x575d, 0x9738, 0x7f62, 0x7238, 0x767d, 0x67cf,
01237 0x767e, 0x6446, 0x4f70, 0x8d25, 0x62dc, 0x7a17, 0x6591, 0x73ed,
01238 0x642c, 0x6273, 0x822c, 0x9881, 0x677f, 0x7248, 0x626e, 0x62cc,
01239 0x4f34, 0x74e3, 0x534a, 0x529e, 0x7eca, 0x90a6, 0x5e2e, 0x6886,
01240 0x699c, 0x8180, 0x7ed1, 0x68d2, 0x78c5, 0x868c, 0x9551, 0x508d,
01241 0x8c24, 0x82de, 0x80de, 0x5305, 0x8912, 0x5265,
01242 /* 0xb1 */
01243 0x76c4, 0x76c7, 0x76c9, 0x76cb, 0x76cc, 0x76d3, 0x76d5, 0x76d9,
01244 0x76da, 0x76dc, 0x76dd, 0x76de, 0x76e0, 0x76e1, 0x76e2, 0x76e3,
01245 0x76e4, 0x76e6, 0x76e7, 0x76e8, 0x76e9, 0x76ea, 0x76eb, 0x76ec,
01246 0x76ed, 0x76f0, 0x76f3, 0x76f5, 0x76f6, 0x76f7, 0x76fa, 0x76fb,
01247 0x76fd, 0x76ff, 0x7700, 0x7702, 0x7703, 0x7705, 0x7706, 0x770a,
01248 0x770c, 0x770e, 0x770f, 0x7710, 0x7711, 0x7712, 0x7713, 0x7714,
01249 0x7715, 0x7716, 0x7717, 0x7718, 0x771b, 0x771c, 0x771d, 0x771e,
01250 0x7721, 0x7723, 0x7724, 0x7725, 0x7727, 0x772a, 0x772b, 0x772c,
01251 0x772e, 0x7730, 0x7731, 0x7732, 0x7733, 0x7734, 0x7739, 0x773b,
01252 0x773d, 0x773e, 0x773f, 0x7742, 0x7744, 0x7745, 0x7746, 0x7748,
01253 0x7749, 0x774a, 0x774b, 0x774c, 0x774d, 0x774e, 0x774f, 0x7752,
01254 0x7753, 0x7754, 0x7755, 0x7756, 0x7757, 0x7758, 0x7759, 0x775c,
01255 0x8584, 0x96f9, 0x4fdd, 0x5821, 0x9971, 0x5b9d, 0x62b1, 0x62a5,
01256 0x66b4, 0x8c79, 0x9c8d, 0x7206, 0x676f, 0x7891, 0x60b2, 0x5351,
01257 0x5317, 0x8f88, 0x80cc, 0x8d1d, 0x94a1, 0x500d, 0x72c8, 0x5907,
01258 0x60eb, 0x7119, 0x88ab, 0x5954, 0x82ef, 0x672c, 0x7b28, 0x5d29,
01259 0x7ef7, 0x752d, 0x6cf5, 0x8e66, 0x8ff8, 0x903c, 0x9f3b, 0x6bd4,
01260 0x9119, 0x7b14, 0x5f7c, 0x78a7, 0x84d6, 0x853d, 0x6bd5, 0x6bd9,
01261 0x6bd6, 0x5e01, 0x5e87, 0x75f9, 0x95ed, 0x655d, 0x5f0a, 0x5fc5,
01262 0x8f9f, 0x58c1, 0x81c2, 0x907f, 0x965b, 0x97ad, 0x8fb9, 0x7f16,
01263 0x8d2c, 0x6241, 0x4fbf, 0x53d8, 0x535e, 0x8fa8, 0x8fa9, 0x8fab,
01264 0x904d, 0x6807, 0x5f6a, 0x8198, 0x8868, 0x9cd6, 0x618b, 0x522b,
01265 0x762a, 0x5f6c, 0x658c, 0x6fd2, 0x6ee8, 0x5bbe, 0x6448, 0x5175,
01266 0x51b0, 0x67c4, 0x4e19, 0x79c9, 0x997c, 0x70b3,
01267 /* 0xb2 */
01268 0x775d, 0x775e, 0x775f, 0x7760, 0x7764, 0x7767, 0x7769, 0x776a,
01269 0x776d, 0x776e, 0x776f, 0x7770, 0x7771, 0x7772, 0x7773, 0x7774,
01270 0x7775, 0x7776, 0x7777, 0x7778, 0x777a, 0x777b, 0x777c, 0x7781,
01271 0x7782, 0x7783, 0x7786, 0x7787, 0x7788, 0x7789, 0x778a, 0x778b,
01272 0x778f, 0x7790, 0x7793, 0x7794, 0x7795, 0x7796, 0x7797, 0x7798,
01273 0x7799, 0x779a, 0x779b, 0x779c, 0x779d, 0x779e, 0x77a1, 0x77a3,
01274 0x77a4, 0x77a6, 0x77a8, 0x77ab, 0x77ad, 0x77ae, 0x77af, 0x77b1,
01275 0x77b2, 0x77b4, 0x77b6, 0x77b7, 0x77b8, 0x77b9, 0x77ba, 0x77bc,
01276 0x77be, 0x77c0, 0x77c1, 0x77c2, 0x77c3, 0x77c4, 0x77c5, 0x77c6,
01277 0x77c7, 0x77c8, 0x77c9, 0x77ca, 0x77cb, 0x77cc, 0x77ce, 0x77cf,
01278 0x77d0, 0x77d1, 0x77d2, 0x77d3, 0x77d4, 0x77d5, 0x77d6, 0x77d8,
01279 0x77d9, 0x77da, 0x77dd, 0x77de, 0x77df, 0x77e0, 0x77e1, 0x77e4,
01280 0x75c5, 0x5e76, 0x73bb, 0x83e0, 0x64ad, 0x62e8, 0x94b5, 0x6ce2,
01281 0x535a, 0x52c3, 0x640f, 0x94c2, 0x7b94, 0x4f2f, 0x5e1b, 0x8236,
01282 0x8116, 0x818a, 0x6e24, 0x6cca, 0x9a73, 0x6355, 0x535c, 0x54fa,
01283 0x8865, 0x57e0, 0x4e0d, 0x5e03, 0x6b65, 0x7c3f, 0x90e8, 0x6016,
01284 0x64e6, 0x731c, 0x88c1, 0x6750, 0x624d, 0x8d22, 0x776c, 0x8e29,
01285 0x91c7, 0x5f69, 0x83dc, 0x8521, 0x9910, 0x53c2, 0x8695, 0x6b8b,
01286 0x60ed, 0x60e8, 0x707f, 0x82cd, 0x8231, 0x4ed3, 0x6ca7, 0x85cf,
01287 0x64cd, 0x7cd9, 0x69fd, 0x66f9, 0x8349, 0x5395, 0x7b56, 0x4fa7,
01288 0x518c, 0x6d4b, 0x5c42, 0x8e6d, 0x63d2, 0x53c9, 0x832c, 0x8336,
01289 0x67e5, 0x78b4, 0x643d, 0x5bdf, 0x5c94, 0x5dee, 0x8be7, 0x62c6,
01290 0x67f4, 0x8c7a, 0x6400, 0x63ba, 0x8749, 0x998b, 0x8c17, 0x7f20,
01291 0x94f2, 0x4ea7, 0x9610, 0x98a4, 0x660c, 0x7316,
01292 /* 0xb3 */
01293 0x77e6, 0x77e8, 0x77ea, 0x77ef, 0x77f0, 0x77f1, 0x77f2, 0x77f4,
01294 0x77f5, 0x77f7, 0x77f9, 0x77fa, 0x77fb, 0x77fc, 0x7803, 0x7804,
01295 0x7805, 0x7806, 0x7807, 0x7808, 0x780a, 0x780b, 0x780e, 0x780f,
01296 0x7810, 0x7813, 0x7815, 0x7819, 0x781b, 0x781e, 0x7820, 0x7821,
01297 0x7822, 0x7824, 0x7828, 0x782a, 0x782b, 0x782e, 0x782f, 0x7831,
01298 0x7832, 0x7833, 0x7835, 0x7836, 0x783d, 0x783f, 0x7841, 0x7842,
01299 0x7843, 0x7844, 0x7846, 0x7848, 0x7849, 0x784a, 0x784b, 0x784d,
01300 0x784f, 0x7851, 0x7853, 0x7854, 0x7858, 0x7859, 0x785a, 0x785b,
01301 0x785c, 0x785e, 0x785f, 0x7860, 0x7861, 0x7862, 0x7863, 0x7864,
01302 0x7865, 0x7866, 0x7867, 0x7868, 0x7869, 0x786f, 0x7870, 0x7871,
01303 0x7872, 0x7873, 0x7874, 0x7875, 0x7876, 0x7878, 0x7879, 0x787a,
01304 0x787b, 0x787d, 0x787e, 0x787f, 0x7880, 0x7881, 0x7882, 0x7883,
01305 0x573a, 0x5c1d, 0x5e38, 0x957f, 0x507f, 0x80a0, 0x5382, 0x655e,
01306 0x7545, 0x5531, 0x5021, 0x8d85, 0x6284, 0x949e, 0x671d, 0x5632,
01307 0x6f6e, 0x5de2, 0x5435, 0x7092, 0x8f66, 0x626f, 0x64a4, 0x63a3,
01308 0x5f7b, 0x6f88, 0x90f4, 0x81e3, 0x8fb0, 0x5c18, 0x6668, 0x5ff1,
01309 0x6c89, 0x9648, 0x8d81, 0x886c, 0x6491, 0x79f0, 0x57ce, 0x6a59,
01310 0x6210, 0x5448, 0x4e58, 0x7a0b, 0x60e9, 0x6f84, 0x8bda, 0x627f,

```

```
01311 0x901e, 0x9a8b, 0x79e4, 0x5403, 0x75f4, 0x6301, 0x5319, 0x6c60,
01312 0x8fdf, 0x5f1b, 0x9a70, 0x803b, 0x9f7f, 0x4f88, 0x5c3a, 0x8d64,
01313 0x7fc5, 0x65a5, 0x70bd, 0x5145, 0x51b2, 0x866b, 0x5d07, 0x5ba0,
01314 0x62bd, 0x916c, 0x7574, 0x8e0c, 0x7a20, 0x6101, 0x7b79, 0x4ec7,
01315 0x7ef8, 0x7785, 0x4e11, 0x81ed, 0x521d, 0x51fa, 0x6a71, 0x53a8,
01316 0x8e87, 0x9504, 0x96cf, 0x6ec1, 0x9664, 0x695a,
01317 /* 0xb4 */
01318 0x7884, 0x7885, 0x7886, 0x7888, 0x788a, 0x788b, 0x788f, 0x7890,
01319 0x7892, 0x7894, 0x7895, 0x7896, 0x7899, 0x789d, 0x789e, 0x78a0,
01320 0x78a2, 0x78a4, 0x78a6, 0x78a8, 0x78a9, 0x78aa, 0x78ab, 0x78ac,
01321 0x78ad, 0x78ae, 0x78af, 0x78b5, 0x78b6, 0x78b7, 0x78b8, 0x78ba,
01322 0x78bb, 0x78bc, 0x78bd, 0x78bf, 0x78c0, 0x78c2, 0x78c3, 0x78c4,
01323 0x78c6, 0x78c7, 0x78c8, 0x78cc, 0x78cd, 0x78ce, 0x78cf, 0x78d1,
01324 0x78d2, 0x78d3, 0x78d6, 0x78d7, 0x78d8, 0x78da, 0x78db, 0x78dc,
01325 0x78dd, 0x78de, 0x78df, 0x78e0, 0x78e1, 0x78e2, 0x78e3, 0x78e4,
01326 0x78e5, 0x78e6, 0x78e7, 0x78e9, 0x78ea, 0x78eb, 0x78ed, 0x78ee,
01327 0x78ef, 0x78f0, 0x78f1, 0x78f3, 0x78f5, 0x78f6, 0x78f8, 0x78f9,
01328 0x78fb, 0x78fc, 0x78fd, 0x78fe, 0x78ff, 0x7900, 0x7902, 0x7903,
01329 0x7904, 0x7906, 0x7907, 0x7908, 0x7909, 0x790a, 0x790b, 0x790c,
01330 0x7840, 0x50a8, 0x77d7, 0x6410, 0x89e6, 0x5904, 0x63e3, 0x5ddd,
01331 0x7a7f, 0x693d, 0x4f20, 0x8239, 0x5598, 0x4e32, 0x75ae, 0x7a97,
01332 0x5e62, 0x5e8a, 0x95ef, 0x521b, 0x5439, 0x708a, 0x6376, 0x9524,
01333 0x5782, 0x6625, 0x693f, 0x9187, 0x5507, 0x6df3, 0x7ea9, 0x8822,
01334 0x6233, 0x7ef0, 0x75b5, 0x8328, 0x78c1, 0x96cc, 0x8f9e, 0x6148,
01335 0x74f7, 0x8bcd, 0x6b64, 0x523a, 0x8d50, 0x6b21, 0x806a, 0x8471,
01336 0x56f1, 0x5306, 0x4ece, 0x4e1b, 0x51d1, 0x7c97, 0x918b, 0x7c07,
01337 0x4fc3, 0x8e7f, 0x7be1, 0x7a9c, 0x6467, 0x5d14, 0x50ac, 0x8106,
01338 0x7601, 0x7cb9, 0x6dec, 0x7fe0, 0x6751, 0x5b58, 0x5bf8, 0x78cb,
01339 0x64ae, 0x6413, 0x63aa, 0x632b, 0x9519, 0x642d, 0x8f8e, 0x7b54,
01340 0x7629, 0x6253, 0x5927, 0x5446, 0x6b79, 0x50a3, 0x6234, 0x5e26,
01341 0x6b86, 0x4ee3, 0x8d37, 0x888b, 0x5f85, 0x902e,
01342 /* 0xb5 */
01343 0x790d, 0x790e, 0x790f, 0x7910, 0x7911, 0x7912, 0x7914, 0x7915,
01344 0x7916, 0x7917, 0x7918, 0x7919, 0x791a, 0x791b, 0x791c, 0x791d,
01345 0x791f, 0x7920, 0x7921, 0x7922, 0x7923, 0x7925, 0x7926, 0x7927,
01346 0x7928, 0x7929, 0x792a, 0x792b, 0x792c, 0x792d, 0x792e, 0x792f,
01347 0x7930, 0x7931, 0x7932, 0x7933, 0x7935, 0x7936, 0x7937, 0x7938,
01348 0x7939, 0x793d, 0x793f, 0x7942, 0x7943, 0x7944, 0x7945, 0x7947,
01349 0x794a, 0x794b, 0x794c, 0x794d, 0x794e, 0x794f, 0x7950, 0x7951,
01350 0x7952, 0x7954, 0x7955, 0x7958, 0x7959, 0x7961, 0x7963, 0x7964,
01351 0x7966, 0x7969, 0x796a, 0x796b, 0x796c, 0x796e, 0x7970, 0x7971,
01352 0x7972, 0x7973, 0x7974, 0x7975, 0x7976, 0x7979, 0x797b, 0x797c,
01353 0x797d, 0x797e, 0x797f, 0x7982, 0x7983, 0x7986, 0x7987, 0x7988,
01354 0x7989, 0x798b, 0x798c, 0x798d, 0x798e, 0x7990, 0x7991, 0x7992,
01355 0x6020, 0x803d, 0x62c5, 0x4e39, 0x5355, 0x90f8, 0x63b8, 0x80c6,
01356 0x65e6, 0x6c2e, 0x4f46, 0x60ee, 0x6de1, 0x8bde, 0x5f39, 0x86cb,
01357 0x5f53, 0x6321, 0x515a, 0x8361, 0x6863, 0x5200, 0x6363, 0x8e48,
01358 0x5012, 0x5c9b, 0x7977, 0x5bfc, 0x5230, 0x7a3b, 0x60bc, 0x9053,
01359 0x76d7, 0x5fb7, 0x5f97, 0x7684, 0x8e6c, 0x706f, 0x767b, 0x7b49,
01360 0x77aa, 0x51f3, 0x9093, 0x5824, 0x4f4e, 0x6ef4, 0x8fea, 0x654c,
01361 0x7b1b, 0x72c4, 0x6da4, 0x7fdf, 0x5ae1, 0x62b5, 0x5e95, 0x5730,
01362 0x8482, 0x7b2c, 0x5e1d, 0x5f1f, 0x9012, 0x7f14, 0x98a0, 0x6382,
01363 0x6ec7, 0x7898, 0x70b9, 0x5178, 0x975b, 0x57ab, 0x7535, 0x4f43,
01364 0x7538, 0x5e97, 0x60e6, 0x5960, 0x6dc0, 0x6bbf, 0x7889, 0x53fc,
01365 0x96d5, 0x51cb, 0x5201, 0x6389, 0x540a, 0x9493, 0x8c03, 0x8dcc,
01366 0x7239, 0x789f, 0x8776, 0x8fed, 0x8c0d, 0x53e0,
01367 /* 0xb6 */
01368 0x7993, 0x7994, 0x7995, 0x7996, 0x7997, 0x7998, 0x7999, 0x799b,
01369 0x799c, 0x799d, 0x799e, 0x799f, 0x79a0, 0x79a1, 0x79a2, 0x79a3,
01370 0x79a4, 0x79a5, 0x79a6, 0x79a8, 0x79a9, 0x79aa, 0x79ab, 0x79ac,
01371 0x79ad, 0x79ae, 0x79af, 0x79b0, 0x79b1, 0x79b2, 0x79b4, 0x79b5,
01372 0x79b6, 0x79b7, 0x79b8, 0x79bc, 0x79bf, 0x79c2, 0x79c4, 0x79c5,
01373 0x79c7, 0x79c8, 0x79ca, 0x79cc, 0x79ce, 0x79cf, 0x79d0, 0x79d3,
01374 0x79d4, 0x79d6, 0x79d7, 0x79d9, 0x79da, 0x79db, 0x79dc, 0x79dd,
01375 0x79de, 0x79e0, 0x79e1, 0x79e2, 0x79e5, 0x79e8, 0x79ea, 0x79ec,
01376 0x79ee, 0x79f1, 0x79f2, 0x79f3, 0x79f4, 0x79f5, 0x79f6, 0x79f7,
01377 0x79f9, 0x79fa, 0x79fc, 0x79fe, 0x79ff, 0x7a01, 0x7a04, 0x7a05,
01378 0x7a07, 0x7a08, 0x7a09, 0x7a0a, 0x7a0c, 0x7a0f, 0x7a10, 0x7a11,
01379 0x7a12, 0x7a13, 0x7a15, 0x7a16, 0x7a18, 0x7a19, 0x7a1b, 0x7a1c,
01380 0x4e01, 0x76ef, 0x53ee, 0x9489, 0x9876, 0x9f0e, 0x952d, 0x5b9a,
01381 0x8ba2, 0x4e22, 0x4e1c, 0x51ac, 0x8463, 0x61c2, 0x52a8, 0x680b,
01382 0x4f97, 0x606b, 0x51bb, 0x6d1e, 0x515c, 0x6296, 0x6597, 0x9661,
01383 0x8c46, 0x9017, 0x75d8, 0x90fd, 0x7763, 0x6bd2, 0x728a, 0x72ec,
01384 0x8bfb, 0x5835, 0x7779, 0x8d4c, 0x675c, 0x9540, 0x809a, 0x5ea6,
01385 0x6e21, 0x5992, 0x7aef, 0x77ed, 0x953b, 0x6bb5, 0x65ad, 0x7f0e,
01386 0x5806, 0x5151, 0x961f, 0x5bf9, 0x58a9, 0x5428, 0x8e72, 0x6566,
01387 0x987f, 0x56e4, 0x949d, 0x76fe, 0x9041, 0x6387, 0x54c6, 0x591a,
01388 0x593a, 0x579b, 0x8eb2, 0x6735, 0x8dfa, 0x8235, 0x5241, 0x60f0,
01389 0x5815, 0x86fe, 0x5ce8, 0x9e45, 0x4fc4, 0x989d, 0x8bb9, 0x5a25,
01390 0x6076, 0x5384, 0x627c, 0x904f, 0x9102, 0x997f, 0x6069, 0x800c,
01391 0x513f, 0x8033, 0x5c14, 0x9975, 0x6d31, 0x4e8c,
01392 /* 0xb7 */
01393 0x7a1d, 0x7a1f, 0x7a21, 0x7a22, 0x7a24, 0x7a25, 0x7a26, 0x7a27,
01394 0x7a28, 0x7a29, 0x7a2a, 0x7a2b, 0x7a2c, 0x7a2d, 0x7a2e, 0x7a2f,
01395 0x7a30, 0x7a31, 0x7a32, 0x7a34, 0x7a35, 0x7a36, 0x7a38, 0x7a3a,
01396 0x7a3e, 0x7a40, 0x7a41, 0x7a42, 0x7a43, 0x7a44, 0x7a45, 0x7a47,
01397 0x7a48, 0x7a49, 0x7a4a, 0x7a4b, 0x7a4c, 0x7a4d, 0x7a4e, 0x7a4f,
```



```

01398 0x7a50, 0x7a52, 0x7a53, 0x7a54, 0x7a55, 0x7a56, 0x7a58, 0x7a59,
01399 0x7a5a, 0x7a5b, 0x7a5c, 0x7a5d, 0x7a5e, 0x7a5f, 0x7a60, 0x7a61,
01400 0x7a62, 0x7a63, 0x7a64, 0x7a65, 0x7a66, 0x7a67, 0x7a68, 0x7a69,
01401 0x7a6a, 0x7a6b, 0x7a6c, 0x7a6d, 0x7a6e, 0x7a6f, 0x7a71, 0x7a72,
01402 0x7a73, 0x7a75, 0x7a7b, 0x7a7c, 0x7a7d, 0x7a7e, 0x7a82, 0x7a85,
01403 0x7a87, 0x7a89, 0x7a8b, 0x7a8c, 0x7a8e, 0x7a8f, 0x7a90,
01404 0x7a93, 0x7a94, 0x7a99, 0x7a9a, 0x7a9b, 0x7a9e, 0x7aa1, 0x7aa2,
01405 0x8d30, 0x53d1, 0x7f5a, 0x7b4f, 0x4f10, 0x4e4f, 0x9600, 0x6cd5,
01406 0x73d0, 0x85e9, 0x5e06, 0x756a, 0x7ffb, 0x6a0a, 0x77fe, 0x9492,
01407 0x7e41, 0x51e1, 0x70e6, 0x53cd, 0x8fd4, 0x8303, 0x8d29, 0x72af,
01408 0x996d, 0x6cdb, 0x574a, 0x82b3, 0x65b9, 0x80aa, 0x623f, 0x9632,
01409 0x59a8, 0x4eff, 0x8bbf, 0x7eba, 0x653e, 0x83f2, 0x975e, 0x5561,
01410 0x98de, 0x80a5, 0x532a, 0x8bfd, 0x5420, 0x80ba, 0x5e9f, 0x6cb8,
01411 0x8d39, 0x82ac, 0x915a, 0x5429, 0x6c1b, 0x5206, 0x7eb7, 0x575f,
01412 0x711a, 0x6c7e, 0x7c89, 0x594b, 0x4efd, 0x5fff, 0x6124, 0x7caa,
01413 0x4e30, 0x5c01, 0x67ab, 0x8702, 0x5cf0, 0x950b, 0x98ce, 0x75af,
01414 0x70fd, 0x9022, 0x51af, 0x7f1d, 0x8bbd, 0x5949, 0x51e4, 0x4f5b,
01415 0x5426, 0x592b, 0x5e77, 0x80a4, 0x5b75, 0x6276, 0x62c2, 0x8f90,
01416 0x5e45, 0x6c1f, 0x7b26, 0x4f0f, 0x4fd8, 0x670d,
01417 /* 0xb8 */
01418 0x7aa3, 0x7aa4, 0x7aa7, 0x7aa9, 0x7aaa, 0x7aab, 0x7aae, 0x7aaf,
01419 0x7ab0, 0x7ab1, 0x7ab2, 0x7ab4, 0x7ab5, 0x7ab6, 0x7ab7, 0x7ab8,
01420 0x7ab9, 0x7aba, 0x7abb, 0x7abc, 0x7abd, 0x7abe, 0x7ac0, 0x7ac1,
01421 0x7ac2, 0x7ac3, 0x7ac4, 0x7ac5, 0x7ac6, 0x7ac7, 0x7ac8, 0x7ac9,
01422 0x7aca, 0x7acc, 0x7acd, 0x7ace, 0x7acf, 0x7ad0, 0x7ad1, 0x7ad2,
01423 0x7ad3, 0x7ad4, 0x7ad5, 0x7ad7, 0x7ad8, 0x7ada, 0x7adb, 0x7adc,
01424 0x7add, 0x7ae1, 0x7ae2, 0x7ae4, 0x7ae7, 0x7ae8, 0x7ae9, 0x7aea,
01425 0x7aeb, 0x7aec, 0x7aee, 0x7af0, 0x7af1, 0x7af2, 0x7af3, 0x7af4,
01426 0x7af5, 0x7af6, 0x7af7, 0x7af8, 0x7afb, 0x7afc, 0x7afe, 0x7b00,
01427 0x7b01, 0x7b02, 0x7b05, 0x7b07, 0x7b09, 0x7b0c, 0x7b0d, 0x7b0e,
01428 0x7b10, 0x7b12, 0x7b13, 0x7b16, 0x7b17, 0x7b18, 0x7b1a, 0x7b1c,
01429 0x7b1d, 0x7b1f, 0x7b21, 0x7b22, 0x7b23, 0x7b27, 0x7b29, 0x7b2d,
01430 0x6d6e, 0x6daa, 0x798f, 0x88b1, 0x5f17, 0x752b, 0x629a, 0x8f85,
01431 0x4fef, 0x91dc, 0x65a7, 0x812f, 0x8151, 0x5e9c, 0x8150, 0x8d74,
01432 0x526f, 0x8986, 0x8d4b, 0x590d, 0x5085, 0x4ed8, 0x961c, 0x7236,
01433 0x8179, 0x8d1f, 0x5bcc, 0x8ba3, 0x9644, 0x5987, 0x7f1a, 0x5490,
01434 0x5676, 0x560e, 0x8be5, 0x6539, 0x6982, 0x9499, 0x76d6, 0x6e89,
01435 0x5e72, 0x7518, 0x6746, 0x67d1, 0x7aff, 0x809d, 0x8d76, 0x611f,
01436 0x79c6, 0x6562, 0x8d63, 0x5188, 0x521a, 0x94a2, 0x7f38, 0x809b,
01437 0x7eb2, 0x5c97, 0x6e2f, 0x6760, 0x7bd9, 0x768b, 0x9ad8, 0x818f,
01438 0x7f94, 0x7cd5, 0x641e, 0x9550, 0x7a3f, 0x544a, 0x54ae, 0x6b4c,
01439 0x6401, 0x6208, 0x9e3d, 0x80f3, 0x7599, 0x5272, 0x9769, 0x845b,
01440 0x683c, 0x86e4, 0x9601, 0x9694, 0x94ec, 0x4e2a, 0x5404, 0x7ed9,
01441 0x6839, 0x8ddf, 0x8015, 0x66f4, 0x5e9a, 0x7fb9,
01442 /* 0xb9 */
01443 0x7b2f, 0x7b30, 0x7b32, 0x7b34, 0x7b35, 0x7b36, 0x7b37, 0x7b39,
01444 0x7b3b, 0x7b3d, 0x7b3f, 0x7b40, 0x7b41, 0x7b42, 0x7b43, 0x7b44,
01445 0x7b46, 0x7b48, 0x7b4a, 0x7b4d, 0x7b4e, 0x7b53, 0x7b55, 0x7b57,
01446 0x7b59, 0x7b5c, 0x7b5e, 0x7b5f, 0x7b61, 0x7b63, 0x7b64, 0x7b65,
01447 0x7b66, 0x7b67, 0x7b68, 0x7b69, 0x7b6a, 0x7b6b, 0x7b6c, 0x7b6d,
01448 0x7b6f, 0x7b70, 0x7b73, 0x7b74, 0x7b76, 0x7b78, 0x7b7a, 0x7b7c,
01449 0x7b7d, 0x7b7f, 0x7b81, 0x7b82, 0x7b83, 0x7b84, 0x7b86, 0x7b87,
01450 0x7b88, 0x7b89, 0x7b8a, 0x7b8b, 0x7b8c, 0x7b8e, 0x7b8f, 0x7b91,
01451 0x7b92, 0x7b93, 0x7b96, 0x7b98, 0x7b99, 0x7b9a, 0x7b9b, 0x7b9e,
01452 0x7b9f, 0x7ba0, 0x7ba3, 0x7ba4, 0x7ba5, 0x7bae, 0x7baf, 0x7bb0,
01453 0x7bb2, 0x7bb3, 0x7bb5, 0x7bb6, 0x7bb7, 0x7bb9, 0x7bba, 0x7bbb,
01454 0x7bbc, 0x7bbd, 0x7bbe, 0x7bbf, 0x7bc0, 0x7bc2, 0x7bc3, 0x7bc4,
01455 0x57c2, 0x803f, 0x6897, 0x5de5, 0x653b, 0x529f, 0x606d, 0x9f9a,
01456 0x4f9b, 0x8eac, 0x516c, 0x5bab, 0x5f13, 0x5de9, 0x6c5e, 0x62f1,
01457 0x8d21, 0x5171, 0x94a9, 0x52fe, 0x6c9f, 0x82df, 0x72d7, 0x57a2,
01458 0x6784, 0x8d2d, 0x591f, 0x8f9c, 0x83c7, 0x5495, 0x7b8d, 0x4f30,
01459 0x6cbd, 0x5b64, 0x59d1, 0x9f13, 0x53e4, 0x86ca, 0x9aa8, 0x8c37,
01460 0x80a1, 0x6545, 0x987e, 0x56fa, 0x96c7, 0x522e, 0x74dc, 0x5250,
01461 0x5be1, 0x6302, 0x8902, 0x4e56, 0x62d0, 0x602a, 0x68fa, 0x5173,
01462 0x5b98, 0x51a0, 0x89c2, 0x7ba1, 0x9986, 0x7f50, 0x60ef, 0x704c,
01463 0x8d2f, 0x5149, 0x5e7f, 0x901b, 0x7470, 0x89c4, 0x572d, 0x7845,
01464 0x5f52, 0x9f9f, 0x95fa, 0x8f68, 0x9b3c, 0x8be1, 0x7678, 0x6842,
01465 0x67dc, 0x8dea, 0x8d35, 0x523d, 0x8f8a, 0x6eda, 0x68cd, 0x9505,
01466 0x90ed, 0x56fd, 0x679c, 0x88f9, 0x8fc7, 0x54c8,
01467 /* 0xba */
01468 0x7bc5, 0x7bc8, 0x7bc9, 0x7bca, 0x7bcb, 0x7bcd, 0x7bce, 0x7bcf,
01469 0x7bd0, 0x7bd2, 0x7bd4, 0x7bd5, 0x7bd6, 0x7bd7, 0x7bd8, 0x7bdb,
01470 0x7bdc, 0x7bde, 0x7bdf, 0x7be0, 0x7be2, 0x7be3, 0x7be4, 0x7be7,
01471 0x7be8, 0x7be9, 0x7beb, 0x7bec, 0x7bed, 0x7bef, 0x7bf0, 0x7bf2,
01472 0x7bf3, 0x7bf4, 0x7bf5, 0x7bf6, 0x7bf8, 0x7bf9, 0x7bfa, 0x7bfb,
01473 0x7bfd, 0x7bff, 0x7c00, 0x7c01, 0x7c02, 0x7c03, 0x7c04, 0x7c05,
01474 0x7c06, 0x7c08, 0x7c09, 0x7c0a, 0x7c0d, 0x7c0e, 0x7c10, 0x7c11,
01475 0x7c12, 0x7c13, 0x7c14, 0x7c15, 0x7c17, 0x7c18, 0x7c19, 0x7c1a,
01476 0x7c1b, 0x7c1c, 0x7c1d, 0x7c1e, 0x7c20, 0x7c21, 0x7c22, 0x7c23,
01477 0x7c24, 0x7c25, 0x7c28, 0x7c29, 0x7c2b, 0x7c2c, 0x7c2d, 0x7c2e,
01478 0x7c2f, 0x7c30, 0x7c31, 0x7c32, 0x7c33, 0x7c34, 0x7c35, 0x7c36,
01479 0x7c37, 0x7c39, 0x7c3a, 0x7c3b, 0x7c3c, 0x7c3d, 0x7c3e, 0x7c42,
01480 0x9ab8, 0x5b69, 0x6d77, 0x6c26, 0x4ea5, 0x5bb3, 0x9a87, 0x9163,
01481 0x61a8, 0x90af, 0x97e9, 0x542b, 0x6db5, 0x5bd2, 0x51fd, 0x558a,
01482 0x7f55, 0x7ff0, 0x64bc, 0x634d, 0x65f1, 0x61be, 0x608d, 0x710a,
01483 0x6c57, 0x6c49, 0x592f, 0x676d, 0x822a, 0x58d5, 0x568e, 0x8c6a,
01484 0x6beb, 0x90dd, 0x597d, 0x8017, 0x53f7, 0x6d69, 0x5475, 0x559d,

```



```
01485 0x8377, 0x83cf, 0x6838, 0x79be, 0x548c, 0x4f55, 0x5408, 0x76d2,
01486 0x8c89, 0x9602, 0x6cb3, 0x6db8, 0x8d6b, 0x8910, 0x9e64, 0x8d3a,
01487 0x563f, 0x9ed1, 0x75d5, 0x5f88, 0x72e0, 0x6068, 0x54fc, 0x4ea8,
01488 0x6a2a, 0x8861, 0x6052, 0x8f70, 0x54c4, 0x70d8, 0x8679, 0x9e3f,
01489 0x6d2a, 0x5b8f, 0x5f18, 0x7ea2, 0x5589, 0x4faf, 0x7334, 0x543c,
01490 0x539a, 0x5019, 0x540e, 0x547c, 0x4e4e, 0x5ffd, 0x745a, 0x58f6,
01491 0x846b, 0x80e1, 0x8774, 0x72d0, 0x7cca, 0x6e56,
01492 /* 0xbb */
01493 0x7c43, 0x7c44, 0x7c45, 0x7c46, 0x7c47, 0x7c48, 0x7c49, 0x7c4a,
01494 0x7c4b, 0x7c4c, 0x7c4e, 0x7c4f, 0x7c50, 0x7c51, 0x7c52, 0x7c53,
01495 0x7c54, 0x7c55, 0x7c56, 0x7c57, 0x7c58, 0x7c59, 0x7c5a, 0x7c5b,
01496 0x7c5c, 0x7c5d, 0x7c5e, 0x7c5f, 0x7c60, 0x7c61, 0x7c62, 0x7c63,
01497 0x7c64, 0x7c65, 0x7c66, 0x7c67, 0x7c68, 0x7c69, 0x7c6a, 0x7c6b,
01498 0x7c6c, 0x7c6d, 0x7c6e, 0x7c6f, 0x7c70, 0x7c71, 0x7c72, 0x7c75,
01499 0x7c76, 0x7c77, 0x7c78, 0x7c79, 0x7c7a, 0x7c7e, 0x7c7f, 0x7c80,
01500 0x7c81, 0x7c82, 0x7c83, 0x7c84, 0x7c85, 0x7c86, 0x7c87, 0x7c88,
01501 0x7c8a, 0x7c8b, 0x7c8c, 0x7c8d, 0x7c8e, 0x7c8f, 0x7c90, 0x7c93,
01502 0x7c94, 0x7c96, 0x7c98, 0x7c99, 0x7c9a, 0x7c9b, 0x7ca0, 0x7ca1, 0x7ca3,
01503 0x7ca6, 0x7ca7, 0x7ca8, 0x7ca9, 0x7cab, 0x7cac, 0x7cad, 0x7caf,
01504 0x7cb0, 0x7cb4, 0x7cb5, 0x7cb6, 0x7cb7, 0x7cb8, 0x7cba, 0x7cbb,
01505 0x5f27, 0x864e, 0x552c, 0x62a4, 0x4e92, 0x6caa, 0x6237, 0x82b1,
01506 0x54d7, 0x534e, 0x733e, 0x6ed1, 0x753b, 0x5212, 0x5316, 0x8bdd,
01507 0x69d0, 0x5f8d, 0x6000, 0x6dee, 0x574f, 0x6b22, 0x73af, 0x6853,
01508 0x8fd8, 0x7f13, 0x6362, 0x60a3, 0x5524, 0x75ae, 0x8c62, 0x7115,
01509 0x6da3, 0x5ba6, 0x5e7b, 0x8352, 0x614c, 0x9ec4, 0x78fa, 0x8757,
01510 0x7c27, 0x7687, 0x51f0, 0x60f6, 0x714c, 0x6643, 0x5e4c, 0x604d,
01511 0x8c0e, 0x7070, 0x6325, 0x8f89, 0x5fbd, 0x6062, 0x86d4, 0x56de,
01512 0x6bc1, 0x6094, 0x6167, 0x5349, 0x60e0, 0x6666, 0x8d3f, 0x79fd,
01513 0x4f1a, 0x70e9, 0x6c47, 0x8bb3, 0x8bf2, 0x7ed8, 0x8364, 0x660f,
01514 0x5a5a, 0x9b42, 0x6d51, 0x6df7, 0x8c41, 0x6d3b, 0x4f19, 0x706b,
01515 0x83b7, 0x6216, 0x60d1, 0x970d, 0x8d27, 0x7978, 0x51fb, 0x573e,
01516 0x57fa, 0x673a, 0x7578, 0x7a3d, 0x79ef, 0x7b95,
01517 /* 0xbc */
01518 0x7cbf, 0x7cc0, 0x7cc2, 0x7cc3, 0x7cc4, 0x7cc6, 0x7cc9, 0x7ccb,
01519 0x7cce, 0x7ccf, 0x7cd0, 0x7cd1, 0x7cd2, 0x7cd3, 0x7cd4, 0x7cd8,
01520 0x7cda, 0x7cdb, 0x7cdd, 0x7cde, 0x7ce1, 0x7ce2, 0x7ce3, 0x7ce4,
01521 0x7ce5, 0x7ce6, 0x7ce7, 0x7ce9, 0x7cea, 0x7ceb, 0x7cec, 0x7ced,
01522 0x7cee, 0x7cf0, 0x7cf1, 0x7cf2, 0x7cf3, 0x7cf4, 0x7cf5, 0x7cf6,
01523 0x7cf7, 0x7cf9, 0x7cfa, 0x7cfc, 0x7cfd, 0x7cfe, 0x7cff, 0x7d00,
01524 0x7d01, 0x7d02, 0x7d03, 0x7d04, 0x7d05, 0x7d06, 0x7d07, 0x7d08,
01525 0x7d09, 0x7d0b, 0x7d0c, 0x7d0d, 0x7d0e, 0x7d0f, 0x7d10, 0x7d11,
01526 0x7d12, 0x7d13, 0x7d14, 0x7d15, 0x7d16, 0x7d17, 0x7d18, 0x7d19,
01527 0x7d1a, 0x7d1b, 0x7d1c, 0x7d1d, 0x7d1e, 0x7d1f, 0x7d21, 0x7d23,
01528 0x7d24, 0x7d25, 0x7d26, 0x7d28, 0x7d29, 0x7d2a, 0x7d2c, 0x7d2d,
01529 0x7d2e, 0x7d30, 0x7d31, 0x7d32, 0x7d33, 0x7d34, 0x7d35, 0x7d36,
01530 0x808c, 0x9965, 0x8ff9, 0x6fc0, 0x8ba5, 0x9e21, 0x59ec, 0x7ee9,
01531 0x7f09, 0x5409, 0x6781, 0x68d8, 0x8f91, 0x7c4d, 0x96c6, 0x53ca,
01532 0x6025, 0x75be, 0x6c72, 0x5373, 0x5ac9, 0x7ea7, 0x6324, 0x51e0,
01533 0x810a, 0x5df1, 0x84df, 0x6280, 0x5180, 0x5b63, 0x4f0e, 0x796d,
01534 0x5242, 0x60b8, 0x6d4e, 0x5bc4, 0x5bc2, 0x8ba1, 0x8bb0, 0x65e2,
01535 0x5fcc, 0x9645, 0x5993, 0x7ee7, 0x7eaa, 0x5609, 0x67b7, 0x5939,
01536 0x4f73, 0x5bb6, 0x52a0, 0x835a, 0x988a, 0x8d3e, 0x7532, 0x94be,
01537 0x5047, 0x7a3c, 0x4ef7, 0x67b6, 0x9a7e, 0x5ac1, 0x6b7c, 0x76d1,
01538 0x575a, 0x5c16, 0x7b3a, 0x95f4, 0x714e, 0x517c, 0x80a9, 0x8270,
01539 0x5978, 0x7f04, 0x8327, 0x68c0, 0x67ec, 0x78b1, 0x7877, 0x62e3,
01540 0x6361, 0x7b80, 0x4fed, 0x526a, 0x51cf, 0x8350, 0x69db, 0x9274,
01541 0x8df5, 0x8d31, 0x89c1, 0x952e, 0x7bad, 0x4ef6,
01542 /* 0xbd */
01543 0x7d37, 0x7d38, 0x7d39, 0x7d3a, 0x7d3b, 0x7d3c, 0x7d3d, 0x7d3e,
01544 0x7d3f, 0x7d40, 0x7d41, 0x7d42, 0x7d43, 0x7d44, 0x7d45, 0x7d46,
01545 0x7d47, 0x7d48, 0x7d49, 0x7d4a, 0x7d4b, 0x7d4c, 0x7d4d, 0x7d4e,
01546 0x7d4f, 0x7d50, 0x7d51, 0x7d52, 0x7d53, 0x7d54, 0x7d55, 0x7d56,
01547 0x7d57, 0x7d58, 0x7d59, 0x7d5a, 0x7d5b, 0x7d5c, 0x7d5d, 0x7d5e,
01548 0x7d5f, 0x7d60, 0x7d61, 0x7d62, 0x7d63, 0x7d64, 0x7d65, 0x7d66,
01549 0x7d67, 0x7d68, 0x7d69, 0x7d6a, 0x7d6b, 0x7d6c, 0x7d6d, 0x7d6f,
01550 0x7d70, 0x7d71, 0x7d72, 0x7d73, 0x7d74, 0x7d75, 0x7d76, 0x7d78,
01551 0x7d79, 0x7d7a, 0x7d7b, 0x7d7c, 0x7d7d, 0x7d7e, 0x7d7f, 0x7d80,
01552 0x7d81, 0x7d82, 0x7d83, 0x7d84, 0x7d85, 0x7d86, 0x7d87, 0x7d88,
01553 0x7d89, 0x7d8a, 0x7d8b, 0x7d8c, 0x7d8d, 0x7d8e, 0x7d8f, 0x7d90,
01554 0x7d91, 0x7d92, 0x7d93, 0x7d94, 0x7d95, 0x7d96, 0x7d97, 0x7d98,
01555 0x5065, 0x8230, 0x5251, 0x996f, 0x6e10, 0x6e85, 0x6da7, 0x5efa,
01556 0x50f5, 0x59dc, 0x5c06, 0x6d46, 0x6c5f, 0x7586, 0x848b, 0x6868,
01557 0x5956, 0x8bb2, 0x5320, 0x9171, 0x964d, 0x8549, 0x6912, 0x7901,
01558 0x7126, 0x80f6, 0x4ea4, 0x90ca, 0x6d47, 0x9a84, 0x5a07, 0x56bc,
01559 0x6405, 0x94f0, 0x77eb, 0x4fa5, 0x811a, 0x72e1, 0x89d2, 0x997a,
01560 0x7f34, 0x7ede, 0x527f, 0x6559, 0x9175, 0x8f7f, 0x8f83, 0x53eb,
01561 0x7a96, 0x63ed, 0x63a5, 0x7686, 0x79f8, 0x8857, 0x9636, 0x622a,
01562 0x52ab, 0x8282, 0x6854, 0x6770, 0x6377, 0x776b, 0x7aed, 0x6d01,
01563 0x7ed3, 0x89e3, 0x59d0, 0x6212, 0x85c9, 0x82a5, 0x754c, 0x501f,
01564 0x4ecb, 0x75a5, 0x8bbeb, 0x5c4a, 0x5dfe, 0x7b4b, 0x65a4, 0x91d1,
01565 0x4eca, 0x6d25, 0x895f, 0x7d27, 0x9526, 0x4ec5, 0x8c28, 0x8fdb,
01566 0x9773, 0x664b, 0x7981, 0x8fd1, 0x70ec, 0x6d78,
01567 /* 0xbe */
01568 0x7d99, 0x7d9a, 0x7d9b, 0x7d9c, 0x7d9d, 0x7d9e, 0x7d9f, 0x7da0,
01569 0x7da1, 0x7da2, 0x7da3, 0x7da4, 0x7da5, 0x7da7, 0x7da8, 0x7da9,
01570 0x7daa, 0x7dad, 0x7dac, 0x7daf, 0x7db0, 0x7db1, 0x7db2,
01571 0x7db3, 0x7db4, 0x7db5, 0x7db6, 0x7db7, 0x7db8, 0x7db9, 0x7dba,
```

```
01572 0x7dbb, 0x7dbc, 0x7dbd, 0x7dbe, 0x7dbf, 0x7dc0, 0x7dc1, 0x7dc2,
01573 0x7dc3, 0x7dc4, 0x7dc5, 0x7dc6, 0x7dc7, 0x7dc8, 0x7dc9, 0x7dca,
01574 0x7dcb, 0x7dcc, 0x7dcd, 0x7dce, 0x7dcf, 0x7dd0, 0x7dd1, 0x7dd2,
01575 0x7dd3, 0x7dd4, 0x7dd5, 0x7dd6, 0x7dd7, 0x7dd8, 0x7dd9, 0x7dda,
01576 0x7ddb, 0x7ddc, 0x7ddd, 0x7dde, 0x7ddf, 0x7de0, 0x7de1, 0x7de2,
01577 0x7de3, 0x7de4, 0x7de5, 0x7de6, 0x7de7, 0x7de8, 0x7de9, 0x7dea,
01578 0x7deb, 0x7dec, 0x7ded, 0x7dee, 0x7def, 0x7df0, 0x7df1, 0x7df2,
01579 0x7df3, 0x7df4, 0x7df5, 0x7df6, 0x7df7, 0x7df8, 0x7df9, 0x7dfa,
01580 0x5c3d, 0x52b2, 0x8346, 0x5162, 0x830e, 0x775b, 0x6676, 0x9cb8,
01581 0x4eac, 0x60ca, 0x7cbe, 0x7cb3, 0x7ecf, 0x4e95, 0x8b66, 0x666f,
01582 0x9888, 0x9759, 0x5883, 0x656c, 0x955c, 0x5f84, 0x75c9, 0x9756,
01583 0x7adf, 0x7ade, 0x51c0, 0x70af, 0x7a98, 0x63ea, 0x7a76, 0x7ea0,
01584 0x7396, 0x97ed, 0x4e45, 0x7078, 0x4e5d, 0x9152, 0x53a9, 0x6551,
01585 0x65e7, 0x81fc, 0x8205, 0x548e, 0x5c31, 0x759a, 0x97a0, 0x62d8,
01586 0x72d9, 0x75bd, 0x5c45, 0x9a79, 0x83ca, 0x5c40, 0x5480, 0x77e9,
01587 0x4e3e, 0x6cae, 0x805a, 0x62d2, 0x636e, 0x5de8, 0x5177, 0x8ddd,
01588 0x8e1e, 0x952f, 0x4ff1, 0x53e5, 0x60e7, 0x70ac, 0x5267, 0x6350,
01589 0x9e43, 0x5a1f, 0x5026, 0x7737, 0x5377, 0x7ee2, 0x6485, 0x652b,
01590 0x6289, 0x6398, 0x5014, 0x7235, 0x89c9, 0x51b3, 0x8bc0, 0x7edd,
01591 0x5747, 0x83cc, 0x94a7, 0x519b, 0x541b, 0x5cfb,
01592 /* 0xbf */
01593 0x7dfb, 0x7dfc, 0x7dfd, 0x7dfe, 0x7dff, 0x7e00, 0x7e01, 0x7e02,
01594 0x7e03, 0x7e04, 0x7e05, 0x7e06, 0x7e07, 0x7e08, 0x7e09, 0x7e0a,
01595 0x7e0b, 0x7e0c, 0x7e0d, 0x7e0e, 0x7e0f, 0x7e10, 0x7e11, 0x7e12,
01596 0x7e13, 0x7e14, 0x7e15, 0x7e16, 0x7e17, 0x7e18, 0x7e19, 0x7e1a,
01597 0x7e1b, 0x7e1c, 0x7e1d, 0x7e1e, 0x7e1f, 0x7e20, 0x7e21, 0x7e22,
01598 0x7e23, 0x7e24, 0x7e25, 0x7e26, 0x7e27, 0x7e28, 0x7e29, 0x7e2a,
01599 0x7e2b, 0x7e2c, 0x7e2d, 0x7e2e, 0x7e2f, 0x7e30, 0x7e31, 0x7e32,
01600 0x7e33, 0x7e34, 0x7e35, 0x7e36, 0x7e37, 0x7e38, 0x7e39, 0x7e3a,
01601 0x7e3c, 0x7e3d, 0x7e3e, 0x7e3f, 0x7e40, 0x7e42, 0x7e43, 0x7e44,
01602 0x7e45, 0x7e46, 0x7e48, 0x7e49, 0x7e4a, 0x7e4b, 0x7e4c, 0x7e4d,
01603 0x7e4e, 0x7e4f, 0x7e50, 0x7e51, 0x7e52, 0x7e53, 0x7e54, 0x7e55,
01604 0x7e56, 0x7e57, 0x7e58, 0x7e59, 0x7e5a, 0x7e5b, 0x7e5c, 0x7e5d,
01605 0x4fca, 0x7ae3, 0x6d5a, 0x90e1, 0x9a8f, 0x5580, 0x5496, 0x5361,
01606 0x54af, 0x5f00, 0x63e9, 0x6977, 0x51ef, 0x6168, 0x520a, 0x582a,
01607 0x52d8, 0x574e, 0x780d, 0x770b, 0x5eb7, 0x6177, 0x7ce0, 0x625b,
01608 0x6297, 0x4ea2, 0x7095, 0x8003, 0x62f7, 0x70e4, 0x9760, 0x5777,
01609 0x82db, 0x67ef, 0x68f5, 0x78d5, 0x9897, 0x79d1, 0x58f3, 0x54b3,
01610 0x53ef, 0x6e34, 0x514b, 0x523b, 0x5ba2, 0x8bfe, 0x80a0, 0x5543,
01611 0x57a6, 0x6073, 0x5751, 0x542d, 0x7a7a, 0x6050, 0x5b54, 0x63a7,
01612 0x62a0, 0x53e3, 0x6263, 0x5bc7, 0x67af, 0x54ed, 0x7a9f, 0x82e6,
01613 0x9177, 0x5e93, 0x88e4, 0x5938, 0x57ae, 0x630e, 0x8de8, 0x80ef,
01614 0x5757, 0x7b77, 0x4fa9, 0x5feb, 0x5bbd, 0x6b3e, 0x5321, 0x7b50,
01615 0x72c2, 0x6846, 0x77ff, 0x7736, 0x65f7, 0x51b5, 0x4e8f, 0x76d4,
01616 0x5cbf, 0x7aa5, 0x8475, 0x594e, 0x9b41, 0x5080,
01617 /* 0xc0 */
01618 0x7e5e, 0x7e5f, 0x7e60, 0x7e61, 0x7e62, 0x7e63, 0x7e64, 0x7e65,
01619 0x7e66, 0x7e67, 0x7e68, 0x7e69, 0x7e6a, 0x7e6b, 0x7e6c, 0x7e6d,
01620 0x7e6e, 0x7e6f, 0x7e70, 0x7e71, 0x7e72, 0x7e73, 0x7e74, 0x7e75,
01621 0x7e76, 0x7e77, 0x7e78, 0x7e79, 0x7e7a, 0x7e7b, 0x7e7c, 0x7e7d,
01622 0x7e7e, 0x7e7f, 0x7e80, 0x7e81, 0x7e82, 0x7e83, 0x7e84, 0x7e85, 0x7e86,
01623 0x7e87, 0x7e88, 0x7e89, 0x7e8a, 0x7e8b, 0x7e8c, 0x7e8d, 0x7e8e,
01624 0x7e8f, 0x7e90, 0x7e91, 0x7e92, 0x7e93, 0x7e94, 0x7e95, 0x7e96,
01625 0x7e97, 0x7e98, 0x7e99, 0x7e9a, 0x7e9c, 0x7e9d, 0x7e9e, 0x7e9f,
01626 0x7eb4, 0x7ebb, 0x7ebc, 0x7ed6, 0x7ee4, 0x7eec, 0x7ef9, 0x7f0a,
01627 0x7f10, 0x7f1e, 0x7f37, 0x7f39, 0x7f3b, 0x7f3c, 0x7f3d, 0x7f3e,
01628 0x7f3f, 0x7f40, 0x7f41, 0x7f43, 0x7f46, 0x7f47, 0x7f48, 0x7f49,
01629 0x7f4a, 0x7f4b, 0x7f4c, 0x7f4d, 0x7f4e, 0x7f4f, 0x7f52, 0x7f53,
01630 0x9988, 0x6127, 0x6e83, 0x5764, 0x6606, 0x6346, 0x56f0, 0x62ec,
01631 0x6269, 0x5ed3, 0x9614, 0x5783, 0x62c9, 0x5587, 0x8721, 0x814a,
01632 0x8fa3, 0x5566, 0x83b1, 0x6765, 0x8d56, 0x84dd, 0x5a6a, 0x680f,
01633 0x62e6, 0x7bee, 0x9611, 0x5170, 0x6f9c, 0x8c30, 0x63fd, 0x89c8,
01634 0x61d2, 0x7f06, 0x70c2, 0x6ee5, 0x7405, 0x699a, 0x72fc, 0x5eca,
01635 0x90ce, 0x6717, 0x6d6a, 0x635e, 0x52b3, 0x7262, 0x8001, 0x4f6c,
01636 0x59e5, 0x916a, 0x70d9, 0x6d9d, 0x52d2, 0x4e50, 0x96f7, 0x956d,
01637 0x857e, 0x78ca, 0x7d2f, 0x5121, 0x5792, 0x64c2, 0x808b, 0x7c7b,
01638 0x6cea, 0x68f1, 0x695e, 0x51b7, 0x5398, 0x68a8, 0x7281, 0x9ece,
01639 0x7bf1, 0x72f8, 0x79bb, 0x6f13, 0x7406, 0x674e, 0x91cc, 0x9ca4,
01640 0x793c, 0x8389, 0x8354, 0x540f, 0x6817, 0x4e3d, 0x5389, 0x52b1,
01641 0x783e, 0x5386, 0x5229, 0x5088, 0x4f8b, 0x4fd0,
01642 /* 0xc1 */
01643 0x7f56, 0x7f59, 0x7f5b, 0x7f5c, 0x7f5d, 0x7f5e, 0x7f60, 0x7f63,
01644 0x7f64, 0x7f65, 0x7f66, 0x7f67, 0x7f6b, 0x7f6c, 0x7f6d, 0x7f6f,
01645 0x7f70, 0x7f73, 0x7f75, 0x7f76, 0x7f77, 0x7f78, 0x7f7a, 0x7f7b,
01646 0x7f7c, 0x7f7d, 0x7f7f, 0x7f80, 0x7f82, 0x7f83, 0x7f84, 0x7f85,
01647 0x7f86, 0x7f87, 0x7f88, 0x7f89, 0x7f8b, 0x7f8d, 0x7f8f, 0x7f90,
01648 0x7f91, 0x7f92, 0x7f93, 0x7f95, 0x7f96, 0x7f97, 0x7f98, 0x7f99,
01649 0x7f9b, 0x7f9c, 0x7fa0, 0x7fa2, 0x7fa3, 0x7fa5, 0x7fa6, 0x7fa8,
01650 0x7fa9, 0x7faa, 0x7fab, 0x7fac, 0x7fad, 0x7fae, 0x7fb1, 0x7fb3,
01651 0x7fb4, 0x7fb5, 0x7fb6, 0x7fb7, 0x7fba, 0x7fbb, 0x7fbe, 0x7fc0,
01652 0x7fc2, 0x7fc3, 0x7fc4, 0x7fc6, 0x7fc7, 0x7fc8, 0x7fc9, 0x7fcb,
01653 0x7fcd, 0x7fcd, 0x7fd0, 0x7fd1, 0x7fd2, 0x7fd3, 0x7fde, 0x7fd7,
01654 0x7fd9, 0x7fdb, 0x7fdc, 0x7fdd, 0x7fde, 0x7fe2, 0x7fe3,
01655 0x75e2, 0x7acb, 0x7c92, 0x6ca5, 0x96b6, 0x529b, 0x7483, 0x54e9,
01656 0x4fe9, 0x8054, 0x83b2, 0x8fde, 0x9570, 0x5ec9, 0x601c, 0x6d9f,
01657 0x5e18, 0x655b, 0x8138, 0x94fe, 0x604b, 0x70bc, 0x7ec3, 0x7cae,
01658 0x51c9, 0x6881, 0x7cb1, 0x826f, 0x4e24, 0x8f86, 0x91cf, 0x667e,
```

```
01659 0x4eae, 0x8c05, 0x64a9, 0x804a, 0x50da, 0x7597, 0x71ce, 0x5be5,
01660 0x8fbd, 0x6f66, 0x4e86, 0x6482, 0x9563, 0x5ed6, 0x6599, 0x5217,
01661 0x88c2, 0x70c8, 0x52a3, 0x730e, 0x7433, 0x6797, 0x78f7, 0x9716,
01662 0x4e34, 0x90bb, 0x9cde, 0x6dcb, 0x51db, 0x8d41, 0x541d, 0x62ce,
01663 0x73b2, 0x83f1, 0x96f6, 0x9f84, 0x94c3, 0x4f36, 0x7f9a, 0x51cc,
01664 0x7075, 0x9675, 0x5cad, 0x9886, 0x53e6, 0x4ee4, 0x6e9c, 0x7409,
01665 0x69b4, 0x786b, 0x998f, 0x7559, 0x5218, 0x7624, 0x6d41, 0x67f3,
01666 0x516d, 0x9f99, 0x804b, 0x5499, 0x7b3c, 0x7abf,
01667 /* 0xc2 */
01668 0x7fe4, 0x7fe7, 0x7fe8, 0x7fea, 0x7feb, 0x7fec, 0x7fed, 0x7fef,
01669 0x7ff2, 0x7ff4, 0x7ff5, 0x7ff6, 0x7ff7, 0x7ff8, 0x7ff9, 0x7ffa,
01670 0x7ffd, 0x7ffe, 0x7fff, 0x8002, 0x8007, 0x8008, 0x8009, 0x800a,
01671 0x800e, 0x800f, 0x8011, 0x8013, 0x801a, 0x801b, 0x801d, 0x801e,
01672 0x801f, 0x8021, 0x8023, 0x8024, 0x802b, 0x802c, 0x802d, 0x802e,
01673 0x802f, 0x8030, 0x8032, 0x8033, 0x8034, 0x8039, 0x803a, 0x803c,
01674 0x8040, 0x8041, 0x8044, 0x8045, 0x8047, 0x8048, 0x8049, 0x804e,
01675 0x804f, 0x8050, 0x8051, 0x8053, 0x8055, 0x8056, 0x8057, 0x8059,
01676 0x805b, 0x805c, 0x805d, 0x805e, 0x805f, 0x8060, 0x8061, 0x8062,
01677 0x8063, 0x8064, 0x8065, 0x8066, 0x8067, 0x8068, 0x806b, 0x806c,
01678 0x806d, 0x806e, 0x806f, 0x8070, 0x8072, 0x8073, 0x8074, 0x8075,
01679 0x8076, 0x8077, 0x8078, 0x8079, 0x807a, 0x807b, 0x807c, 0x807d,
01680 0x9686, 0x5784, 0x62e2, 0x9647, 0x697c, 0x5a04, 0x6402, 0x7bd3,
01681 0x6f0f, 0x964b, 0x82a6, 0x5362, 0x9885, 0x5e90, 0x7089, 0x63b3,
01682 0x5364, 0x864f, 0x9c81, 0x9e93, 0x788c, 0x9732, 0x8def, 0x8d42,
01683 0x9e7f, 0x6f5e, 0x7984, 0x5f55, 0x9646, 0x622e, 0x9a74, 0x5415,
01684 0x94dd, 0x4fa3, 0x65c5, 0x5c65, 0x5c61, 0x7f15, 0x8651, 0x6c2f,
01685 0x5f8b, 0x7387, 0x6ee4, 0x7eff, 0x5ce6, 0x631b, 0x5b6a, 0x6ee6,
01686 0x5375, 0x4e71, 0x63a0, 0x7565, 0x62a1, 0x8f6e, 0x4f26, 0x4ed1,
01687 0x6ca6, 0x7eb6, 0x8bba, 0x841d, 0x87ba, 0x7f57, 0x903b, 0x9523,
01688 0x7ba9, 0x9aa1, 0x88f8, 0x843d, 0x6dlb, 0x9a86, 0x7edc, 0x5988,
01689 0x9ebb, 0x739b, 0x7801, 0x8682, 0x9a6c, 0x9a82, 0x561b, 0x5417,
01690 0x57cb, 0x4e70, 0x9ea6, 0x5356, 0x8fc8, 0x8109, 0x7792, 0x9992,
01691 0x86ee, 0x6ee1, 0x8513, 0x66fc, 0x6162, 0x6f2b,
01692 /* 0xc3 */
01693 0x807e, 0x8081, 0x8082, 0x8085, 0x8088, 0x808a, 0x808d, 0x808e,
01694 0x808f, 0x8090, 0x8091, 0x8092, 0x8094, 0x8095, 0x8097, 0x8099,
01695 0x809e, 0x80a3, 0x80a6, 0x80a7, 0x80a8, 0x80ac, 0x80b0, 0x80b3,
01696 0x80b5, 0x80b6, 0x80b8, 0x80b9, 0x80bb, 0x80c5, 0x80c7, 0x80c8,
01697 0x80c9, 0x80ca, 0x80cb, 0x80cf, 0x80d0, 0x80d1, 0x80d2, 0x80d3,
01698 0x80d4, 0x80d5, 0x80d8, 0x80df, 0x80e0, 0x80e2, 0x80e3, 0x80e6,
01699 0x80ee, 0x80f5, 0x80f7, 0x80f9, 0x80fb, 0x80fe, 0x80ff, 0x8100,
01700 0x8101, 0x8103, 0x8104, 0x8105, 0x8107, 0x8108, 0x810b, 0x810c,
01701 0x8115, 0x8117, 0x8119, 0x811b, 0x811c, 0x811d, 0x811f, 0x8120,
01702 0x8121, 0x8122, 0x8123, 0x8124, 0x8125, 0x8126, 0x8127, 0x8128,
01703 0x8129, 0x812a, 0x812b, 0x812d, 0x812e, 0x8130, 0x8133, 0x8134,
01704 0x8135, 0x8137, 0x8139, 0x813a, 0x813b, 0x813c, 0x813d, 0x813f,
01705 0x8c29, 0x8292, 0x832b, 0x76f2, 0x6c13, 0x5fd9, 0x83bd, 0x732b,
01706 0x8305, 0x951a, 0x6bdb, 0x77db, 0x94c6, 0x536f, 0x8302, 0x5192,
01707 0x5e3d, 0x8c8c, 0x8d38, 0x4e48, 0x73ab, 0x679a, 0x6885, 0x9176,
01708 0x9709, 0x7164, 0x6ca1, 0x7709, 0x5a92, 0x9541, 0x6bcf, 0x7f8e,
01709 0x6627, 0x5bd0, 0x59b9, 0x5a9a, 0x95e8, 0x95f7, 0x4eec, 0x840c,
01710 0x8499, 0x6aac, 0x76df, 0x9530, 0x731b, 0x68a6, 0x5b5f, 0x772f,
01711 0x919a, 0x9761, 0x7cdc, 0x8ff7, 0x8c1c, 0x5f25, 0x7c73, 0x79d8,
01712 0x89c5, 0x6ccc, 0x871c, 0x5bc6, 0x5e42, 0x68c9, 0x7720, 0x7ef5,
01713 0x5195, 0x514d, 0x52c9, 0x5a29, 0x7f05, 0x9762, 0x82d7, 0x63cf,
01714 0x7784, 0x85d0, 0x79d2, 0x6e3a, 0x5e99, 0x5999, 0x8511, 0x706d,
01715 0x6c11, 0x62bf, 0x76bf, 0x654f, 0x60af, 0x95fd, 0x660e, 0x879f,
01716 0x9e23, 0x94ed, 0x540d, 0x547d, 0x8c2c, 0x6478,
01717 /* 0xc4 */
01718 0x8140, 0x8141, 0x8142, 0x8143, 0x8144, 0x8145, 0x8147, 0x8149,
01719 0x814d, 0x814e, 0x814f, 0x8152, 0x8156, 0x8157, 0x8158, 0x815b,
01720 0x815c, 0x815d, 0x815e, 0x815f, 0x8161, 0x8162, 0x8163, 0x8164,
01721 0x8166, 0x8168, 0x816a, 0x816b, 0x816c, 0x816f, 0x8172, 0x8173,
01722 0x8175, 0x8176, 0x8177, 0x8178, 0x8181, 0x8183, 0x8184, 0x8185,
01723 0x8186, 0x8187, 0x8189, 0x818b, 0x818c, 0x818d, 0x818e, 0x8190,
01724 0x8192, 0x8193, 0x8194, 0x8195, 0x8196, 0x8197, 0x8199, 0x819a,
01725 0x819e, 0x819f, 0x81a0, 0x81a1, 0x81a2, 0x81a4, 0x81a5, 0x81a7,
01726 0x81a9, 0x81ab, 0x81ac, 0x81ad, 0x81ae, 0x81af, 0x81b0, 0x81b1,
01727 0x81b2, 0x81b4, 0x81b5, 0x81b6, 0x81b7, 0x81b8, 0x81b9, 0x81bc,
01728 0x81bd, 0x81be, 0x81bf, 0x81c4, 0x81c5, 0x81c7, 0x81c8, 0x81c9,
01729 0x81cb, 0x81cd, 0x81ce, 0x81cf, 0x81d0, 0x81d1, 0x81d2, 0x81d3,
01730 0x6479, 0x8611, 0x6a21, 0x819c, 0x78e8, 0x6469, 0x9b54, 0x62b9,
01731 0x672b, 0x83ab, 0x58a8, 0x9ed8, 0x6cab, 0x6f20, 0x5bde, 0x964c,
01732 0x8c0b, 0x725f, 0x67d0, 0x62c7, 0x7261, 0x4ea9, 0x59c6, 0x6bcd,
01733 0x5893, 0x66ae, 0x5e55, 0x52df, 0x6155, 0x6728, 0x76ee, 0x7766,
01734 0x7267, 0x7a46, 0x62ff, 0x54ea, 0x5450, 0x94a0, 0x90a3, 0x5a1c,
01735 0x7eb3, 0x6c16, 0x4e43, 0x5976, 0x8010, 0x5948, 0x5357, 0x7537,
01736 0x96be, 0x56ca, 0x6320, 0x8111, 0x607c, 0x95f9, 0x6dd6, 0x5462,
01737 0x9981, 0x5185, 0x5ae9, 0x80fd, 0x59ae, 0x9713, 0x502a, 0x6ce5,
01738 0x5c3c, 0x62df, 0x4f60, 0x533f, 0x817b, 0x9006, 0x6eba, 0x852b,
01739 0x62c8, 0x5e74, 0x78be, 0x64b5, 0x637b, 0x5ff5, 0x5a18, 0x917f,
01740 0x9e1f, 0x5c3f, 0x634f, 0x8042, 0x5b7d, 0x556e, 0x954a, 0x954d,
01741 0x6d85, 0x60a8, 0x67e0, 0x72de, 0x51dd, 0x5b81,
01742 /* 0xc5 */
01743 0x81d4, 0x81d5, 0x81d6, 0x81d7, 0x81d8, 0x81d9, 0x81da, 0x81db,
01744 0x81dc, 0x81dd, 0x81de, 0x81df, 0x81e0, 0x81e1, 0x81e2, 0x81e4,
01745 0x81e5, 0x81e6, 0x81e8, 0x81e9, 0x81eb, 0x81ee, 0x81ef, 0x81f0,
```

```

01746 0x81f1, 0x81f2, 0x81f5, 0x81f6, 0x81f7, 0x81f8, 0x81f9, 0x81fa,
01747 0x81fd, 0x81ff, 0x8203, 0x8207, 0x8208, 0x8209, 0x820a, 0x820b,
01748 0x820e, 0x820f, 0x8211, 0x8213, 0x8215, 0x8216, 0x8217, 0x8218,
01749 0x8219, 0x821a, 0x821d, 0x8220, 0x8224, 0x8225, 0x8226, 0x8227,
01750 0x8229, 0x822e, 0x8232, 0x823a, 0x823c, 0x823d, 0x823f, 0x8240,
01751 0x8241, 0x8242, 0x8243, 0x8245, 0x8246, 0x8248, 0x824a, 0x824c,
01752 0x824d, 0x824e, 0x8250, 0x8251, 0x8252, 0x8253, 0x8254, 0x8255,
01753 0x8256, 0x8257, 0x8259, 0x825b, 0x825c, 0x825d, 0x825e, 0x8260,
01754 0x8261, 0x8262, 0x8263, 0x8264, 0x8265, 0x8266, 0x8267, 0x8269,
01755 0x62e7, 0x6cde, 0x725b, 0x626d, 0x94ae, 0x7ebd, 0x8113, 0x6d53,
01756 0x519c, 0x5f04, 0x5974, 0x52aa, 0x6012, 0x5973, 0x6696, 0x8650,
01757 0x759f, 0x632a, 0x61e6, 0x7cef, 0x8bfa, 0x54e6, 0x6b27, 0x9e25,
01758 0x6bb4, 0x85d5, 0x5455, 0x5076, 0x6ca4, 0x556a, 0x8db4, 0x722c,
01759 0x5e15, 0x6015, 0x7436, 0x62cd, 0x6392, 0x724c, 0x5f98, 0x6e43,
01760 0x6d3e, 0x6500, 0x6f58, 0x76d8, 0x78d0, 0x76fc, 0x7554, 0x5224,
01761 0x53db, 0x4e53, 0x5e9e, 0x65c1, 0x802a, 0x80d6, 0x629b, 0x5486,
01762 0x5228, 0x70ae, 0x888d, 0x8dd1, 0x6ce1, 0x5478, 0x80da, 0x57f9,
01763 0x88f4, 0x8d54, 0x966a, 0x914d, 0x4f69, 0x6c9b, 0x55b7, 0x76c6,
01764 0x7830, 0x62a8, 0x70f9, 0x6f8e, 0x5f6d, 0x84ec, 0x68da, 0x787c,
01765 0x7bf7, 0x81a8, 0x670b, 0x9e4f, 0x6367, 0x78b0, 0x576f, 0x7812,
01766 0x9739, 0x6279, 0x62ab, 0x5288, 0x7435, 0x6bd7,
01767 /* 0xc6 */
01768 0x826a, 0x826b, 0x826c, 0x826d, 0x8271, 0x8275, 0x8276, 0x8277,
01769 0x8278, 0x827b, 0x827c, 0x8280, 0x8281, 0x8283, 0x8285, 0x8286,
01770 0x8287, 0x8289, 0x828c, 0x8290, 0x8293, 0x8294, 0x8295, 0x8296,
01771 0x829a, 0x829b, 0x829e, 0x82a0, 0x82a2, 0x82a3, 0x82a7, 0x82b2,
01772 0x82b5, 0x82b6, 0x82ba, 0x82bb, 0x82bc, 0x82bf, 0x82c0, 0x82c2,
01773 0x82c3, 0x82c5, 0x82c6, 0x82c9, 0x82d0, 0x82d6, 0x82d9, 0x82da,
01774 0x82dd, 0x82e2, 0x82e7, 0x82e8, 0x82e9, 0x82ea, 0x82ec, 0x82ed,
01775 0x82ee, 0x82f0, 0x82f2, 0x82f3, 0x82f5, 0x82f6, 0x82f8, 0x82fa,
01776 0x82fc, 0x82fd, 0x82fe, 0x82ff, 0x8300, 0x830a, 0x830b, 0x830d,
01777 0x8310, 0x8312, 0x8313, 0x8316, 0x8318, 0x8319, 0x831d, 0x831e,
01778 0x831f, 0x8320, 0x8321, 0x8322, 0x8323, 0x8324, 0x8325, 0x8326,
01779 0x8329, 0x832a, 0x832e, 0x8330, 0x8332, 0x8337, 0x833b, 0x833d,
01780 0x5564, 0x813e, 0x75b2, 0x76ae, 0x5339, 0x75de, 0x50fb, 0x5c41,
01781 0x8b6c, 0x7bc7, 0x504f, 0x7247, 0x9a97, 0x98d8, 0x6f02, 0x74e2,
01782 0x7968, 0x6487, 0x77a5, 0x62fc, 0x9891, 0x8d2b, 0x54c1, 0x8058,
01783 0x4e52, 0x576a, 0x82f9, 0x840d, 0x5e73, 0x51ed, 0x74f6, 0x8bc4,
01784 0x5c4f, 0x5761, 0x6cfc, 0x9887, 0x5a46, 0x7834, 0x9b44, 0x8feb,
01785 0x7c95, 0x5256, 0x6251, 0x94fa, 0x4ec6, 0x8386, 0x8461, 0x83e9,
01786 0x84b2, 0x57d4, 0x6734, 0x5703, 0x666e, 0x6d66, 0x8c31, 0x66dd,
01787 0x7011, 0x671f, 0x6b3a, 0x6816, 0x621a, 0x59bb, 0x4e03, 0x51c4,
01788 0x6f06, 0x67d2, 0x6c8f, 0x5176, 0x68cb, 0x5947, 0x6b67, 0x7566,
01789 0x5d0e, 0x8110, 0x9f50, 0x65d7, 0x7948, 0x7941, 0x9a91, 0x8d77,
01790 0x5c82, 0x4e5e, 0x4f01, 0x542f, 0x5951, 0x780c, 0x5668, 0x6c14,
01791 0x8fc4, 0x5f03, 0x6c7d, 0x6ce3, 0x8bab, 0x6390,
01792 /* 0xc7 */
01793 0x833e, 0x833f, 0x8341, 0x8342, 0x8344, 0x8345, 0x8348, 0x834a,
01794 0x834b, 0x834c, 0x834d, 0x834e, 0x8353, 0x8355, 0x8356, 0x8357,
01795 0x8358, 0x8359, 0x835d, 0x8362, 0x8370, 0x8371, 0x8372, 0x8373,
01796 0x8374, 0x8375, 0x8376, 0x8379, 0x837a, 0x837e, 0x837f, 0x8380,
01797 0x8381, 0x8382, 0x8383, 0x8384, 0x8387, 0x8388, 0x838a, 0x838b,
01798 0x838c, 0x838d, 0x838f, 0x8390, 0x8391, 0x8394, 0x8395, 0x8396,
01799 0x8397, 0x8399, 0x839a, 0x839d, 0x839f, 0x83a1, 0x83a2, 0x83a3,
01800 0x83a4, 0x83a5, 0x83a6, 0x83a7, 0x83ac, 0x83ad, 0x83ae, 0x83af,
01801 0x83b5, 0x83bb, 0x83be, 0x83bf, 0x83c2, 0x83c3, 0x83c4, 0x83c6,
01802 0x83c8, 0x83c9, 0x83cb, 0x83cd, 0x83ce, 0x83d0, 0x83d1, 0x83d2,
01803 0x83d3, 0x83d5, 0x83d7, 0x83d9, 0x83da, 0x83db, 0x83de, 0x83e2,
01804 0x83e3, 0x83e4, 0x83e6, 0x83e7, 0x83e8, 0x83eb, 0x83ec, 0x83ed,
01805 0x6070, 0x6d3d, 0x7275, 0x6266, 0x948e, 0x94c5, 0x5343, 0x8fc1,
01806 0x7b7e, 0x4edf, 0x8c26, 0x4e7e, 0x9ed4, 0x94b1, 0x94b3, 0x524d,
01807 0x6f5c, 0x9063, 0x6d45, 0x8c34, 0x5811, 0x5d4c, 0x6b20, 0x6b49,
01808 0x67aa, 0x545b, 0x8154, 0x7f8c, 0x5899, 0x8537, 0x5f3a, 0x62a2,
01809 0x6a47, 0x9539, 0x6572, 0x6084, 0x6865, 0x77a7, 0x4e54, 0x4fa8,
01810 0x5de7, 0x9798, 0x64ac, 0x7fd8, 0x5ced, 0x4fcf, 0x7a8d, 0x5207,
01811 0x8304, 0x4e14, 0x602f, 0x7a83, 0x94a6, 0x4fb5, 0x4eb2, 0x79e6,
01812 0x7434, 0x52e4, 0x82b9, 0x64d2, 0x79bd, 0x5bdd, 0x6c81, 0x9752,
01813 0x8f7b, 0x6c22, 0x503e, 0x537f, 0x6e05, 0x64ce, 0x6674, 0x6c30,
01814 0x60c5, 0x8977, 0x8bf7, 0x5e86, 0x743c, 0x7a77, 0x79cb, 0x4e18,
01815 0x90b1, 0x7403, 0x6c42, 0x56da, 0x914b, 0x6cc5, 0x8d8b, 0x533a,
01816 0x86c6, 0x66f2, 0x8eaf, 0x5c48, 0x9a71, 0x6e20,
01817 /* 0xc8 */
01818 0x83ee, 0x83ef, 0x83f3, 0x83f4, 0x83f5, 0x83f6, 0x83f7, 0x83fa,
01819 0x83fb, 0x83fc, 0x83fd, 0x83fe, 0x8400, 0x8402, 0x8405, 0x8407,
01820 0x8408, 0x8409, 0x840a, 0x8410, 0x8412, 0x8413, 0x8414, 0x8415,
01821 0x8416, 0x8417, 0x8419, 0x841a, 0x841b, 0x841e, 0x841f, 0x8420,
01822 0x8421, 0x8422, 0x8423, 0x8429, 0x842a, 0x842b, 0x842c, 0x842d,
01823 0x842e, 0x842f, 0x8430, 0x8432, 0x8433, 0x8434, 0x8435, 0x8436,
01824 0x8437, 0x8439, 0x843a, 0x843b, 0x843e, 0x843f, 0x8440, 0x8441,
01825 0x8442, 0x8443, 0x8444, 0x8445, 0x8447, 0x8448, 0x8449, 0x844a,
01826 0x844b, 0x844c, 0x844d, 0x844e, 0x844f, 0x8450, 0x8452, 0x8453,
01827 0x8454, 0x8455, 0x8456, 0x8458, 0x845d, 0x845e, 0x845f, 0x8460,
01828 0x8462, 0x8464, 0x8465, 0x8466, 0x8467, 0x8468, 0x846a, 0x846e,
01829 0x846f, 0x8470, 0x8472, 0x8474, 0x8477, 0x8479, 0x847b, 0x847c,
01830 0x53d6, 0x5a36, 0x9f8b, 0x8da3, 0x53bb, 0x5708, 0x98a7, 0x6743,
01831 0x919b, 0x6cc9, 0x5168, 0x75ca, 0x62f3, 0x72ac, 0x5238, 0x529d,
01832 0x7f3a, 0x7094, 0x7638, 0x5374, 0x9e4a, 0x69b7, 0x786e, 0x96c0,

```

```
01833 0x88d9, 0x7fa4, 0x7136, 0x71c3, 0x5189, 0x67d3, 0x74e4, 0x58e4,
01834 0x6518, 0x56b7, 0x8ba9, 0x9976, 0x6270, 0x7ed5, 0x60f9, 0x70ed,
01835 0x58ec, 0x4ec1, 0x4eba, 0x5fcd, 0x97e7, 0x4efb, 0x8ba4, 0x5203,
01836 0x598a, 0x7eab, 0x6254, 0x4ecd, 0x65e5, 0x620e, 0x8338, 0x84c9,
01837 0x8363, 0x878d, 0x7194, 0x6eb6, 0x5bb9, 0x7ed2, 0x5197, 0x63c9,
01838 0x67d4, 0x8089, 0x8339, 0x8815, 0x5112, 0x5b7a, 0x5982, 0x8fb1,
01839 0x4e73, 0x6c5d, 0x5165, 0x8925, 0x8f6f, 0x962e, 0x854a, 0x745e,
01840 0x9510, 0x95f0, 0x6da6, 0x82e5, 0x5f31, 0x6492, 0x6d12, 0x8428,
01841 0x816e, 0x9cc3, 0x585e, 0x8d5b, 0x4e09, 0x53c1,
01842 /* 0xc9 */
01843 0x847d, 0x847e, 0x847f, 0x8480, 0x8481, 0x8483, 0x8484, 0x8485,
01844 0x8486, 0x848a, 0x848d, 0x848f, 0x8490, 0x8491, 0x8492, 0x8493,
01845 0x8494, 0x8495, 0x8496, 0x8498, 0x849a, 0x849b, 0x849d, 0x849e,
01846 0x849f, 0x84a0, 0x84a2, 0x84a3, 0x84a4, 0x84a5, 0x84a6, 0x84a7,
01847 0x84a8, 0x84a9, 0x84aa, 0x84ab, 0x84ac, 0x84ad, 0x84ae, 0x84b0,
01848 0x84b1, 0x84b3, 0x84b5, 0x84b6, 0x84b7, 0x84bb, 0x84bc, 0x84be,
01849 0x84c0, 0x84c2, 0x84c3, 0x84c5, 0x84c6, 0x84c7, 0x84c8, 0x84cb,
01850 0x84cc, 0x84ce, 0x84cf, 0x84d2, 0x84d4, 0x84d5, 0x84d7, 0x84d8,
01851 0x84d9, 0x84da, 0x84db, 0x84dc, 0x84de, 0x84e1, 0x84e2, 0x84e4,
01852 0x84e7, 0x84e8, 0x84e9, 0x84ea, 0x84eb, 0x84ed, 0x84ee, 0x84ef,
01853 0x84f1, 0x84f2, 0x84f3, 0x84f4, 0x84f5, 0x84f6, 0x84f7, 0x84f8,
01854 0x84f9, 0x84fa, 0x84fb, 0x84fd, 0x84fe, 0x8500, 0x8501, 0x8502,
01855 0x4f1e, 0x6563, 0x6851, 0x55d3, 0x4e27, 0x6414, 0x9a9a, 0x626b,
01856 0x5ac2, 0x745f, 0x8272, 0x6da9, 0x68ee, 0x50e7, 0x838e, 0x7802,
01857 0x6740, 0x5239, 0x6c99, 0x7eb1, 0x50bb, 0x5565, 0x715e, 0x7b5b,
01858 0x6652, 0x73ca, 0x82eb, 0x6749, 0x5c71, 0x5220, 0x717d, 0x886b,
01859 0x95ea, 0x9655, 0x64c5, 0x8d61, 0x81b3, 0x5584, 0x6c55, 0x6247,
01860 0x7f2e, 0x5892, 0x4f24, 0x5546, 0x8d4f, 0x664c, 0x4e0a, 0x5c1a,
01861 0x88f3, 0x68a2, 0x634e, 0x7a0d, 0x70e7, 0x828d, 0x52fa, 0x97f6,
01862 0x5c11, 0x54e8, 0x90b5, 0x7ecd, 0x5962, 0x8d4a, 0x86c7, 0x820c,
01863 0x820d, 0x8d66, 0x6444, 0x5c04, 0x6151, 0x6d89, 0x793e, 0x8bbe,
01864 0x7837, 0x7533, 0x547b, 0x4f38, 0x8eab, 0x6df1, 0x5a20, 0x7ec5,
01865 0x795e, 0x6c88, 0x5ba1, 0x5a76, 0x751a, 0x80be, 0x614e, 0x6e17,
01866 0x58f0, 0x751f, 0x7525, 0x7272, 0x5347, 0x7ef3,
01867 /* 0xca */
01868 0x8503, 0x8504, 0x8505, 0x8506, 0x8507, 0x8508, 0x8509, 0x850a,
01869 0x850b, 0x850d, 0x850e, 0x850f, 0x8510, 0x8512, 0x8514, 0x8515,
01870 0x8516, 0x8518, 0x8519, 0x851b, 0x851c, 0x851d, 0x851e, 0x8520,
01871 0x8522, 0x8523, 0x8524, 0x8525, 0x8526, 0x8527, 0x8528, 0x8529,
01872 0x852a, 0x852d, 0x852e, 0x852f, 0x8530, 0x8531, 0x8532, 0x8533,
01873 0x8534, 0x8535, 0x8536, 0x853e, 0x853f, 0x8540, 0x8541, 0x8542,
01874 0x8544, 0x8545, 0x8546, 0x8547, 0x854b, 0x854c, 0x854d, 0x854e,
01875 0x854f, 0x8550, 0x8551, 0x8552, 0x8553, 0x8554, 0x8555, 0x8557,
01876 0x8558, 0x855a, 0x855b, 0x855c, 0x855d, 0x855f, 0x8560, 0x8561,
01877 0x8562, 0x8563, 0x8565, 0x8566, 0x8567, 0x8569, 0x856a, 0x856b,
01878 0x856c, 0x856d, 0x856e, 0x856f, 0x8570, 0x8571, 0x8573, 0x8575,
01879 0x8576, 0x8577, 0x8578, 0x857c, 0x857d, 0x857f, 0x8580, 0x8581,
01880 0x7701, 0x76db, 0x5269, 0x80dc, 0x5723, 0x5e08, 0x5931, 0x72ee,
01881 0x65bd, 0x6e7f, 0x8bd7, 0x5c38, 0x8671, 0x5341, 0x77f3, 0x62fe,
01882 0x65f6, 0x4ec0, 0x98df, 0x8680, 0x5b9e, 0x8bc6, 0x53f2, 0x77e2,
01883 0x4f7f, 0x5c4e, 0x9a76, 0x59cb, 0x5f0f, 0x793a, 0x58eb, 0x4e16,
01884 0x67ff, 0x4e8b, 0x62ed, 0x8a93, 0x901d, 0x52bf, 0x662f, 0x55dc,
01885 0x566c, 0x9002, 0x4ed5, 0x4f8d, 0x91ca, 0x9970, 0x6c0f, 0x5e02,
01886 0x6043, 0x5ba4, 0x89c6, 0x8bd5, 0x6536, 0x624b, 0x9996, 0x5b88,
01887 0x5bff, 0x6388, 0x552e, 0x53d7, 0x7626, 0x517d, 0x852c, 0x67a2,
01888 0x68b3, 0x6b8a, 0x6292, 0x8f93, 0x53d4, 0x8212, 0x6dd1, 0x758f,
01889 0x4e66, 0x8d4e, 0x5b70, 0x719f, 0x85af, 0x6691, 0x66d9, 0x7f72,
01890 0x8700, 0x9ecd, 0x9f20, 0x5c5e, 0x672f, 0x8ff0, 0x6811, 0x675f,
01891 0x620d, 0x7ad6, 0x5885, 0x5eb6, 0x6570, 0x6f31,
01892 /* 0xcb */
01893 0x8582, 0x8583, 0x8586, 0x8588, 0x8589, 0x858a, 0x858b, 0x858c,
01894 0x858d, 0x858e, 0x8590, 0x8591, 0x8592, 0x8593, 0x8594, 0x8595,
01895 0x8596, 0x8597, 0x8598, 0x8599, 0x859a, 0x859d, 0x859e, 0x859f,
01896 0x85a0, 0x85a1, 0x85a2, 0x85a3, 0x85a5, 0x85a6, 0x85a7, 0x85a9,
01897 0x85ab, 0x85ac, 0x85ad, 0x85b1, 0x85b2, 0x85b3, 0x85b4, 0x85b5,
01898 0x85b6, 0x85b8, 0x85ba, 0x85bb, 0x85bc, 0x85bd, 0x85be, 0x85bf,
01899 0x85c0, 0x85c2, 0x85c3, 0x85c4, 0x85c5, 0x85c6, 0x85c7, 0x85c8,
01900 0x85ca, 0x85cb, 0x85cc, 0x85cd, 0x85ce, 0x85d1, 0x85d2, 0x85d4,
01901 0x85d6, 0x85d7, 0x85d8, 0x85d9, 0x85da, 0x85db, 0x85dd, 0x85de,
01902 0x85df, 0x85e0, 0x85e1, 0x85e2, 0x85e3, 0x85e5, 0x85e6, 0x85e7,
01903 0x85e8, 0x85ea, 0x85eb, 0x85ec, 0x85ed, 0x85ee, 0x85ef, 0x85f0,
01904 0x85f1, 0x85f2, 0x85f3, 0x85f4, 0x85f5, 0x85f6, 0x85f7, 0x85f8,
01905 0x6055, 0x5237, 0x800d, 0x6454, 0x8870, 0x7529, 0x5e05, 0x6813,
01906 0x62f4, 0x971c, 0x53cc, 0x723d, 0x8c01, 0x6c34, 0x7761, 0x7a0e,
01907 0x542e, 0x77ac, 0x987a, 0x821c, 0x8bf4, 0x7855, 0x6714, 0x70c1,
01908 0x65af, 0x6495, 0x5636, 0x601d, 0x79c1, 0x53f8, 0x4e1d, 0x6b7b,
01909 0x8086, 0x5bfa, 0x553e, 0x56db, 0x4f3a, 0x4f3c, 0x9972, 0x5df3,
01910 0x677e, 0x8038, 0x6002, 0x9882, 0x9001, 0x5b8b, 0x8bbc, 0x8bf5,
01911 0x641c, 0x8258, 0x64de, 0x55fd, 0x82cf, 0x9165, 0x4fd7, 0x7d20,
01912 0x901f, 0x7c9f, 0x50f3, 0x5851, 0x6eaf, 0x5bbf, 0x8bc9, 0x8083,
01913 0x9178, 0x849c, 0x7b97, 0x867d, 0x968b, 0x968f, 0x7ee5, 0x9ad3,
01914 0x788e, 0x5c81, 0x7a57, 0x9042, 0x96a7, 0x795f, 0x5b59, 0x635f,
01915 0x7b0b, 0x84d1, 0x68ad, 0x5506, 0x7f29, 0x7410, 0x7d22, 0x9501,
01916 0x6240, 0x584c, 0x4ed6, 0x5b83, 0x5979, 0x5854,
01917 /* 0xcc */
01918 0x85f9, 0x85fa, 0x85fc, 0x85fd, 0x85fe, 0x8600, 0x8601, 0x8602,
01919 0x8603, 0x8604, 0x8606, 0x8607, 0x8608, 0x8609, 0x860a, 0x860b,
```

```

01920 0x860c, 0x860d, 0x860e, 0x860f, 0x8610, 0x8612, 0x8613, 0x8614,
01921 0x8615, 0x8617, 0x8618, 0x8619, 0x861a, 0x861b, 0x861c, 0x861d,
01922 0x861e, 0x861f, 0x8620, 0x8621, 0x8622, 0x8623, 0x8624, 0x8625,
01923 0x8626, 0x8628, 0x862a, 0x862b, 0x862c, 0x862d, 0x862e, 0x862f,
01924 0x8630, 0x8631, 0x8632, 0x8633, 0x8634, 0x8635, 0x8636, 0x8637,
01925 0x8639, 0x863a, 0x863b, 0x863d, 0x863e, 0x863f, 0x8640, 0x8641,
01926 0x8642, 0x8643, 0x8644, 0x8645, 0x8646, 0x8647, 0x8648, 0x8649,
01927 0x864a, 0x864b, 0x864c, 0x8652, 0x8653, 0x8655, 0x8656, 0x8657,
01928 0x8658, 0x8659, 0x865b, 0x865c, 0x865d, 0x865f, 0x8660, 0x8661,
01929 0x8663, 0x8664, 0x8665, 0x8666, 0x8667, 0x8668, 0x8669, 0x866a,
01930 0x736d, 0x631e, 0x8e4b, 0x8e0f, 0x80ce, 0x82d4, 0x62ac, 0x53f0,
01931 0x6cf0, 0x915e, 0x592a, 0x6001, 0x6c70, 0x574d, 0x644a, 0x8d2a,
01932 0x762b, 0x6ee9, 0x575b, 0x6a80, 0x75f0, 0x6f6d, 0x8c2d, 0x8c08,
01933 0x5766, 0x6bef, 0x8892, 0x78b3, 0x63a2, 0x53f9, 0x70ad, 0x6c64,
01934 0x5858, 0x642a, 0x5802, 0x68e0, 0x819b, 0x5510, 0x7cd6, 0x5018,
01935 0x8eba, 0x6dcc, 0x8d9f, 0x70eb, 0x638f, 0x6d9b, 0x6ed4, 0x7ee6,
01936 0x8404, 0x6843, 0x9003, 0x6dd8, 0x9676, 0x8ba8, 0x5957, 0x7279,
01937 0x85e4, 0x817e, 0x75bc, 0x8a8a, 0x68af, 0x5254, 0x8e22, 0x9511,
01938 0x63d0, 0x9898, 0x8e44, 0x557c, 0x4f53, 0x66ff, 0x568f, 0x60d5,
01939 0x6d95, 0x5243, 0x5c49, 0x5929, 0x6dfb, 0x586b, 0x7530, 0x751c,
01940 0x606c, 0x8214, 0x8146, 0x6311, 0x6761, 0x8fe2, 0x773a, 0x8df3,
01941 0x8d34, 0x94c1, 0x5e16, 0x5385, 0x542c, 0x70c3,
01942 /* 0xcd */
01943 0x866d, 0x866f, 0x8670, 0x8672, 0x8673, 0x8674, 0x8675, 0x8676,
01944 0x8677, 0x8678, 0x8683, 0x8684, 0x8685, 0x8686, 0x8687, 0x8688,
01945 0x8689, 0x868e, 0x868f, 0x8690, 0x8691, 0x8692, 0x8694, 0x8696,
01946 0x8697, 0x8698, 0x8699, 0x869a, 0x869b, 0x869e, 0x869f, 0x86a0,
01947 0x86a1, 0x86a2, 0x86a5, 0x86a6, 0x86ab, 0x86ad, 0x86ae, 0x86b2,
01948 0x86b3, 0x86b7, 0x86b8, 0x86b9, 0x86bb, 0x86bc, 0x86bd, 0x86be,
01949 0x86bf, 0x86c1, 0x86c2, 0x86c3, 0x86c5, 0x86c8, 0x86cc, 0x86cd,
01950 0x86d2, 0x86d3, 0x86d5, 0x86d6, 0x86d7, 0x86da, 0x86dc, 0x86dd,
01951 0x86e0, 0x86e1, 0x86e2, 0x86e3, 0x86e5, 0x86e6, 0x86e7, 0x86e8,
01952 0x86ea, 0x86eb, 0x86ec, 0x86ef, 0x86f5, 0x86f6, 0x86f7, 0x86fa,
01953 0x86fb, 0x86fc, 0x86fd, 0x86ff, 0x8701, 0x8704, 0x8705, 0x8706,
01954 0x870b, 0x870c, 0x870e, 0x870f, 0x8710, 0x8711, 0x8714, 0x8716,
01955 0x6c40, 0x5eff7, 0x505c, 0x4ead, 0x5ead, 0x633a, 0x8247, 0x901a,
01956 0x6850, 0x916e, 0x77b3, 0x540c, 0x94dc, 0x5f64, 0x7ae5, 0x6876,
01957 0x6345, 0x7b52, 0x7edf, 0x75db, 0x5077, 0x6295, 0x5934, 0x900f,
01958 0x51f8, 0x79c3, 0x7a81, 0x56fe, 0x5f92, 0x9014, 0x6d82, 0x5c60,
01959 0x571f, 0x5410, 0x5154, 0x6e4d, 0x56e2, 0x63a8, 0x9893, 0x817f,
01960 0x8715, 0x892a, 0x9000, 0x541e, 0x5c6f, 0x81c0, 0x62d6, 0x6258,
01961 0x8131, 0x9e35, 0x9640, 0x9a6e, 0x9a7c, 0x692d, 0x59a5, 0x62d3,
01962 0x553e, 0x6316, 0x54c7, 0x86d9, 0x6d3c, 0x5a03, 0x74e6, 0x889c,
01963 0x6b6a, 0x5916, 0x8c4c, 0x5f2f, 0x6e7e, 0x73a9, 0x987d, 0x4e38,
01964 0x70f7, 0x5b8c, 0x7897, 0x633d, 0x665a, 0x7696, 0x60cb, 0x5b9b,
01965 0x5a49, 0x4e07, 0x8155, 0x6c6a, 0x738b, 0x4ea1, 0x6789, 0x7f51,
01966 0x5f80, 0x65fa, 0x671b, 0x5fd8, 0x5984, 0x5a01,
01967 /* 0xce */
01968 0x8719, 0x871b, 0x871d, 0x871f, 0x8720, 0x8724, 0x8726, 0x8727,
01969 0x8728, 0x872a, 0x872b, 0x872c, 0x872d, 0x872f, 0x8730, 0x8732,
01970 0x8733, 0x8735, 0x8736, 0x8738, 0x8739, 0x873a, 0x873c, 0x873d,
01971 0x8740, 0x8741, 0x8742, 0x8743, 0x8744, 0x8745, 0x8746, 0x874a,
01972 0x874b, 0x874d, 0x874f, 0x8750, 0x8751, 0x8752, 0x8754, 0x8755,
01973 0x8756, 0x8758, 0x875a, 0x875b, 0x875c, 0x875d, 0x875e, 0x875f,
01974 0x8761, 0x8762, 0x8766, 0x8767, 0x8768, 0x8769, 0x876a, 0x876b,
01975 0x876c, 0x876d, 0x876f, 0x8771, 0x8772, 0x8773, 0x8775, 0x8777,
01976 0x8778, 0x8779, 0x877a, 0x877f, 0x8780, 0x8781, 0x8784, 0x8786,
01977 0x8787, 0x8789, 0x878a, 0x878c, 0x878e, 0x878f, 0x8790, 0x8791,
01978 0x8792, 0x8794, 0x8795, 0x8796, 0x8798, 0x8799, 0x879a, 0x879b,
01979 0x879c, 0x879d, 0x879e, 0x87a0, 0x87a1, 0x87a2, 0x87a3, 0x87a4,
01980 0x5dcd, 0x5fae, 0x5371, 0x97e6, 0x8fdd, 0x6845, 0x56f4, 0x552f,
01981 0x60df, 0x4e3a, 0x6f4d, 0x7ef4, 0x82c7, 0x840e, 0x59d4, 0x4f1f,
01982 0x4f2a, 0x5c3e, 0x7eac, 0x672a, 0x851a, 0x5473, 0x754f, 0x80c3,
01983 0x5582, 0x9b4f, 0x4f4d, 0x6e2d, 0x8c13, 0x5c09, 0x6170, 0x536b,
01984 0x761f, 0x6e29, 0x868a, 0x6587, 0x95fb, 0x7eb9, 0x543b, 0x7a33,
01985 0x7d0a, 0x95ee, 0x55e1, 0x7fc1, 0x74ee, 0x631d, 0x8717, 0x6da1,
01986 0x7a9d, 0x6211, 0x65a1, 0x5367, 0x63e1, 0x6c83, 0x5deb, 0x545c,
01987 0x94a8, 0x4e4c, 0x6c61, 0x8bec, 0x5c4b, 0x65e0, 0x829c, 0x68a7,
01988 0x543e, 0x5434, 0x6bcb, 0x6b66, 0x4e94, 0x6342, 0x5348, 0x821e,
01989 0x4f0d, 0x4fae, 0x575e, 0x620a, 0x96fe, 0x6664, 0x7269, 0x52ff,
01990 0x52a1, 0x609f, 0x8bef, 0x6614, 0x7199, 0x6790, 0x897f, 0x7852,
01991 0x77fd, 0x6670, 0x563b, 0x5438, 0x9521, 0x727a,
01992 /* 0xcf */
01993 0x87a5, 0x87a6, 0x87a7, 0x87a9, 0x87aa, 0x87ae, 0x87b0, 0x87b1,
01994 0x87b2, 0x87b4, 0x87b6, 0x87b7, 0x87b8, 0x87b9, 0x87bb, 0x87bc,
01995 0x87be, 0x87bf, 0x87c1, 0x87c2, 0x87c3, 0x87c4, 0x87c5, 0x87c7,
01996 0x87c8, 0x87c9, 0x87cc, 0x87cd, 0x87ce, 0x87cf, 0x87d0, 0x87d4,
01997 0x87d5, 0x87d6, 0x87d7, 0x87d8, 0x87d9, 0x87da, 0x87dc, 0x87dd,
01998 0x87de, 0x87df, 0x87e1, 0x87e2, 0x87e3, 0x87e4, 0x87e6, 0x87e7,
01999 0x87e8, 0x87e9, 0x87eb, 0x87ec, 0x87ed, 0x87ef, 0x87f0, 0x87f1,
02000 0x87f2, 0x87f3, 0x87f4, 0x87f5, 0x87f6, 0x87f7, 0x87f8, 0x87fa,
02001 0x87fb, 0x87fc, 0x87fd, 0x87ff, 0x8800, 0x8801, 0x8802, 0x8804,
02002 0x8805, 0x8806, 0x8807, 0x8808, 0x8809, 0x880b, 0x880c, 0x880d,
02003 0x880e, 0x880f, 0x8810, 0x8811, 0x8812, 0x8814, 0x8817, 0x8818,
02004 0x8819, 0x881a, 0x881c, 0x881d, 0x881e, 0x881f, 0x8820, 0x8823,
02005 0x7a00, 0x606f, 0x5e0c, 0x6089, 0x819d, 0x5915, 0x60dc, 0x7184,
02006 0x70ef, 0x6eaa, 0x6c50, 0x7280, 0x6a84, 0x88ad, 0x5e2d, 0x4e60,

```



```
02007 0x5ab3, 0x559c, 0x94e3, 0x6d17, 0x7cfb, 0x9699, 0x620f, 0x7ec6,
02008 0x778e, 0x867e, 0x5323, 0x971e, 0x8f96, 0x6687, 0x5ce1, 0x4fa0,
02009 0x72ed, 0x4e0b, 0x53a6, 0x590f, 0x5413, 0x6380, 0x9528, 0x5148,
02010 0x4ed9, 0x9c9c, 0x7ea4, 0x54b8, 0x8d24, 0x8854, 0x8237, 0x95f2,
02011 0x6d8e, 0x5f26, 0x5acc, 0x663e, 0x9669, 0x73b0, 0x732e, 0x53bf,
02012 0x817a, 0x9985, 0x7fa1, 0x5baa, 0x9677, 0x9650, 0x7ebf, 0x76f8,
02013 0x53a2, 0x9576, 0x9999, 0x7bb1, 0x8944, 0x6e58, 0x4e61, 0x7fd4,
02014 0x7965, 0x8be6, 0x60f3, 0x54cd, 0x4eab, 0x9879, 0x5df7, 0x6a61,
02015 0x50cf, 0x5411, 0x8c61, 0x8427, 0x785d, 0x9704, 0x524a, 0x54ee,
02016 0x56a3, 0x9500, 0x6d88, 0x5bb5, 0x6dc6, 0x6653,
02017 /* 0xd0 */
02018 0x8824, 0x8825, 0x8826, 0x8827, 0x8828, 0x8829, 0x882a, 0x882b,
02019 0x882c, 0x882d, 0x882e, 0x882f, 0x8830, 0x8831, 0x8833, 0x8834,
02020 0x8835, 0x8836, 0x8837, 0x8838, 0x883a, 0x883b, 0x883d, 0x883e,
02021 0x883f, 0x8841, 0x8842, 0x8843, 0x8846, 0x8847, 0x8848, 0x8849,
02022 0x884a, 0x884b, 0x884e, 0x884f, 0x8850, 0x8851, 0x8852, 0x8853,
02023 0x8855, 0x8856, 0x8858, 0x885a, 0x885b, 0x885c, 0x885d, 0x885e,
02024 0x885f, 0x8860, 0x8866, 0x8867, 0x886a, 0x886d, 0x886f, 0x8871,
02025 0x8873, 0x8874, 0x8875, 0x8876, 0x8878, 0x8879, 0x887a, 0x887b,
02026 0x887c, 0x8880, 0x8883, 0x8886, 0x8887, 0x8889, 0x888a, 0x888c,
02027 0x888e, 0x888f, 0x8890, 0x8891, 0x8893, 0x8894, 0x8895, 0x8897,
02028 0x8898, 0x8899, 0x889a, 0x889b, 0x889d, 0x889e, 0x889f, 0x88a0,
02029 0x88a1, 0x88a3, 0x88a5, 0x88a6, 0x88a7, 0x88a8, 0x88a9, 0x88aa,
02030 0x5c0f, 0x5b5d, 0x6821, 0x8096, 0x5578, 0x7b11, 0x6548, 0x6954,
02031 0x4e9b, 0x6b47, 0x874e, 0x978b, 0x534f, 0x631f, 0x643a, 0x90aa,
02032 0x659c, 0x80c1, 0x8c10, 0x5199, 0x68b0, 0x5378, 0x87f9, 0x61c8,
02033 0x6cc4, 0x6cfb, 0x8c22, 0x5c51, 0x85aa, 0x82af, 0x950c, 0x6b23,
02034 0x8f9b, 0x65b0, 0x5ffb, 0x5fc3, 0x4fe1, 0x8845, 0x661f, 0x8165,
02035 0x7329, 0x60fa, 0x5174, 0x5211, 0x578b, 0x5f62, 0x90a2, 0x884c,
02036 0x9192, 0x5e78, 0x674f, 0x6027, 0x59d3, 0x5144, 0x51f6, 0x80f8,
02037 0x5308, 0x6c79, 0x96c4, 0x718a, 0x4f11, 0x4fee, 0x7f9e, 0x673d,
02038 0x55c5, 0x9508, 0x79c0, 0x8896, 0x7ee3, 0x589f, 0x620c, 0x9700,
02039 0x865a, 0x5618, 0x987b, 0x5f90, 0x8bb8, 0x84c4, 0x9157, 0x53d9,
02040 0x65ed, 0x5e8f, 0x755c, 0x6064, 0x7d6e, 0x5a7f, 0x7eea, 0x7eed,
02041 0x8f69, 0x55a7, 0x5ba3, 0x60ac, 0x65cb, 0x7384,
02042 /* 0xd1 */
02043 0x88ac, 0x88ae, 0x88af, 0x88b0, 0x88b2, 0x88b3, 0x88b4, 0x88b5,
02044 0x88b6, 0x88b8, 0x88b9, 0x88ba, 0x88bb, 0x88bd, 0x88be, 0x88bf,
02045 0x88c0, 0x88c3, 0x88c4, 0x88c7, 0x88c8, 0x88ca, 0x88cb, 0x88cc,
02046 0x88cd, 0x88cf, 0x88d0, 0x88d1, 0x88d3, 0x88d6, 0x88d7, 0x88da,
02047 0x88db, 0x88dc, 0x88dd, 0x88de, 0x88e0, 0x88e1, 0x88e6, 0x88e7,
02048 0x88e9, 0x88ea, 0x88eb, 0x88ec, 0x88ed, 0x88ee, 0x88ef, 0x88f2,
02049 0x88f5, 0x88f6, 0x88f7, 0x88fa, 0x88fb, 0x88fd, 0x88ff, 0x8900,
02050 0x8901, 0x8903, 0x8904, 0x8905, 0x8906, 0x8907, 0x8908, 0x8909,
02051 0x890b, 0x890c, 0x890d, 0x890e, 0x890f, 0x8911, 0x8914, 0x8915,
02052 0x8916, 0x8917, 0x8918, 0x891c, 0x891d, 0x891e, 0x891f, 0x8920,
02053 0x8922, 0x8923, 0x8924, 0x8926, 0x8927, 0x8928, 0x8929, 0x892c,
02054 0x892d, 0x892e, 0x892f, 0x8931, 0x8932, 0x8933, 0x8935, 0x8937,
02055 0x9009, 0x7663, 0x7729, 0x7eda, 0x9774, 0x859b, 0x5b66, 0x7a74,
02056 0x96ea, 0x8840, 0x52cb, 0x718f, 0x5faa, 0x65ec, 0x8be2, 0x5bfb,
02057 0x9a6f, 0x5de1, 0x6b89, 0x6c5b, 0x8bad, 0x8baf, 0x900a, 0x8fc5,
02058 0x538b, 0x62bc, 0x9e26, 0x9e2d, 0x5440, 0x4e2b, 0x82bd, 0x7259,
02059 0x869c, 0x5d16, 0x8859, 0x6daf, 0x96c5, 0x54d1, 0x4e9a, 0x8bb6,
02060 0x7109, 0x54bd, 0x9609, 0x70df, 0x6df9, 0x76d0, 0x4e25, 0x7814,
02061 0x8712, 0x5ca9, 0x5ef6, 0x8a00, 0x989c, 0x960e, 0x708e, 0x6cbf,
02062 0x5944, 0x63a9, 0x773c, 0x884d, 0x6f14, 0x8273, 0x5830, 0x71d5,
02063 0x538c, 0x781a, 0x96c1, 0x5501, 0x5f66, 0x7130, 0x5bb4, 0x8c1a,
02064 0x9a8c, 0x6b83, 0x592e, 0x9e2f, 0x79e7, 0x6768, 0x626c, 0x4f6f,
02065 0x75a1, 0x7f8a, 0x6d0b, 0x9633, 0x6c27, 0x4ef0, 0x75d2, 0x517b,
02066 0x6837, 0x6f3e, 0x9080, 0x8170, 0x5996, 0x7476,
02067 /* 0xd2 */
02068 0x8938, 0x8939, 0x893a, 0x893b, 0x893c, 0x893d, 0x893e, 0x893f,
02069 0x8940, 0x8942, 0x8943, 0x8945, 0x8946, 0x8947, 0x8948, 0x8949,
02070 0x894a, 0x894b, 0x894c, 0x894d, 0x894e, 0x894f, 0x8950, 0x8951,
02071 0x8952, 0x8953, 0x8954, 0x8955, 0x8956, 0x8957, 0x8958, 0x8959,
02072 0x895a, 0x895b, 0x895c, 0x895d, 0x8960, 0x8961, 0x8962, 0x8963,
02073 0x8964, 0x8965, 0x8967, 0x8968, 0x8969, 0x896a, 0x896b, 0x896c,
02074 0x896d, 0x896e, 0x896f, 0x8970, 0x8971, 0x8972, 0x8973, 0x8974,
02075 0x8975, 0x8976, 0x8977, 0x8978, 0x8979, 0x897a, 0x897c, 0x897d,
02076 0x897e, 0x8980, 0x8982, 0x8984, 0x8985, 0x8987, 0x8988, 0x8989,
02077 0x898a, 0x898b, 0x898c, 0x898d, 0x898e, 0x898f, 0x8990, 0x8991,
02078 0x8992, 0x8993, 0x8994, 0x8995, 0x8996, 0x8997, 0x8998, 0x8999,
02079 0x899a, 0x899b, 0x899c, 0x899d, 0x899e, 0x899f, 0x89a0, 0x89a1,
02080 0x6447, 0x5c27, 0x9065, 0x7a91, 0x8c23, 0x59da, 0x54ac, 0x8200,
02081 0x836f, 0x8981, 0x8000, 0x6930, 0x564e, 0x8036, 0x7237, 0x91ce,
02082 0x51b6, 0x4e5f, 0x9875, 0x6396, 0x4e1a, 0x53f6, 0x66f3, 0x814b,
02083 0x591c, 0x6db2, 0x4e00, 0x58f9, 0x533b, 0x63d6, 0x94f1, 0x4f9d,
02084 0x4f0a, 0x8863, 0x9890, 0x5937, 0x9057, 0x79fb, 0x4eea, 0x80f0,
02085 0x7591, 0x6c82, 0x5b9c, 0x59e8, 0x5f5d, 0x6905, 0x8681, 0x501a,
02086 0x5df2, 0x4e59, 0x77e3, 0x4ee5, 0x827a, 0x6291, 0x6613, 0x9091,
02087 0x5c79, 0x4ebf, 0x5f79, 0x81c6, 0x9038, 0x8084, 0x75ab, 0x4ea6,
02088 0x88d4, 0x610f, 0x6bc5, 0x5fc6, 0x4e49, 0x76ca, 0x6ea2, 0x8be3,
02089 0x8bae, 0x8c0a, 0x8bd1, 0x5f02, 0x7ffc, 0x7fcc, 0x7ece, 0x8335,
02090 0x836b, 0x56e0, 0x6bb7, 0x97f3, 0x9634, 0x59fb, 0x541f, 0x94f6,
02091 0x6deb, 0x5bc5, 0x996e, 0x5c39, 0x5f15, 0x9690,
02092 /* 0xd3 */
02093 0x89a2, 0x89a3, 0x89a4, 0x89a5, 0x89a6, 0x89a7, 0x89a8, 0x89a9,
```

```

02094 0x89aa, 0x89ab, 0x89ac, 0x89ad, 0x89ae, 0x89af, 0x89b0, 0x89b1,
02095 0x89b2, 0x89b3, 0x89b4, 0x89b5, 0x89b6, 0x89b7, 0x89b8, 0x89b9,
02096 0x89ba, 0x89bb, 0x89bc, 0x89bd, 0x89be, 0x89bf, 0x89c0, 0x89c3,
02097 0x89cd, 0x89d3, 0x89d4, 0x89d5, 0x89d7, 0x89d8, 0x89d9, 0x89db,
02098 0x89dd, 0x89df, 0x89e0, 0x89e1, 0x89e2, 0x89e4, 0x89e7, 0x89e8,
02099 0x89e9, 0x89ea, 0x89ec, 0x89ed, 0x89ee, 0x89f0, 0x89f1, 0x89f2,
02100 0x89f4, 0x89f5, 0x89f6, 0x89f7, 0x89f8, 0x89f9, 0x89fa, 0x89fb,
02101 0x89fc, 0x89fd, 0x89fe, 0x89ff, 0x8a01, 0x8a02, 0x8a03, 0x8a04,
02102 0x8a05, 0x8a06, 0x8a08, 0x8a09, 0x8a0a, 0x8a0b, 0x8a0c, 0x8a0d,
02103 0x8a0e, 0x8a0f, 0x8a10, 0x8a11, 0x8a12, 0x8a13, 0x8a14, 0x8a15,
02104 0x8a16, 0x8a17, 0x8a18, 0x8a19, 0x8a1a, 0x8a1b, 0x8a1c, 0x8a1d,
02105 0x5370, 0x82f1, 0x6a31, 0x5a74, 0x9e70, 0x5e94, 0x7f28, 0x83b9,
02106 0x8424, 0x8425, 0x8367, 0x8747, 0x8fce, 0x8d62, 0x76c8, 0x5f71,
02107 0x9896, 0x786c, 0x6620, 0x54df, 0x62e5, 0x4f63, 0x81c3, 0x75c8,
02108 0x5eb8, 0x96cd, 0x8e0a, 0x86f9, 0x548f, 0x6cf3, 0x6d8c, 0x6c38,
02109 0x607f, 0x52c7, 0x7528, 0x5e7d, 0x4f18, 0x60a0, 0x5fe7, 0x5c24,
02110 0x7531, 0x90ae, 0x94c0, 0x72b9, 0x6cb9, 0x6e38, 0x9149, 0x6709,
02111 0x53cb, 0x53f3, 0x4f51, 0x91c9, 0x8bf1, 0x53c8, 0x5e7c, 0x8fc2,
02112 0x6de4, 0x4e8e, 0x76c2, 0x6986, 0x865e, 0x611a, 0x8206, 0x4f59,
02113 0x4fde, 0x903e, 0x9c7c, 0x6109, 0x6e1d, 0x6e14, 0x9685, 0x4e88,
02114 0x5a31, 0x96e8, 0x4e0e, 0x5c7f, 0x79b9, 0x5b87, 0x8bed, 0x7fbd,
02115 0x7389, 0x57df, 0x828b, 0x90c1, 0x5401, 0x9047, 0x55bb, 0x5cea,
02116 0x5fa1, 0x6108, 0x6b32, 0x72f1, 0x80b2, 0x8a89,
02117 /* 0xd4 */
02118 0x8a1e, 0x8a1f, 0x8a20, 0x8a21, 0x8a22, 0x8a23, 0x8a24, 0x8a25,
02119 0x8a26, 0x8a27, 0x8a28, 0x8a29, 0x8a2a, 0x8a2b, 0x8a2c, 0x8a2d,
02120 0x8a2e, 0x8a2f, 0x8a30, 0x8a31, 0x8a32, 0x8a33, 0x8a34, 0x8a35,
02121 0x8a36, 0x8a37, 0x8a38, 0x8a39, 0x8a3a, 0x8a3b, 0x8a3c, 0x8a3d,
02122 0x8a3f, 0x8a40, 0x8a41, 0x8a42, 0x8a43, 0x8a44, 0x8a45, 0x8a46,
02123 0x8a47, 0x8a49, 0x8a4a, 0x8a4b, 0x8a4c, 0x8a4d, 0x8a4e, 0x8a4f,
02124 0x8a50, 0x8a51, 0x8a52, 0x8a53, 0x8a54, 0x8a55, 0x8a56, 0x8a57,
02125 0x8a58, 0x8a59, 0x8a5a, 0x8a5b, 0x8a5c, 0x8a5d, 0x8a5e, 0x8a5f,
02126 0x8a60, 0x8a61, 0x8a62, 0x8a63, 0x8a64, 0x8a65, 0x8a66, 0x8a67,
02127 0x8a68, 0x8a69, 0x8a6a, 0x8a6b, 0x8a6c, 0x8a6d, 0x8a6e, 0x8a6f,
02128 0x8a70, 0x8a71, 0x8a72, 0x8a73, 0x8a74, 0x8a75, 0x8a76, 0x8a77,
02129 0x8a78, 0x8a7a, 0x8a7b, 0x8a7c, 0x8a7d, 0x8a7e, 0x8a7f, 0x8a80,
02130 0x6d74, 0x5bd3, 0x88d5, 0x9884, 0x8c6b, 0x9a6d, 0x9e33, 0x6e0a,
02131 0x51a4, 0x5143, 0x57a3, 0x8881, 0x539f, 0x63f4, 0x8f95, 0x56ed,
02132 0x5458, 0x5706, 0x733f, 0x6e90, 0x7f18, 0x8fdc, 0x82d1, 0x613f,
02133 0x6028, 0x9662, 0x66f0, 0x7ea6, 0x8d8a, 0x8dc3, 0x94a5, 0x5cb3,
02134 0x7ca4, 0x6708, 0x60a6, 0x9605, 0x8018, 0x4e91, 0x90e7, 0x5300,
02135 0x9668, 0x5141, 0x8fd0, 0x8574, 0x915d, 0x6655, 0x97f5, 0x5b55,
02136 0x531d, 0x7838, 0x6742, 0x683d, 0x54c9, 0x707e, 0x5bb0, 0x8f7d,
02137 0x518d, 0x5728, 0x54b1, 0x6512, 0x6682, 0x8d5e, 0x8d43, 0x810f,
02138 0x846c, 0x906d, 0x7cdf, 0x51ff, 0x85fb, 0x67a3, 0x65e9, 0x6fa1,
02139 0x86a4, 0x8e81, 0x566a, 0x9020, 0x7682, 0x7076, 0x71e5, 0x8d23,
02140 0x62e9, 0x5219, 0x6cfd, 0x8d3c, 0x600e, 0x589e, 0x618e, 0x66fe,
02141 0x8d60, 0x624e, 0x55b3, 0x6e23, 0x672d, 0x8f67,
02142 /* 0xd5 */
02143 0x8a81, 0x8a82, 0x8a83, 0x8a84, 0x8a85, 0x8a86, 0x8a87, 0x8a88,
02144 0x8a8b, 0x8a8c, 0x8a8d, 0x8a8e, 0x8a8f, 0x8a90, 0x8a91, 0x8a92,
02145 0x8a94, 0x8a95, 0x8a96, 0x8a97, 0x8a98, 0x8a99, 0x8a9a, 0x8a9b,
02146 0x8a9c, 0x8a9d, 0x8a9e, 0x8a9f, 0x8aa0, 0x8aa1, 0x8aa2, 0x8aa3,
02147 0x8aa4, 0x8aa5, 0x8aa6, 0x8aa7, 0x8aa8, 0x8aa9, 0x8aaa, 0x8aab,
02148 0x8aac, 0x8aad, 0x8aae, 0x8aaf, 0x8ab0, 0x8ab1, 0x8ab2, 0x8ab3,
02149 0x8ab4, 0x8ab5, 0x8ab6, 0x8ab7, 0x8ab8, 0x8ab9, 0x8aba, 0x8abb,
02150 0x8abc, 0x8abd, 0x8abe, 0x8abf, 0x8ac0, 0x8ac1, 0x8ac2, 0x8ac3,
02151 0x8ac4, 0x8ac5, 0x8ac6, 0x8ac7, 0x8ac8, 0x8ac9, 0x8aca, 0x8acb,
02152 0x8acc, 0x8acd, 0x8ace, 0x8acf, 0x8ad0, 0x8ad1, 0x8ad2, 0x8ad3,
02153 0x8ad4, 0x8ad5, 0x8ad6, 0x8ad7, 0x8ad8, 0x8ad9, 0x8ada, 0x8adb,
02154 0x8adc, 0x8add, 0x8ade, 0x8adf, 0x8ae0, 0x8ae1, 0x8ae2, 0x8ae3,
02155 0x94e1, 0x95f8, 0x7728, 0x6805, 0x69a8, 0x548b, 0x4e4d, 0x70b8,
02156 0x8bc8, 0x6458, 0x658b, 0x5b85, 0x7a84, 0x503a, 0x5be8, 0x77bb,
02157 0x6be1, 0x8a79, 0x7c98, 0x6cbe, 0x76cf, 0x65a9, 0x8f97, 0x5d2d,
02158 0x5c55, 0x8638, 0x6808, 0x5360, 0x6218, 0x7ad9, 0x6e5b, 0x7efd,
02159 0x6a1f, 0x7ae0, 0x5f70, 0x6f33, 0x5f20, 0x638c, 0x6da8, 0x6756,
02160 0x4e08, 0x5e10, 0x8d26, 0x4ed7, 0x80c0, 0x7634, 0x969c, 0x62db,
02161 0x662d, 0x627e, 0x6cbc, 0x8d75, 0x7167, 0x7f69, 0x5146, 0x8087,
02162 0x53ec, 0x906e, 0x6298, 0x54f2, 0x86f0, 0x8f99, 0x8005, 0x9517,
02163 0x8517, 0x8fd9, 0x6d59, 0x73cd, 0x659f, 0x771f, 0x7504, 0x7827,
02164 0x81fb, 0x8d1e, 0x9488, 0x4fa6, 0x6795, 0x75b9, 0x8bca, 0x9707,
02165 0x632f, 0x9547, 0x9635, 0x84b8, 0x6323, 0x7741, 0x5f81, 0x72f0,
02166 0x4e89, 0x6014, 0x6574, 0x62ef, 0x6b63, 0x653f,
02167 /* 0xd6 */
02168 0x8ae4, 0x8ae5, 0x8ae6, 0x8ae7, 0x8ae8, 0x8ae9, 0x8aea, 0x8aeb,
02169 0x8aec, 0x8aed, 0x8aee, 0x8aef, 0x8af0, 0x8af1, 0x8af2, 0x8af3,
02170 0x8af4, 0x8af5, 0x8af6, 0x8af7, 0x8af8, 0x8af9, 0x8afa, 0x8afb,
02171 0x8afc, 0x8afd, 0x8afe, 0x8aff, 0x8b00, 0x8b01, 0x8b02, 0x8b03,
02172 0x8b04, 0x8b05, 0x8b06, 0x8b08, 0x8b09, 0x8b0a, 0x8b0b, 0x8b0c,
02173 0x8b0d, 0x8b0e, 0x8b0f, 0x8b10, 0x8b11, 0x8b12, 0x8b13, 0x8b14,
02174 0x8b15, 0x8b16, 0x8b17, 0x8b18, 0x8b19, 0x8b1a, 0x8b1b, 0x8b1c,
02175 0x8b1d, 0x8b1e, 0x8b1f, 0x8b20, 0x8b21, 0x8b22, 0x8b23, 0x8b24,
02176 0x8b25, 0x8b27, 0x8b28, 0x8b29, 0x8b2a, 0x8b2b, 0x8b2c, 0x8b2d,
02177 0x8b2e, 0x8b2f, 0x8b30, 0x8b31, 0x8b32, 0x8b33, 0x8b34, 0x8b35,
02178 0x8b36, 0x8b37, 0x8b38, 0x8b39, 0x8b3a, 0x8b3b, 0x8b3c, 0x8b3d,
02179 0x8b3e, 0x8b3f, 0x8b40, 0x8b41, 0x8b42, 0x8b43, 0x8b44, 0x8b45,
02180 0x5e27, 0x75c7, 0x90d1, 0x8bbc1, 0x829d, 0x679d, 0x652f, 0x5431,

```



```
02181 0x8718, 0x77e5, 0x80a2, 0x8102, 0x6c41, 0x4e4b, 0x7ec7, 0x804c,
02182 0x76f4, 0x690a, 0x6b96, 0x6267, 0x503c, 0x4f84, 0x5740, 0x6307,
02183 0x6b62, 0x8dbb, 0x53ea, 0x65e8, 0x7eb8, 0x5fd7, 0x631a, 0x63b7,
02184 0x81f3, 0x81f4, 0x7f6e, 0x5e1c, 0x5cd9, 0x5236, 0x667a, 0x79e9,
02185 0x7a1a, 0x8d28, 0x7099, 0x75d4, 0x6ede, 0x6cbb, 0x7a92, 0x4e2d,
02186 0x76c5, 0x5fe0, 0x949f, 0x8877, 0x7ec8, 0x79cd, 0x80bf, 0x91cd,
02187 0x4ef2, 0x4f17, 0x821f, 0x5468, 0x5dde, 0x6d32, 0x8bcc, 0x7ca5,
02188 0x8f74, 0x8098, 0x5e1a, 0x5492, 0x76b1, 0x5b99, 0x663c, 0x9aa4,
02189 0x73e0, 0x682a, 0x86db, 0x6731, 0x732a, 0x8bf8, 0x8bdb, 0x9010,
02190 0x7af9, 0x70db, 0x716e, 0x62c4, 0x77a9, 0x5631, 0x4e3b, 0x8457,
02191 0x67f1, 0x52a9, 0x86c0, 0x8d2e, 0x94f8, 0x7b51,
02192 /* 0xd7 */
02193 0x8b46, 0x8b47, 0x8b48, 0x8b49, 0x8b4a, 0x8b4b, 0x8b4c, 0x8b4d,
02194 0x8b4e, 0x8b4f, 0x8b50, 0x8b51, 0x8b52, 0x8b53, 0x8b54, 0x8b55,
02195 0x8b56, 0x8b57, 0x8b58, 0x8b59, 0x8b5a, 0x8b5b, 0x8b5c, 0x8b5d,
02196 0x8b5e, 0x8b5f, 0x8b60, 0x8b61, 0x8b62, 0x8b63, 0x8b64, 0x8b65,
02197 0x8b67, 0x8b68, 0x8b69, 0x8b6a, 0x8b6b, 0x8b6d, 0x8b6e, 0x8b6f,
02198 0x8b70, 0x8b71, 0x8b72, 0x8b73, 0x8b74, 0x8b75, 0x8b76, 0x8b77,
02199 0x8b78, 0x8b79, 0x8b7a, 0x8b7b, 0x8b7c, 0x8b7d, 0x8b7e, 0x8b7f,
02200 0x8b80, 0x8b81, 0x8b82, 0x8b83, 0x8b84, 0x8b85, 0x8b86, 0x8b87,
02201 0x8b88, 0x8b89, 0x8b8a, 0x8b8b, 0x8b8c, 0x8b8d, 0x8b8e, 0x8b8f,
02202 0x8b90, 0x8b91, 0x8b92, 0x8b93, 0x8b94, 0x8b95, 0x8b96, 0x8b97,
02203 0x8b98, 0x8b99, 0x8b9a, 0x8b9b, 0x8b9c, 0x8b9d, 0x8b9e, 0x8b9f,
02204 0x8bac, 0x8bb1, 0x8bbb, 0x8bc7, 0x8bd0, 0x8bea, 0x8c09, 0x8c1e,
02205 0x4f4f, 0x6ce8, 0x795d, 0x9a7b, 0x6293, 0x722a, 0x62fd, 0x4e13,
02206 0x7816, 0x8f6c, 0x64b0, 0x8d5a, 0x7bc6, 0x6869, 0x5e84, 0x88c5,
02207 0x5986, 0x649e, 0x58ee, 0x72b6, 0x690e, 0x9525, 0x8ffd, 0x8d58,
02208 0x5760, 0x7f00, 0x8c06, 0x51c6, 0x6349, 0x62d9, 0x5353, 0x684c,
02209 0x7422, 0x8301, 0x914c, 0x5544, 0x7740, 0x707c, 0x6d4a, 0x5179,
02210 0x54a8, 0x8d44, 0x59ff, 0x6ecb, 0x6dc4, 0x5b5c, 0x7d2b, 0x4ed4,
02211 0x7c7d, 0x6ed3, 0x5b50, 0x81ea, 0x6e0d, 0x5b57, 0x9b03, 0x68d5,
02212 0x8e2a, 0x5b97, 0x7efc, 0x603b, 0x7eb5, 0x90b9, 0x8d70, 0x594f,
02213 0x63cd, 0x79df, 0x8db3, 0x5352, 0x65cf, 0x7956, 0x8bc5, 0x963b,
02214 0x7ec4, 0x94bb, 0x7e82, 0x5634, 0x9189, 0x6700, 0x7f6a, 0x5c0a,
02215 0x9075, 0x6628, 0x5de6, 0x4f50, 0x67de, 0x505a, 0x4f5c, 0x5750,
02216 0x5ea7, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
02217 /* 0xd8 */
02218 0x8c38, 0x8c39, 0x8c3a, 0x8c3b, 0x8c3c, 0x8c3d, 0x8c3e, 0x8c3f,
02219 0x8c40, 0x8c42, 0x8c43, 0x8c44, 0x8c45, 0x8c48, 0x8c4a, 0x8c4b,
02220 0x8c4d, 0x8c4e, 0x8c4f, 0x8c50, 0x8c51, 0x8c52, 0x8c53, 0x8c54,
02221 0x8c56, 0x8c57, 0x8c58, 0x8c59, 0x8c5b, 0x8c5c, 0x8c5d, 0x8c5e,
02222 0x8c5f, 0x8c60, 0x8c63, 0x8c64, 0x8c65, 0x8c66, 0x8c67, 0x8c68,
02223 0x8c69, 0x8c6c, 0x8c6d, 0x8c6e, 0x8c6f, 0x8c70, 0x8c71, 0x8c72,
02224 0x8c74, 0x8c75, 0x8c76, 0x8c77, 0x8c7b, 0x8c7c, 0x8c7d, 0x8c7e,
02225 0x8c7f, 0x8c80, 0x8c81, 0x8c83, 0x8c84, 0x8c86, 0x8c87, 0x8c88,
02226 0x8c8b, 0x8c8d, 0x8c8e, 0x8c8f, 0x8c90, 0x8c91, 0x8c92, 0x8c93,
02227 0x8c95, 0x8c96, 0x8c97, 0x8c99, 0x8c9a, 0x8c9b, 0x8c9c, 0x8c9d,
02228 0x8c9e, 0x8c9f, 0x8ca0, 0x8ca1, 0x8ca2, 0x8ca3, 0x8ca4, 0x8ca5,
02229 0x8ca6, 0x8ca7, 0x8ca8, 0x8ca9, 0x8caa, 0x8cab, 0x8cac, 0x8cad,
02230 0x4e8d, 0x4e0c, 0x5140, 0x4e10, 0x5eff, 0x5345, 0x4e15, 0x4e98,
02231 0x4e1e, 0x9b32, 0x5b6c, 0x5669, 0x4e28, 0x79ba, 0x4e3f, 0x5315,
02232 0x4e47, 0x592d, 0x723b, 0x536e, 0x6c10, 0x56df, 0x80e4, 0x9997,
02233 0x6bd3, 0x777e, 0x9f17, 0x4e36, 0x4e9f, 0x9f10, 0x4e5c, 0x4e69,
02234 0x4e93, 0x8288, 0x5b5b, 0x556c, 0x560f, 0x4ec4, 0x538d, 0x539d,
02235 0x53a3, 0x53a5, 0x53ae, 0x9765, 0x8d5d, 0x531a, 0x53f5, 0x5326,
02236 0x532e, 0x533e, 0x8d5c, 0x5366, 0x5363, 0x5202, 0x5208, 0x520e,
02237 0x522d, 0x5233, 0x523f, 0x5240, 0x524c, 0x525e, 0x5261, 0x525c,
02238 0x84af, 0x527d, 0x5282, 0x5281, 0x5290, 0x5293, 0x5182, 0x7f54,
02239 0x4ebb, 0x4ec3, 0x4ec9, 0x4ec2, 0x4ee8, 0x4ee1, 0x4eeb, 0x4ede,
02240 0x4f1b, 0x4ef3, 0x4f22, 0x4f64, 0x4ef5, 0x4f25, 0x4f27, 0x4f09,
02241 0x4f2b, 0x4f5e, 0x4f67, 0x6538, 0x4f5a, 0x4f5d,
02242 /* 0xd9 */
02243 0x8cae, 0x8caf, 0x8cb0, 0x8cb1, 0x8cb2, 0x8cb3, 0x8cb4, 0x8cb5,
02244 0x8cb6, 0x8cb7, 0x8cb8, 0x8cb9, 0x8cba, 0x8cbb, 0x8cbc, 0x8cbd,
02245 0x8cbe, 0x8cbf, 0x8cc0, 0x8cc1, 0x8cc2, 0x8cc3, 0x8cc4, 0x8cc5,
02246 0x8cc6, 0x8cc7, 0x8cc8, 0x8cc9, 0x8cca, 0x8ccb, 0x8ccc, 0x8ccd,
02247 0x8cce, 0x8ccf, 0x8cd0, 0x8cd1, 0x8cd2, 0x8cd3, 0x8cd4, 0x8cd5,
02248 0x8cd6, 0x8cd7, 0x8cd8, 0x8cd9, 0x8cda, 0x8cdb, 0x8cdc, 0x8cdd,
02249 0x8cde, 0x8cdf, 0x8ce0, 0x8ce1, 0x8ce2, 0x8ce3, 0x8ce4, 0x8ce5,
02250 0x8ce6, 0x8ce7, 0x8ce8, 0x8ce9, 0x8cea, 0x8ceb, 0x8cec, 0x8ced,
02251 0x8cee, 0x8cef, 0x8cf0, 0x8cf1, 0x8cf2, 0x8cf3, 0x8cf4, 0x8cf5,
02252 0x8cf6, 0x8cf7, 0x8cf8, 0x8cf9, 0x8cfa, 0x8cfb, 0x8cfc, 0x8cfd,
02253 0x8cfe, 0x8cff, 0x8d00, 0x8d01, 0x8d02, 0x8d03, 0x8d04, 0x8d05,
02254 0x8d06, 0x8d07, 0x8d08, 0x8d09, 0x8d0a, 0x8d0b, 0x8d0c, 0x8d0d,
02255 0x4f5f, 0x4f57, 0x4f32, 0x4f3d, 0x4f76, 0x4f74, 0x4f91, 0x4f89,
02256 0x4f83, 0x4f8f, 0x4f7e, 0x4f7b, 0x4faa, 0x4f7c, 0x4fac, 0x4f94,
02257 0x4fe6, 0x4fe8, 0x4fea, 0x4fc5, 0x4fda, 0x4fe3, 0x4fdc, 0x4fd1,
02258 0x4fdf, 0x4ff8, 0x5029, 0x504c, 0x4ff3, 0x502c, 0x500f, 0x502e,
02259 0x502d, 0x4ffe, 0x501c, 0x500c, 0x5025, 0x5028, 0x507e, 0x5043,
02260 0x5055, 0x5048, 0x504e, 0x504c, 0x507b, 0x50a5, 0x50a7, 0x50a9,
02261 0x50ba, 0x50d6, 0x5106, 0x50ed, 0x50ec, 0x50e6, 0x50ee, 0x5107,
02262 0x510b, 0x4edd, 0x6c3d, 0x4f58, 0x4f65, 0x4fce, 0x9fa0, 0x6c46,
02263 0x7c74, 0x516e, 0x5dfd, 0x9ec9, 0x9998, 0x5181, 0x5914, 0x52f9,
02264 0x530d, 0x8a07, 0x5310, 0x51eb, 0x5919, 0x5155, 0x4ea0, 0x5156,
02265 0x4eb3, 0x886e, 0x88a4, 0x4eb5, 0x8114, 0x88d2, 0x7980, 0x5b34,
02266 0x8803, 0x7fb8, 0x51ab, 0x51b1, 0x51bd, 0x51bc,
02267 /* 0xda */
```

```
02268 0x8d0e, 0x8d0f, 0x8d10, 0x8d11, 0x8d12, 0x8d13, 0x8d14, 0x8d15,
02269 0x8d16, 0x8d17, 0x8d18, 0x8d19, 0x8d1a, 0x8d1b, 0x8d1c, 0x8d20,
02270 0x8d51, 0x8d52, 0x8d57, 0x8d5f, 0x8d65, 0x8d68, 0x8d69, 0x8d6a,
02271 0x8d6c, 0x8d6e, 0x8d6f, 0x8d71, 0x8d72, 0x8d78, 0x8d79, 0x8d7a,
02272 0x8d7b, 0x8d7c, 0x8d7d, 0x8d7e, 0x8d7f, 0x8d80, 0x8d82, 0x8d83,
02273 0x8d86, 0x8d87, 0x8d88, 0x8d89, 0x8d8c, 0x8d8d, 0x8d8e, 0x8d8f,
02274 0x8d90, 0x8d92, 0x8d93, 0x8d95, 0x8d96, 0x8d97, 0x8d98, 0x8d99,
02275 0x8d9a, 0x8d9b, 0x8d9c, 0x8d9d, 0x8d9e, 0x8da0, 0x8da1, 0x8da2,
02276 0x8da4, 0x8da5, 0x8da6, 0x8da7, 0x8da8, 0x8da9, 0x8daa, 0x8dab,
02277 0x8dac, 0x8dad, 0x8dae, 0x8daf, 0x8db0, 0x8db2, 0x8db6, 0x8db7,
02278 0x8db9, 0x8dbb, 0x8dbd, 0x8dc0, 0x8dc1, 0x8dc2, 0x8dc5, 0x8dc7,
02279 0x8dc8, 0x8dc9, 0x8dca, 0x8dcd, 0x8dd0, 0x8dd2, 0x8dd3, 0x8dd4,
02280 0x51c7, 0x5196, 0x51a2, 0x51a5, 0x8ba0, 0x8ba6, 0x8ba7, 0x8baa,
02281 0x8bb4, 0x8bb5, 0x8bb7, 0x8bc2, 0x8bc3, 0x8bcb, 0x8bcf, 0x8bce,
02282 0x8bd2, 0x8bd3, 0x8bd4, 0x8bd6, 0x8bd8, 0x8bd9, 0x8bdc, 0x8bdf,
02283 0x8be0, 0x8be4, 0x8be8, 0x8be9, 0x8bee, 0x8bf0, 0x8bf3, 0x8bf6,
02284 0x8bf9, 0x8bfc, 0x8bff, 0x8c00, 0x8c02, 0x8c04, 0x8c07, 0x8c0c,
02285 0x8c0f, 0x8c11, 0x8c12, 0x8c14, 0x8c15, 0x8c16, 0x8c19, 0x8c1b,
02286 0x8c18, 0x8c1d, 0x8c1f, 0x8c20, 0x8c21, 0x8c25, 0x8c27, 0x8c2a,
02287 0x8c2b, 0x8c2e, 0x8c2f, 0x8c32, 0x8c33, 0x8c35, 0x8c36, 0x5369,
02288 0x537a, 0x961d, 0x9622, 0x9621, 0x9631, 0x962a, 0x963d, 0x963c,
02289 0x9642, 0x9649, 0x9654, 0x965f, 0x9667, 0x966c, 0x9672, 0x9674,
02290 0x9688, 0x968d, 0x9697, 0x96b0, 0x9097, 0x909b, 0x909d, 0x9099,
02291 0x90ac, 0x90a1, 0x90b4, 0x90b3, 0x90b6, 0x90ba,
02292 /* 0xdb */
02293 0x8dd5, 0x8dd8, 0x8dd9, 0x8ddc, 0x8de0, 0x8de1, 0x8de2, 0x8de5,
02294 0x8de6, 0x8de7, 0x8de9, 0x8ded, 0x8dee, 0x8df0, 0x8df1, 0x8df2,
02295 0x8df4, 0x8df6, 0x8dfc, 0x8dfe, 0x8dff, 0x8e00, 0x8e01, 0x8e02,
02296 0x8e03, 0x8e04, 0x8e06, 0x8e07, 0x8e08, 0x8e0b, 0x8e0d, 0x8e0e,
02297 0x8e10, 0x8e11, 0x8e12, 0x8e13, 0x8e15, 0x8e16, 0x8e17, 0x8e18,
02298 0x8e19, 0x8e1a, 0x8e1b, 0x8e1c, 0x8e20, 0x8e21, 0x8e24, 0x8e25,
02299 0x8e26, 0x8e27, 0x8e28, 0x8e2b, 0x8e2d, 0x8e30, 0x8e32, 0x8e33,
02300 0x8e34, 0x8e36, 0x8e37, 0x8e38, 0x8e3b, 0x8e3c, 0x8e3e, 0x8e3f,
02301 0x8e43, 0x8e45, 0x8e46, 0x8e4c, 0x8e4d, 0x8e4e, 0x8e4f, 0x8e50,
02302 0x8e53, 0x8e54, 0x8e55, 0x8e56, 0x8e57, 0x8e58, 0x8e5a, 0x8e5b,
02303 0x8e5c, 0x8e5d, 0x8e5e, 0x8e5f, 0x8e60, 0x8e61, 0x8e62, 0x8e63,
02304 0x8e64, 0x8e65, 0x8e67, 0x8e68, 0x8e6a, 0x8e6b, 0x8e6e, 0x8e71,
02305 0x90b8, 0x90b0, 0x90cf, 0x90c5, 0x90be, 0x90d0, 0x90c4, 0x90c7,
02306 0x90d3, 0x90e6, 0x90e2, 0x90dc, 0x90d7, 0x90db, 0x90eb, 0x90ef,
02307 0x90fe, 0x9104, 0x9122, 0x911e, 0x9123, 0x9131, 0x912f, 0x9139,
02308 0x9143, 0x9146, 0x520d, 0x5942, 0x52a2, 0x52ac, 0x52ad, 0x52be,
02309 0x54ff, 0x52d0, 0x52d6, 0x52f0, 0x53df, 0x71ee, 0x77cd, 0x5ef4,
02310 0x51f5, 0x51fc, 0x9b2f, 0x53b6, 0x5f01, 0x755a, 0x5def, 0x574c,
02311 0x57a9, 0x57a1, 0x587e, 0x58bc, 0x58c5, 0x58d1, 0x5729, 0x572c,
02312 0x572a, 0x5733, 0x5739, 0x572e, 0x572f, 0x572c, 0x573b, 0x5742,
02313 0x5769, 0x5785, 0x576b, 0x5786, 0x577c, 0x577b, 0x5768, 0x576d,
02314 0x5776, 0x5773, 0x57ad, 0x57a4, 0x578c, 0x57b2, 0x57cf, 0x57a7,
02315 0x57b4, 0x5793, 0x57a0, 0x57d5, 0x57d8, 0x57da, 0x57d9, 0x57d2,
02316 0x57b8, 0x57f4, 0x57ef, 0x57f8, 0x57e4, 0x57dd,
02317 /* 0xdc */
02318 0x8e73, 0x8e75, 0x8e77, 0x8e78, 0x8e79, 0x8e7a, 0x8e7b, 0x8e7d,
02319 0x8e7e, 0x8e80, 0x8e82, 0x8e83, 0x8e84, 0x8e86, 0x8e88, 0x8e89,
02320 0x8e8a, 0x8e8b, 0x8e8c, 0x8e8d, 0x8e8e, 0x8e91, 0x8e92, 0x8e93,
02321 0x8e95, 0x8e96, 0x8e97, 0x8e98, 0x8e99, 0x8e9a, 0x8e9b, 0x8e9d,
02322 0x8e9f, 0x8ea0, 0x8ea1, 0x8ea2, 0x8ea3, 0x8ea4, 0x8ea5, 0x8ea6,
02323 0x8ea7, 0x8ea8, 0x8ea9, 0x8eaa, 0x8ead, 0x8eae, 0x8eb0, 0x8eb1,
02324 0x8eb3, 0x8eb4, 0x8eb5, 0x8eb6, 0x8eb7, 0x8eb8, 0x8eb9, 0x8ebb,
02325 0x8ebc, 0x8ebd, 0x8ebe, 0x8ebf, 0x8ec0, 0x8ec1, 0x8ec2, 0x8ec3,
02326 0x8ec4, 0x8ec5, 0x8ec6, 0x8ec7, 0x8ec8, 0x8ec9, 0x8eca, 0x8ecb,
02327 0x8ecc, 0x8ecd, 0x8ecf, 0x8ed0, 0x8ed1, 0x8ed2, 0x8ed3, 0x8ed4,
02328 0x8ed5, 0x8ed6, 0x8ed7, 0x8ed8, 0x8ed9, 0x8eda, 0x8edb, 0x8edc,
02329 0x8edd, 0x8ede, 0x8edf, 0x8ee0, 0x8ee1, 0x8ee2, 0x8ee3, 0x8ee4,
02330 0x580b, 0x580d, 0x57fd, 0x57ed, 0x5800, 0x581e, 0x5819, 0x5844,
02331 0x5820, 0x5865, 0x586c, 0x5881, 0x5889, 0x589a, 0x5880, 0x99a8,
02332 0x9f19, 0x61ff, 0x8279, 0x827d, 0x827f, 0x828f, 0x828a, 0x82a8,
02333 0x8284, 0x828e, 0x8291, 0x8297, 0x8299, 0x82ab, 0x82b8, 0x82be,
02334 0x82b0, 0x82c8, 0x82ca, 0x82e3, 0x8298, 0x82b7, 0x82ae, 0x82cb,
02335 0x82cc, 0x82c1, 0x82a9, 0x82b4, 0x82a1, 0x82aa, 0x829f, 0x82c4,
02336 0x82ce, 0x82a4, 0x82e1, 0x8309, 0x82f7, 0x82e4, 0x830f, 0x8307,
02337 0x82dc, 0x82f4, 0x82d2, 0x82d8, 0x830c, 0x82fb, 0x82d3, 0x8311,
02338 0x831a, 0x8306, 0x8314, 0x8315, 0x82e0, 0x82d5, 0x831c, 0x8351,
02339 0x835b, 0x835c, 0x8308, 0x8392, 0x833c, 0x8334, 0x8331, 0x839b,
02340 0x835e, 0x832f, 0x834f, 0x8347, 0x8343, 0x835f, 0x8340, 0x8317,
02341 0x8360, 0x832d, 0x833a, 0x8333, 0x8366, 0x8365,
02342 /* 0xdd */
02343 0x8ee5, 0x8ee6, 0x8ee7, 0x8ee8, 0x8ee9, 0x8eea, 0x8eeb, 0x8eec,
02344 0x8eed, 0x8eee, 0x8eef, 0x8ef0, 0x8ef1, 0x8ef2, 0x8ef3, 0x8ef4,
02345 0x8ef5, 0x8ef6, 0x8ef7, 0x8ef8, 0x8ef9, 0x8efa, 0x8efb, 0x8efc,
02346 0x8efd, 0x8efe, 0x8eff, 0x8f00, 0x8f01, 0x8f02, 0x8f03, 0x8f04,
02347 0x8f05, 0x8f06, 0x8f07, 0x8f08, 0x8f09, 0x8f0a, 0x8f0b, 0x8f0c,
02348 0x8f0d, 0x8f0e, 0x8f0f, 0x8f10, 0x8f11, 0x8f12, 0x8f13, 0x8f14,
02349 0x8f15, 0x8f16, 0x8f17, 0x8f18, 0x8f19, 0x8f1a, 0x8f1b, 0x8f1c,
02350 0x8f1d, 0x8f1e, 0x8f1f, 0x8f20, 0x8f21, 0x8f22, 0x8f23, 0x8f24,
02351 0x8f25, 0x8f26, 0x8f27, 0x8f28, 0x8f29, 0x8f2a, 0x8f2b, 0x8f2c,
02352 0x8f2d, 0x8f2e, 0x8f2f, 0x8f30, 0x8f31, 0x8f32, 0x8f33, 0x8f34,
02353 0x8f35, 0x8f36, 0x8f37, 0x8f38, 0x8f39, 0x8f3a, 0x8f3b, 0x8f3c,
02354 0x8f3d, 0x8f3e, 0x8f3f, 0x8f40, 0x8f41, 0x8f42, 0x8f43, 0x8f44,
```

```
02355 0x8368, 0x831b, 0x8369, 0x836c, 0x836a, 0x836d, 0x836e, 0x83b0,
02356 0x8378, 0x83b3, 0x83b4, 0x83a0, 0x83aa, 0x8393, 0x839c, 0x8385,
02357 0x837c, 0x83b6, 0x83b7, 0x83a9, 0x837d, 0x83b8, 0x837b, 0x8398, 0x839e,
02358 0x83a8, 0x83ba, 0x83bc, 0x83c1, 0x8401, 0x83e5, 0x83d8, 0x5807,
02359 0x8418, 0x840b, 0x83dd, 0x83fd, 0x83d6, 0x841c, 0x8438, 0x8411,
02360 0x8406, 0x83d4, 0x83df, 0x840f, 0x8403, 0x83f8, 0x83f9, 0x83ea,
02361 0x83c5, 0x83c0, 0x8426, 0x83f0, 0x83e1, 0x845c, 0x8451, 0x845a,
02362 0x8459, 0x8473, 0x8487, 0x8488, 0x847a, 0x8489, 0x8478, 0x843c,
02363 0x8446, 0x8469, 0x8476, 0x848c, 0x848e, 0x8431, 0x846d, 0x84c1,
02364 0x84cd, 0x84d0, 0x84e6, 0x84bd, 0x84d3, 0x84ca, 0x84bf, 0x84ba,
02365 0x84e0, 0x84a1, 0x84b9, 0x84b4, 0x8497, 0x84e5, 0x84e3, 0x850c,
02366 0x750d, 0x8538, 0x84f0, 0x8539, 0x851f, 0x853a,
02367 /* 0xde */
02368 0x8f45, 0x8f46, 0x8f47, 0x8f48, 0x8f49, 0x8f4a, 0x8f4b, 0x8f4c,
02369 0x8f4d, 0x8f4e, 0x8f4f, 0x8f50, 0x8f51, 0x8f52, 0x8f53, 0x8f54,
02370 0x8f55, 0x8f56, 0x8f57, 0x8f58, 0x8f59, 0x8f5a, 0x8f5b, 0x8f5c,
02371 0x8f5d, 0x8f5e, 0x8f5f, 0x8f60, 0x8f61, 0x8f62, 0x8f63, 0x8f64,
02372 0x8f65, 0x8f6a, 0x8f80, 0x8f8c, 0x8f92, 0x8f9d, 0x8fa0, 0x8fa1,
02373 0x8fa2, 0x8fa4, 0x8fa5, 0x8fa6, 0x8fa7, 0x8faa, 0x8fac, 0x8fad,
02374 0x8fae, 0x8faf, 0x8fb2, 0x8fb3, 0x8fb4, 0x8fb5, 0x8fb7, 0x8fb8,
02375 0x8fba, 0x8fbb, 0x8fbc, 0x8fbf, 0x8fc0, 0x8fc3, 0x8fc6, 0x8fc9,
02376 0x8fca, 0x8fcb, 0xfcc, 0xfcd, 0xfcf, 0xfd2, 0xfd6, 0xfd7,
02377 0xfda, 0xfe0, 0xfe1, 0xfe3, 0xfe7, 0xfec, 0xfeF, 0xff1,
02378 0xff2, 0xff4, 0xff5, 0xff6, 0xffa, 0xffb, 0xffc, 0xffe,
02379 0xffff, 0x9007, 0x9008, 0x900c, 0x900e, 0x9013, 0x9015, 0x9018,
02380 0x8556, 0x853b, 0x84ff, 0x84fc, 0x8559, 0x8548, 0x8568, 0x8564,
02381 0x855e, 0x857a, 0x77a2, 0x8543, 0x8572, 0x857b, 0x85a4, 0x85a8,
02382 0x8587, 0x858f, 0x8579, 0x85ae, 0x859c, 0x8585, 0x85b9, 0x85b7,
02383 0x85b0, 0x85d3, 0x85c1, 0x85dc, 0x85ff, 0x8627, 0x8605, 0x8629,
02384 0x8616, 0x863c, 0x5efe, 0x5f08, 0x593c, 0x5941, 0x8037, 0x5955,
02385 0x595a, 0x5958, 0x530f, 0x5c22, 0x5c25, 0x5c2c, 0x5c34, 0x624c,
02386 0x626a, 0x629f, 0x62bb, 0x62ca, 0x62da, 0x62d7, 0x62ee, 0x6322,
02387 0x6046, 0x6339, 0x634b, 0x6343, 0x63ad, 0x63f6, 0x6371, 0x637a,
02388 0x638e, 0x63b4, 0x636d, 0x63ac, 0x638a, 0x6369, 0x63ae, 0x63bc,
02389 0x63f2, 0x63f8, 0x63e0, 0x63ff, 0x63c4, 0x63de, 0x63ce, 0x6452,
02390 0x63c6, 0x63be, 0x6445, 0x6441, 0x640b, 0x641b, 0x6420, 0x640c,
02391 0x6426, 0x6421, 0x645e, 0x6484, 0x646d, 0x6496,
02392 /* 0xdf */
02393 0x9019, 0x901c, 0x9023, 0x9024, 0x9025, 0x9027, 0x9028, 0x9029,
02394 0x902a, 0x902b, 0x902c, 0x9030, 0x9031, 0x9032, 0x9033, 0x9034,
02395 0x9037, 0x9039, 0x903a, 0x903d, 0x903f, 0x9040, 0x9043, 0x9045,
02396 0x9046, 0x9048, 0x9049, 0x904a, 0x904b, 0x904c, 0x904e, 0x9054,
02397 0x9055, 0x9056, 0x9059, 0x905a, 0x905c, 0x905d, 0x905e, 0x905f,
02398 0x9060, 0x9061, 0x9064, 0x9066, 0x9067, 0x9069, 0x906a, 0x906b,
02399 0x906c, 0x906f, 0x9070, 0x9071, 0x9072, 0x9073, 0x9076, 0x9077,
02400 0x9078, 0x9079, 0x907a, 0x907b, 0x907c, 0x907e, 0x9081, 0x9084,
02401 0x9085, 0x9086, 0x9087, 0x9089, 0x908a, 0x908c, 0x908d, 0x908e,
02402 0x908f, 0x9090, 0x9092, 0x9094, 0x9096, 0x9098, 0x909a, 0x909c,
02403 0x909e, 0x909f, 0x90a0, 0x90a4, 0x90a5, 0x90a7, 0x90a8, 0x90a9,
02404 0x90ab, 0x90ad, 0x90b2, 0x90b7, 0x90bc, 0x90bd, 0x90bf, 0x90c0,
02405 0x647a, 0x64b7, 0x64b8, 0x6499, 0x64ba, 0x64c0, 0x64d0, 0x64d7,
02406 0x64e4, 0x64e2, 0x6509, 0x6525, 0x652e, 0x5f0b, 0x5fd2, 0x7519,
02407 0x5f11, 0x535f, 0x53f1, 0x53fd, 0x53e9, 0x53e8, 0x53fb, 0x5412,
02408 0x5416, 0x5406, 0x544b, 0x5452, 0x5453, 0x5454, 0x5456, 0x5443,
02409 0x5421, 0x5457, 0x5459, 0x5423, 0x5432, 0x5482, 0x5494, 0x5477,
02410 0x5471, 0x5464, 0x549a, 0x549b, 0x5484, 0x5476, 0x5466, 0x549d,
02411 0x54d0, 0x54ad, 0x54c2, 0x54b4, 0x54d2, 0x54a7, 0x54a6, 0x54d3,
02412 0x54d4, 0x5472, 0x54a3, 0x54d5, 0x54bb, 0x54bf, 0x54cc, 0x54d9,
02413 0x54da, 0x54dc, 0x54a9, 0x54aa, 0x54a4, 0x54dd, 0x54cf, 0x54de,
02414 0x551b, 0x54e7, 0x5520, 0x54fd, 0x5514, 0x54f3, 0x5522, 0x5523,
02415 0x550f, 0x5511, 0x5527, 0x552a, 0x5567, 0x558f, 0x55b5, 0x5549,
02416 0x556d, 0x5541, 0x5555, 0x553f, 0x5550, 0x553c,
02417 /* 0xe0 */
02418 0x90c2, 0x90c3, 0x90c6, 0x90c8, 0x90c9, 0x90cb, 0x90cc, 0x90cd,
02419 0x90d2, 0x90d4, 0x90d5, 0x90d6, 0x90d8, 0x90d9, 0x90da, 0x90de,
02420 0x90df, 0x90e0, 0x90e3, 0x90e4, 0x90e5, 0x90e9, 0x90ea, 0x90ec,
02421 0x90ee, 0x90f0, 0x90f1, 0x90f2, 0x90f3, 0x90f5, 0x90f6, 0x90f7,
02422 0x90f9, 0x90fa, 0x90fb, 0x90fc, 0x90ff, 0x9100, 0x9101, 0x9103,
02423 0x9105, 0x9106, 0x9107, 0x9108, 0x9109, 0x910a, 0x910b, 0x910c,
02424 0x910d, 0x910e, 0x910f, 0x9110, 0x9111, 0x9112, 0x9113, 0x9114,
02425 0x9115, 0x9116, 0x9117, 0x9118, 0x911a, 0x911b, 0x911c, 0x911d,
02426 0x911f, 0x9120, 0x9121, 0x9124, 0x9125, 0x9126, 0x9127, 0x9128,
02427 0x9129, 0x912a, 0x912b, 0x912c, 0x912d, 0x912e, 0x9130, 0x9132,
02428 0x9133, 0x9134, 0x9135, 0x9136, 0x9137, 0x9138, 0x913a, 0x913b,
02429 0x913c, 0x913d, 0x913e, 0x913f, 0x9140, 0x9141, 0x9142, 0x9144,
02430 0x5537, 0x5556, 0x5575, 0x5576, 0x5577, 0x5533, 0x5530, 0x555c,
02431 0x558b, 0x55d2, 0x5583, 0x55b1, 0x55b9, 0x5588, 0x5581, 0x559f,
02432 0x557e, 0x55d6, 0x5591, 0x557b, 0x55df, 0x55bd, 0x55be, 0x5594,
02433 0x5599, 0x55ea, 0x55f7, 0x55c9, 0x561f, 0x55d1, 0x55eb, 0x55ec,
02434 0x55d4, 0x55e6, 0x55dd, 0x55c4, 0x55ef, 0x55e5, 0x55f2, 0x55f3,
02435 0x55cc, 0x55cd, 0x55e8, 0x55f5, 0x55e4, 0x8f94, 0x561e, 0x5608,
02436 0x560c, 0x5601, 0x5624, 0x5623, 0x55fe, 0x5600, 0x5627, 0x562d,
02437 0x5658, 0x5639, 0x5657, 0x562c, 0x564d, 0x5662, 0x5659, 0x565c,
02438 0x564c, 0x5654, 0x5686, 0x5664, 0x5671, 0x566b, 0x567b, 0x567c,
02439 0x5685, 0x5693, 0x56af, 0x56d4, 0x56d7, 0x56dd, 0x56e1, 0x56f5,
02440 0x56eb, 0x56f9, 0x56ff, 0x5704, 0x570a, 0x5709, 0x571c, 0x5e0f,
02441 0x5e19, 0x5e14, 0x5e11, 0x5e31, 0x5e3b, 0x5e3c,
```

```

02442  /* 0xe1 */
02443  0x9145, 0x9147, 0x9148, 0x9151, 0x9153, 0x9154, 0x9155, 0x9156,
02444  0x9158, 0x9159, 0x915b, 0x915c, 0x915f, 0x9160, 0x9166, 0x9167,
02445  0x9168, 0x916b, 0x916d, 0x9173, 0x917a, 0x917b, 0x917c, 0x9180,
02446  0x9181, 0x9182, 0x9183, 0x9184, 0x9186, 0x9188, 0x918a, 0x918e,
02447  0x918f, 0x9193, 0x9194, 0x9195, 0x9196, 0x9197, 0x9198, 0x9199,
02448  0x919c, 0x919d, 0x919e, 0x919f, 0x91a0, 0x91a1, 0x91a4, 0x91a5,
02449  0x91a6, 0x91a7, 0x91a8, 0x91a9, 0x91ab, 0x91ac, 0x91b0, 0x91b1,
02450  0x91b2, 0x91b3, 0x91b6, 0x91b7, 0x91b8, 0x91b9, 0x91bb, 0x91bc,
02451  0x91bd, 0x91be, 0x91bf, 0x91c0, 0x91c1, 0x91c2, 0x91c3, 0x91c4,
02452  0x91c5, 0x91c6, 0x91c8, 0x91cb, 0x91d0, 0x91d2, 0x91d3, 0x91d4,
02453  0x91d5, 0x91d6, 0x91d7, 0x91d8, 0x91d9, 0x91da, 0x91db, 0x91dd,
02454  0x91de, 0x91df, 0x91e0, 0x91e1, 0x91e2, 0x91e3, 0x91e4, 0x91e5,
02455  0x5e37, 0x5e44, 0x5e54, 0x5e5b, 0x5e5e, 0x5e61, 0x5c8c, 0x5c7a,
02456  0x5c8d, 0x5c90, 0x5c96, 0x5c98, 0x5c99, 0x5c91, 0x5c9a,
02457  0x5c9c, 0x5cb5, 0x5ca2, 0x5cbd, 0x5cac, 0x5cab, 0x5cb1, 0x5ca3,
02458  0x5cc1, 0x5cb7, 0x5cc4, 0x5cd2, 0x5ce4, 0x5ccb, 0x5ce5, 0x5d02,
02459  0x5d03, 0x5d27, 0x5d26, 0x5d2e, 0x5d24, 0x5d1e, 0x5d06, 0x5d1b,
02460  0x5d58, 0x5d3e, 0x5d34, 0x5d3d, 0x5d6c, 0x5d5b, 0x5d6f, 0x5d5d,
02461  0x5d6b, 0x5d4b, 0x5d4a, 0x5d69, 0x5d74, 0x5d82, 0x5d99, 0x5d9d,
02462  0x8c73, 0x5db7, 0x5dc5, 0x5f73, 0x5f77, 0x5f82, 0x5f87, 0x5f89,
02463  0x5f8c, 0x5f95, 0x5f99, 0x5f9c, 0x5fa8, 0x5fad, 0x5fb5, 0x5fbc,
02464  0x8862, 0x5f61, 0x72ad, 0x72b0, 0x72b4, 0x72b7, 0x72b8, 0x72c3,
02465  0x72c1, 0x72ce, 0x72cd, 0x72d2, 0x72e8, 0x72ef, 0x72e9, 0x72f2,
02466  0x72f4, 0x72f7, 0x7301, 0x72f3, 0x7303, 0x72fa,
02467  /* 0xe2 */
02468  0x91e6, 0x91e7, 0x91e8, 0x91e9, 0x91ea, 0x91eb, 0x91ec, 0x91ed,
02469  0x91ee, 0x91ef, 0x91f0, 0x91f1, 0x91f2, 0x91f3, 0x91f4, 0x91f5,
02470  0x91f6, 0x91f7, 0x91f8, 0x91f9, 0x91fa, 0x91fb, 0x91fc, 0x91fd,
02471  0x91fe, 0x91ff, 0x9200, 0x9201, 0x9202, 0x9203, 0x9204, 0x9205,
02472  0x9206, 0x9207, 0x9208, 0x9209, 0x920a, 0x920b, 0x920c, 0x920d,
02473  0x920e, 0x920f, 0x9210, 0x9211, 0x9212, 0x9213, 0x9214, 0x9215,
02474  0x9216, 0x9217, 0x9218, 0x9219, 0x921a, 0x921b, 0x921c, 0x921d,
02475  0x921e, 0x921f, 0x9220, 0x9221, 0x9222, 0x9223, 0x9224, 0x9225,
02476  0x9226, 0x9227, 0x9228, 0x9229, 0x922a, 0x922b, 0x922c, 0x922d,
02477  0x922e, 0x922f, 0x9230, 0x9231, 0x9232, 0x9233, 0x9234, 0x9235,
02478  0x9236, 0x9237, 0x9238, 0x9239, 0x923a, 0x923b, 0x923c, 0x923d,
02479  0x923e, 0x923f, 0x9240, 0x9241, 0x9242, 0x9243, 0x9244, 0x9245,
02480  0x72fb, 0x7317, 0x7313, 0x7321, 0x730a, 0x731e, 0x731d, 0x7315,
02481  0x7322, 0x7339, 0x7325, 0x732c, 0x7338, 0x7331, 0x7350, 0x734d,
02482  0x7357, 0x7360, 0x736c, 0x736f, 0x737e, 0x821b, 0x5925, 0x98e7,
02483  0x5924, 0x5902, 0x9963, 0x9967, 0x9968, 0x9969, 0x996a, 0x996b,
02484  0x996c, 0x9974, 0x9977, 0x997d, 0x9980, 0x9984, 0x9987, 0x998a,
02485  0x998d, 0x9990, 0x9991, 0x9993, 0x9994, 0x9995, 0x5e80, 0x5e91,
02486  0x5e8b, 0x5e96, 0x5ea5, 0x5ea0, 0x5eb9, 0x5eb5, 0x5ebe, 0x5eb3,
02487  0x8d53, 0x5ed2, 0x5ed1, 0x5edb, 0x5ee8, 0x5eea, 0x81ba, 0x5fc4,
02488  0x5fc9, 0x5fd6, 0x5fcf, 0x6003, 0x5fee, 0x6004, 0x5fel, 0x5fe4,
02489  0x5ffe, 0x6005, 0x6006, 0x5fea, 0x5fed, 0x5ff8, 0x6019, 0x6035,
02490  0x6026, 0x601b, 0x600f, 0x600d, 0x6029, 0x602b, 0x600a, 0x603f,
02491  0x6021, 0x6078, 0x6079, 0x607b, 0x607a, 0x6042,
02492  /* 0xe3 */
02493  0x9246, 0x9247, 0x9248, 0x9249, 0x924a, 0x924b, 0x924c, 0x924d,
02494  0x924e, 0x924f, 0x9250, 0x9251, 0x9252, 0x9253, 0x9254, 0x9255,
02495  0x9256, 0x9257, 0x9258, 0x9259, 0x925a, 0x925b, 0x925c, 0x925d,
02496  0x925e, 0x925f, 0x9260, 0x9261, 0x9262, 0x9263, 0x9264, 0x9265,
02497  0x9266, 0x9267, 0x9268, 0x9269, 0x926a, 0x926b, 0x926c, 0x926d,
02498  0x926e, 0x926f, 0x9270, 0x9271, 0x9272, 0x9273, 0x9275, 0x9276,
02499  0x9277, 0x9278, 0x9279, 0x927a, 0x927b, 0x927c, 0x927d, 0x927e,
02500  0x927f, 0x9280, 0x9281, 0x9282, 0x9283, 0x9284, 0x9285, 0x9286,
02501  0x9287, 0x9288, 0x9289, 0x928a, 0x928b, 0x928c, 0x928d, 0x928f,
02502  0x9290, 0x9291, 0x9292, 0x9293, 0x9294, 0x9295, 0x9296, 0x9297,
02503  0x9298, 0x9299, 0x929a, 0x929b, 0x929c, 0x929d, 0x929e, 0x929f,
02504  0x92a0, 0x92a1, 0x92a2, 0x92a3, 0x92a4, 0x92a5, 0x92a6, 0x92a7,
02505  0x606a, 0x607d, 0x6096, 0x609a, 0x60ad, 0x609d, 0x6083, 0x6092,
02506  0x608c, 0x609b, 0x60ec, 0x60bb, 0x60b1, 0x60dd, 0x60d8, 0x60c6,
02507  0x60da, 0x60b4, 0x6120, 0x6126, 0x6115, 0x6123, 0x60f4, 0x6100,
02508  0x610e, 0x612b, 0x614a, 0x6175, 0x61ac, 0x6194, 0x61a7, 0x61b7,
02509  0x61d4, 0x61f5, 0x5fdd, 0x96b3, 0x95e9, 0x95eb, 0x95f1, 0x95f3,
02510  0x95f5, 0x95f6, 0x95fc, 0x95fe, 0x9603, 0x9604, 0x9606, 0x9608,
02511  0x960a, 0x960b, 0x960c, 0x960d, 0x960f, 0x9612, 0x9615, 0x9616,
02512  0x9617, 0x9619, 0x961a, 0x4e2c, 0x723f, 0x6215, 0x6c35, 0x6c54,
02513  0x6c5c, 0x6c4a, 0x6ca3, 0x6c85, 0x6c90, 0x6c94, 0x6c8c, 0x6c68,
02514  0x6c69, 0x6c74, 0x6c76, 0x6c86, 0x6ca9, 0x6cd0, 0x6cd4, 0x6cad,
02515  0x6cf7, 0x6cf8, 0x6cf1, 0x6cd7, 0x6cb2, 0x6ce0, 0x6cd6, 0x6cfa,
02516  0x6ceb, 0x6cee, 0x6cb1, 0x6cd3, 0x6cef, 0x6cfe,
02517  /* 0xe4 */
02518  0x92a8, 0x92a9, 0x92aa, 0x92ab, 0x92ac, 0x92ad, 0x92af, 0x92b0,
02519  0x92b1, 0x92b2, 0x92b3, 0x92b4, 0x92b5, 0x92b6, 0x92b7, 0x92b8,
02520  0x92b9, 0x92ba, 0x92bb, 0x92bc, 0x92bd, 0x92be, 0x92bf, 0x92c0,
02521  0x92c1, 0x92c2, 0x92c3, 0x92c4, 0x92c5, 0x92c6, 0x92c7, 0x92c9,
02522  0x92ca, 0x92cb, 0x92cc, 0x92cd, 0x92ce, 0x92cf, 0x92d0, 0x92d1,
02523  0x92d2, 0x92d3, 0x92d4, 0x92d5, 0x92d6, 0x92d7, 0x92d8, 0x92d9,
02524  0x92da, 0x92db, 0x92dc, 0x92dd, 0x92de, 0x92df, 0x92e0, 0x92e1,
02525  0x92e2, 0x92e3, 0x92e4, 0x92e5, 0x92e6, 0x92e7, 0x92e8, 0x92e9,
02526  0x92ea, 0x92eb, 0x92ec, 0x92ed, 0x92ee, 0x92ef, 0x92f0, 0x92f1,
02527  0x92f2, 0x92f3, 0x92f4, 0x92f5, 0x92f6, 0x92f7, 0x92f8, 0x92f9,
02528  0x92fa, 0x92fb, 0x92fc, 0x92fd, 0x92fe, 0x92ff, 0x9300, 0x9301,

```

```
02529 0x9302, 0x9303, 0x9304, 0x9305, 0x9306, 0x9307, 0x9308, 0x9309,
02530 0x6d39, 0x6d27, 0x6d0c, 0x6d43, 0x6d48, 0x6d07, 0x6d04, 0x6d19,
02531 0x6d0e, 0x6d2b, 0x6d4d, 0x6d2e, 0x6d35, 0x6d1a, 0x6d4f, 0x6d52,
02532 0x6d54, 0x6d33, 0x6d91, 0x6d6f, 0x6d9e, 0x6da0, 0x6d5e, 0x6d93,
02533 0x6d94, 0x6d5c, 0x6d60, 0x6d7c, 0x6d63, 0x6e1a, 0x6dc7, 0x6dc5,
02534 0x6dde, 0x6de0e, 0x6dbf, 0x6de0, 0x6e11, 0x6de6, 0x6ddd, 0x6dd9,
02535 0x6e16, 0x6dab, 0x6e0c, 0x6dae, 0x6e2b, 0x6e6e, 0x6e4e, 0x6e6b,
02536 0x6eb2, 0x6e5f, 0x6e86, 0x6e53, 0x6e54, 0x6e32, 0x6e25, 0x6e44,
02537 0x6edf, 0x6eb1, 0x6e98, 0x6ee0, 0x6f2d, 0x6ee2, 0x6ea5, 0x6ea7,
02538 0x6ebd, 0x6ebb, 0x6eb7, 0x6ed7, 0x6eb4, 0x6ecf, 0x6e8f, 0x6ec2,
02539 0x6e9f, 0x6f62, 0x6f46, 0x6f47, 0x6f24, 0x6f15, 0x6ef9, 0x6f2f,
02540 0x6f36, 0x6f4b, 0x6f74, 0x6f2a, 0x6f09, 0x6f29, 0x6f89, 0x6f8d,
02541 0x6f8c, 0x6f78, 0x6f72, 0x6f7c, 0x6f7a, 0x6fd1,
02542 /* 0xe5 */
02543 0x930a, 0x930b, 0x930c, 0x930d, 0x930e, 0x930f, 0x9310, 0x9311,
02544 0x9312, 0x9313, 0x9314, 0x9315, 0x9316, 0x9317, 0x9318, 0x9319,
02545 0x931a, 0x931b, 0x931c, 0x931d, 0x931e, 0x931f, 0x9320, 0x9321,
02546 0x9322, 0x9323, 0x9324, 0x9325, 0x9326, 0x9327, 0x9328, 0x9329,
02547 0x932a, 0x932b, 0x932c, 0x932d, 0x932e, 0x932f, 0x9330, 0x9331,
02548 0x9332, 0x9333, 0x9334, 0x9335, 0x9336, 0x9337, 0x9338, 0x9339,
02549 0x933a, 0x933b, 0x933c, 0x933d, 0x933e, 0x933f, 0x9340, 0x9341, 0x9342,
02550 0x9343, 0x9344, 0x9345, 0x9346, 0x9347, 0x9348, 0x9349, 0x934a,
02551 0x934b, 0x934c, 0x934d, 0x934e, 0x934f, 0x9350, 0x9351, 0x9352,
02552 0x9353, 0x9354, 0x9355, 0x9356, 0x9357, 0x9358, 0x9359, 0x935a,
02553 0x935b, 0x935c, 0x935d, 0x935e, 0x935f, 0x9360, 0x9361, 0x9362,
02554 0x9363, 0x9364, 0x9365, 0x9366, 0x9367, 0x9368, 0x9369, 0x936a,
02555 0x6fc9, 0x6fa7, 0x6fb9, 0x6fb6, 0x6fc2, 0x6fe1, 0x6fee, 0x6fde,
02556 0x6fe0, 0x6fef, 0x701a, 0x7023, 0x701b, 0x7039, 0x7035, 0x704f,
02557 0x705e, 0x5b80, 0x5b84, 0x5b95, 0x5b93, 0x5ba5, 0x5bb8, 0x752f,
02558 0x9a9e, 0x6434, 0x5be4, 0x5bee, 0x8930, 0x5bf0, 0x8e47, 0x8b07,
02559 0x8fb6, 0x8fd3, 0x8fd5, 0x8fe5, 0x8fee, 0x8fe4, 0x8fe9, 0x8fe6,
02560 0x8ff3, 0x8fe8, 0x9005, 0x9004, 0x900b, 0x9026, 0x9011, 0x900d,
02561 0x9016, 0x9021, 0x9035, 0x9036, 0x902d, 0x902f, 0x9044, 0x9051,
02562 0x9052, 0x9050, 0x9068, 0x9058, 0x9062, 0x905b, 0x66b9, 0x9074,
02563 0x907d, 0x9082, 0x9088, 0x9083, 0x908b, 0x5f50, 0x5f57, 0x5f56,
02564 0x5f58, 0x5c3b, 0x54ab, 0x5c50, 0x5c59, 0x5b71, 0x5c63, 0x5c66,
02565 0x7fbc, 0x5f2a, 0x5f29, 0x5f2d, 0x8274, 0x5f3c, 0x9b3b, 0x5c6e,
02566 0x5981, 0x5983, 0x598d, 0x59a9, 0x59aa, 0x59a3,
02567 /* 0xe6 */
02568 0x936c, 0x936d, 0x936e, 0x936f, 0x9370, 0x9371, 0x9372, 0x9373,
02569 0x9374, 0x9375, 0x9376, 0x9377, 0x9378, 0x9379, 0x937a, 0x937b,
02570 0x937c, 0x937d, 0x937e, 0x937f, 0x9380, 0x9381, 0x9382, 0x9383,
02571 0x9384, 0x9385, 0x9386, 0x9387, 0x9388, 0x9389, 0x938a, 0x938b,
02572 0x938c, 0x938d, 0x938e, 0x9390, 0x9391, 0x9392, 0x9393, 0x9394,
02573 0x9395, 0x9396, 0x9397, 0x9398, 0x9399, 0x939a, 0x939b, 0x939c,
02574 0x939d, 0x939e, 0x939f, 0x93a0, 0x93a1, 0x93a2, 0x93a3, 0x93a4,
02575 0x93a5, 0x93a6, 0x93a7, 0x93a8, 0x93a9, 0x93aa, 0x93ab, 0x93ac,
02576 0x93ad, 0x93ae, 0x93af, 0x93b0, 0x93b1, 0x93b2, 0x93b3, 0x93b4,
02577 0x93b5, 0x93b6, 0x93b7, 0x93b8, 0x93b9, 0x93ba, 0x93bb, 0x93bc,
02578 0x93bd, 0x93be, 0x93bf, 0x93c0, 0x93c1, 0x93c2, 0x93c3, 0x93c4,
02579 0x93c5, 0x93c6, 0x93c7, 0x93c8, 0x93c9, 0x93cb, 0x93cc, 0x93cd,
02580 0x5997, 0x59ca, 0x59ab, 0x599e, 0x59a4, 0x59d2, 0x59b2, 0x59af,
02581 0x59d7, 0x59be, 0x5a05, 0x5a06, 0x59dd, 0x5a08, 0x59e3, 0x59d8,
02582 0x59f9, 0x5a0c, 0x5a09, 0x5a32, 0x5a34, 0x5a11, 0x5a23, 0x5a13,
02583 0x5a40, 0x5a67, 0x5a4a, 0x5a55, 0x5a3c, 0x5a62, 0x5a75, 0x80ec,
02584 0x5aa4, 0x5a9b, 0x5a77, 0x5a7a, 0x5abe, 0x5aeb, 0x5ab2, 0x5ad2,
02585 0x5ad4, 0x5ab8, 0x5ae0, 0x5ae3, 0x5af1, 0x5ad6, 0x5ae6, 0x5ad8,
02586 0x5adc, 0x5b09, 0x5b17, 0x5b16, 0x5b32, 0x5b37, 0x5b40, 0x5c15,
02587 0x5c1c, 0x5b5a, 0x5b65, 0x5b73, 0x5b51, 0x5b53, 0x5b62, 0x9a75,
02588 0x9a77, 0x9a78, 0x9a7a, 0x9a7f, 0x9a7d, 0x9a80, 0x9a81, 0x9a85,
02589 0x9a88, 0x9a8a, 0x9a90, 0x9a92, 0x9a93, 0x9a96, 0x9a98, 0x9a9b,
02590 0x9a9c, 0x9a9d, 0x9a9f, 0x9aa0, 0x9aa2, 0x9aa3, 0x9aa5, 0x9aa7,
02591 0x7e9f, 0x7ea1, 0x7ea3, 0x7ea5, 0x7ea8, 0x7ea9,
02592 /* 0xe7 */
02593 0x93ce, 0x93cf, 0x93d0, 0x93d1, 0x93d2, 0x93d3, 0x93d4, 0x93d5,
02594 0x93d7, 0x93d8, 0x93d9, 0x93da, 0x93db, 0x93dc, 0x93dd, 0x93de,
02595 0x93df, 0x93e0, 0x93e1, 0x93e2, 0x93e3, 0x93e4, 0x93e5, 0x93e6,
02596 0x93e7, 0x93e8, 0x93e9, 0x93ea, 0x93eb, 0x93ec, 0x93ed, 0x93ee,
02597 0x93ef, 0x93f0, 0x93f1, 0x93f2, 0x93f3, 0x93f4, 0x93f5, 0x93f6,
02598 0x93f7, 0x93f8, 0x93f9, 0x93fa, 0x93fb, 0x93fc, 0x93fd, 0x93fe,
02599 0x93ff, 0x9400, 0x9401, 0x9402, 0x9403, 0x9404, 0x9405, 0x9406,
02600 0x9407, 0x9408, 0x9409, 0x940a, 0x940b, 0x940c, 0x940d, 0x940e,
02601 0x940f, 0x9410, 0x9411, 0x9412, 0x9413, 0x9414, 0x9415, 0x9416,
02602 0x9417, 0x9418, 0x9419, 0x941a, 0x941b, 0x941c, 0x941d, 0x941e,
02603 0x941f, 0x9420, 0x9421, 0x9422, 0x9423, 0x9424, 0x9425, 0x9426,
02604 0x9427, 0x9428, 0x9429, 0x942a, 0x942b, 0x942c, 0x942d, 0x942e,
02605 0x7ead, 0x7eb0, 0x7ebe, 0x7ec0, 0x7ec1, 0x7ec2, 0x7ec9, 0x7ecb,
02606 0x7ecc, 0x7ed0, 0x7ed4, 0x7ed7, 0x7edb, 0x7ee0, 0x7ee1, 0x7ee8,
02607 0x7eeb, 0x7eee, 0x7eef, 0x7ef1, 0x7ef2, 0x7f0d, 0x7ef6, 0x7efa,
02608 0x7efb, 0x7efe, 0x7ef0, 0x7ef1, 0x7ef2, 0x7ef3, 0x7ef7, 0x7ef8, 0x7f0b,
02609 0x7f0c, 0x7f0f, 0x7f11, 0x7f12, 0x7f17, 0x7f19, 0x7f1c, 0x7f1b,
02610 0x7f1f, 0x7f21, 0x7f22, 0x7f23, 0x7f24, 0x7f25, 0x7f26, 0x7f27,
02611 0x7f2a, 0x7f2b, 0x7f2c, 0x7f2d, 0x7f2f, 0x7f30, 0x7f31, 0x7f32,
02612 0x7f33, 0x7f35, 0x5e7a, 0x757f, 0x5ddb, 0x753e, 0x9095, 0x738e,
02613 0x7391, 0x73ae, 0x73a2, 0x739f, 0x73cf, 0x73c2, 0x73d1, 0x73b7,
02614 0x73b3, 0x73c0, 0x73c1, 0x73c8, 0x73c9, 0x73e5, 0x73d9, 0x987c, 0x740a,
02615 0x73e9, 0x73e7, 0x73de, 0x73ba, 0x73f2, 0x740f, 0x742a, 0x745b,
```

```

02616 0x7426, 0x7425, 0x7428, 0x7430, 0x742e, 0x742c,
02617 /* 0xe8 */
02618 0x942f, 0x9430, 0x9431, 0x9432, 0x9433, 0x9434, 0x9435, 0x9436,
02619 0x9437, 0x9438, 0x9439, 0x943a, 0x943b, 0x943c, 0x943d, 0x943f,
02620 0x9440, 0x9441, 0x9442, 0x9443, 0x9444, 0x9445, 0x9446, 0x9447,
02621 0x9448, 0x9449, 0x944a, 0x944b, 0x944c, 0x944d, 0x944e, 0x944f,
02622 0x9450, 0x9451, 0x9452, 0x9453, 0x9454, 0x9455, 0x9456, 0x9457,
02623 0x9458, 0x9459, 0x945a, 0x945b, 0x945c, 0x945d, 0x945e, 0x945f,
02624 0x9460, 0x9461, 0x9462, 0x9463, 0x9464, 0x9465, 0x9466, 0x9467,
02625 0x9468, 0x9469, 0x946a, 0x946c, 0x946d, 0x946e, 0x946f, 0x9470,
02626 0x9471, 0x9472, 0x9473, 0x9474, 0x9475, 0x9476, 0x9477, 0x9478,
02627 0x9479, 0x947a, 0x947b, 0x947c, 0x947d, 0x947e, 0x947f, 0x9480,
02628 0x9481, 0x9482, 0x9483, 0x9484, 0x9491, 0x9496, 0x9498, 0x94c7,
02629 0x94cf, 0x94d3, 0x94d4, 0x94da, 0x94e6, 0x94fb, 0x951c, 0x9520,
02630 0x741b, 0x741a, 0x7441, 0x7441, 0x745c, 0x7457, 0x7455, 0x7459, 0x7477,
02631 0x746d, 0x747e, 0x749c, 0x748e, 0x7480, 0x7481, 0x7487, 0x748b,
02632 0x749e, 0x74a8, 0x74a9, 0x7490, 0x74a7, 0x74d2, 0x74ba, 0x97ea,
02633 0x97eb, 0x97ec, 0x674c, 0x6753, 0x675e, 0x6748, 0x6769, 0x67a5,
02634 0x6787, 0x676a, 0x6773, 0x6798, 0x67a7, 0x6775, 0x67a8, 0x679e,
02635 0x67ad, 0x678b, 0x6777, 0x677c, 0x67f0, 0x6809, 0x67d8, 0x680a,
02636 0x67e9, 0x67b0, 0x680c, 0x67d9, 0x67b5, 0x67da, 0x67b3, 0x67dd,
02637 0x6800, 0x67c3, 0x67b8, 0x67e2, 0x680e, 0x67c1, 0x67fd, 0x6832,
02638 0x6833, 0x6860, 0x6861, 0x684e, 0x6862, 0x6844, 0x6864, 0x6883,
02639 0x681d, 0x6855, 0x6866, 0x6841, 0x6867, 0x6840, 0x683e, 0x684a,
02640 0x6849, 0x6829, 0x68b5, 0x688f, 0x6874, 0x6877, 0x6893, 0x686b,
02641 0x68c2, 0x696e, 0x68fc, 0x691f, 0x6920, 0x68f9,
02642 /* 0xe9 */
02643 0x9527, 0x9533, 0x953d, 0x9543, 0x9548, 0x954b, 0x9555, 0x955a,
02644 0x9560, 0x956e, 0x9574, 0x9575, 0x9577, 0x9578, 0x9579, 0x957a,
02645 0x957b, 0x957c, 0x957d, 0x957e, 0x9580, 0x9581, 0x9582, 0x9583,
02646 0x9584, 0x9585, 0x9586, 0x9587, 0x9588, 0x9589, 0x958a, 0x958b,
02647 0x958c, 0x958d, 0x958e, 0x958f, 0x9590, 0x9591, 0x9592, 0x9593,
02648 0x9594, 0x9595, 0x9596, 0x9597, 0x9598, 0x9599, 0x959a, 0x959b,
02649 0x959c, 0x959d, 0x959e, 0x959f, 0x95a0, 0x95a1, 0x95a2, 0x95a3,
02650 0x95a4, 0x95a5, 0x95a6, 0x95a7, 0x95a8, 0x95a9, 0x95aa, 0x95ab,
02651 0x95ac, 0x95ad, 0x95ae, 0x95af, 0x95b0, 0x95b1, 0x95b2, 0x95b3,
02652 0x95b4, 0x95b5, 0x95b6, 0x95b7, 0x95b8, 0x95b9, 0x95ba, 0x95bb,
02653 0x95bc, 0x95bd, 0x95be, 0x95bf, 0x95c0, 0x95c1, 0x95c2, 0x95c3,
02654 0x95c4, 0x95c5, 0x95c6, 0x95c7, 0x95c8, 0x95c9, 0x95ca, 0x95cb,
02655 0x6924, 0x68f0, 0x690b, 0x6901, 0x6957, 0x68e3, 0x6910, 0x6971,
02656 0x6939, 0x6960, 0x6942, 0x695d, 0x6984, 0x696b, 0x6980, 0x6998,
02657 0x6978, 0x6934, 0x69cc, 0x6987, 0x6988, 0x69ce, 0x6989, 0x6966,
02658 0x6963, 0x6979, 0x699b, 0x69a7, 0x69bb, 0x69ab, 0x69ad, 0x69d4,
02659 0x69b1, 0x69c1, 0x69ca, 0x69df, 0x6995, 0x69e0, 0x698d, 0x69ff,
02660 0x6a2f, 0x69ed, 0x6a17, 0x6a18, 0x6a65, 0x69f2, 0x6a44, 0x6a3e,
02661 0x6aa0, 0x6a50, 0x6a5b, 0x6a35, 0x6a8e, 0x6a79, 0x6a3d, 0x6a28,
02662 0x6a58, 0x6a7c, 0x6a91, 0x6a90, 0x6aa9, 0x6a97, 0x6aab, 0x7337,
02663 0x7352, 0x6b81, 0x6b82, 0x6b87, 0x6b84, 0x6b92, 0x6b93, 0x6b8d,
02664 0x6b9a, 0x6b9b, 0x6ba1, 0x6baa, 0x8f6b, 0x8f6d, 0x8f71, 0x8f72,
02665 0x8f73, 0x8f75, 0x8f76, 0x8f78, 0x8f77, 0x8f79, 0x8f7a, 0x8f7c,
02666 0x8f7e, 0x8f81, 0x8f82, 0x8f84, 0x8f87, 0x8f8b,
02667 /* 0xea */
02668 0x95cc, 0x95cd, 0x95ce, 0x95cf, 0x95d0, 0x95d1, 0x95d2, 0x95d3,
02669 0x95d4, 0x95d5, 0x95d6, 0x95d7, 0x95d8, 0x95d9, 0x95da, 0x95db,
02670 0x95dc, 0x95dd, 0x95de, 0x95df, 0x95e0, 0x95e1, 0x95e2, 0x95e3,
02671 0x95e4, 0x95e5, 0x95e6, 0x95e7, 0x95ec, 0x95ff, 0x9607, 0x9613,
02672 0x9618, 0x961b, 0x961e, 0x9620, 0x9623, 0x9624, 0x9625, 0x9626,
02673 0x9627, 0x9628, 0x9629, 0x962b, 0x962c, 0x962d, 0x962f, 0x9630,
02674 0x9637, 0x9638, 0x9639, 0x963a, 0x963e, 0x9641, 0x9643, 0x964a,
02675 0x964e, 0x964f, 0x9651, 0x9652, 0x9653, 0x9656, 0x9657, 0x9658,
02676 0x9659, 0x965a, 0x965c, 0x965d, 0x965e, 0x9660, 0x9663, 0x9665,
02677 0x9666, 0x966b, 0x966d, 0x966e, 0x966f, 0x9670, 0x9671, 0x9673,
02678 0x9678, 0x9679, 0x967a, 0x967b, 0x967c, 0x967d, 0x967e, 0x967f,
02679 0x9680, 0x9681, 0x9682, 0x9683, 0x9684, 0x9687, 0x9689, 0x968a,
02680 0x8f8d, 0x8f8e, 0x8f8f, 0x8f98, 0x8f9a, 0x8ece, 0x620b, 0x6217,
02681 0x621b, 0x621f, 0x6222, 0x6221, 0x6225, 0x6224, 0x622c, 0x81e7,
02682 0x74ef, 0x74ff, 0x750f, 0x7511, 0x7513, 0x6534, 0x65ee,
02683 0x65ef, 0x65f0, 0x660a, 0x6619, 0x6772, 0x6603, 0x6615, 0x6600,
02684 0x7085, 0x66f7, 0x661d, 0x6634, 0x6631, 0x6636, 0x6635, 0x8006,
02685 0x665f, 0x6654, 0x6641, 0x664f, 0x6656, 0x6661, 0x6657, 0x6677,
02686 0x6684, 0x668c, 0x66a7, 0x669d, 0x66be, 0x66db, 0x66dc, 0x66e6,
02687 0x66e9, 0x8d32, 0x8d33, 0x8d36, 0x8d3b, 0x8d3d, 0x8d40, 0x8d45,
02688 0x8d46, 0x8d48, 0x8d49, 0x8d47, 0x8d4d, 0x8d55, 0x8d59, 0x89c7,
02689 0x89ca, 0x89cb, 0x89cc, 0x89ce, 0x89cf, 0x89d0, 0x89d1, 0x726e,
02690 0x729f, 0x725d, 0x7266, 0x726f, 0x727e, 0x727f, 0x7284, 0x728b,
02691 0x728d, 0x728f, 0x7292, 0x6308, 0x6332, 0x63b0,
02692 /* 0xeb */
02693 0x968c, 0x968e, 0x9691, 0x9692, 0x9693, 0x9695, 0x9696, 0x969a,
02694 0x969b, 0x969d, 0x969e, 0x969f, 0x96a0, 0x96a1, 0x96a2, 0x96a3,
02695 0x96a4, 0x96a5, 0x96a6, 0x96a8, 0x96a9, 0x96aa, 0x96ab, 0x96ac,
02696 0x96ad, 0x96ae, 0x96af, 0x96b1, 0x96b2, 0x96b4, 0x96b5, 0x96b7,
02697 0x96b8, 0x96ba, 0x96bb, 0x96bf, 0x96c2, 0x96c3, 0x96c8, 0x96ca,
02698 0x96cb, 0x96cd, 0x96d1, 0x96d3, 0x96d4, 0x96d6, 0x96d7, 0x96d8,
02699 0x96d9, 0x96da, 0x96db, 0x96dc, 0x96dd, 0x96de, 0x96df, 0x96e1,
02700 0x96e2, 0x96e3, 0x96e4, 0x96e5, 0x96e6, 0x96e7, 0x96e8, 0x96ec,
02701 0x96ed, 0x96ee, 0x96ef, 0x96f1, 0x96f2, 0x96f4, 0x96f5, 0x96f8,
02702 0x96fa, 0x96fb, 0x96fc, 0x96fd, 0x96ff, 0x9702, 0x9703, 0x9705,

```



```
02703 0x970a, 0x970b, 0x970c, 0x9710, 0x9711, 0x9712, 0x9714, 0x9715,
02704 0x9717, 0x9718, 0x9719, 0x971a, 0x971b, 0x971d, 0x971f, 0x9720,
02705 0x643f, 0x64d8, 0x8004, 0x6bea, 0x6bf3, 0x6bfd, 0x6bf5, 0x6bf9,
02706 0x6c05, 0x6c07, 0x6c06, 0x6c0d, 0x6c15, 0x6c18, 0x6c19, 0x6c1a,
02707 0x6c21, 0x6c29, 0x6c24, 0x6c2a, 0x6c32, 0x6535, 0x6555, 0x656b,
02708 0x724d, 0x7252, 0x7256, 0x7230, 0x8662, 0x5216, 0x809f, 0x809c,
02709 0x8093, 0x80bc, 0x670a, 0x80bd, 0x80b1, 0x80ab, 0x80ad, 0x80b4,
02710 0x80b7, 0x80e7, 0x80e8, 0x80e9, 0x80ea, 0x80db, 0x80c2, 0x80c4,
02711 0x80d9, 0x80cd, 0x80d7, 0x6710, 0x80dd, 0x80eb, 0x80f1, 0x80f4,
02712 0x80ed, 0x810d, 0x810e, 0x80f2, 0x80fc, 0x6715, 0x8112, 0x8c5a,
02713 0x8136, 0x811e, 0x812c, 0x8118, 0x8132, 0x8148, 0x814c, 0x8153,
02714 0x8174, 0x8159, 0x815a, 0x8171, 0x8160, 0x8169, 0x817c, 0x817d,
02715 0x816d, 0x8167, 0x584d, 0x5ab5, 0x8188, 0x8182, 0x8191, 0x6ed5,
02716 0x81a3, 0x81aa, 0x81cc, 0x6726, 0x81ca, 0x81bb,
02717 /* 0xec */
02718 0x9721, 0x9722, 0x9723, 0x9724, 0x9725, 0x9726, 0x9727, 0x9728,
02719 0x9729, 0x972b, 0x972c, 0x972e, 0x972f, 0x9731, 0x9733, 0x9734,
02720 0x9735, 0x9736, 0x9737, 0x973a, 0x973b, 0x973c, 0x973d, 0x973f,
02721 0x9740, 0x9741, 0x9742, 0x9743, 0x9744, 0x9745, 0x9746, 0x9747,
02722 0x9748, 0x9749, 0x974a, 0x974b, 0x974c, 0x974d, 0x974e, 0x974f,
02723 0x9750, 0x9751, 0x9754, 0x9755, 0x9757, 0x9758, 0x975a, 0x975c,
02724 0x975d, 0x975f, 0x9763, 0x9764, 0x9766, 0x9767, 0x9768, 0x976a,
02725 0x976b, 0x976c, 0x976d, 0x976e, 0x976f, 0x9770, 0x9771, 0x9772,
02726 0x9775, 0x9777, 0x9778, 0x9779, 0x977a, 0x977b, 0x977d, 0x977e,
02727 0x977f, 0x9780, 0x9781, 0x9782, 0x9783, 0x9784, 0x9786, 0x9787,
02728 0x9788, 0x9789, 0x978a, 0x978c, 0x978e, 0x978f, 0x9790, 0x9793,
02729 0x9795, 0x9796, 0x9797, 0x9799, 0x979a, 0x979b, 0x979c, 0x979d,
02730 0x81c1, 0x81a6, 0x6b24, 0x6b37, 0x6b39, 0x6b43, 0x6b46, 0x6b59,
02731 0x98d1, 0x98d2, 0x98d3, 0x98d5, 0x98d9, 0x98da, 0x6bb3, 0x5f40,
02732 0x6bc2, 0x89f3, 0x6590, 0x9f51, 0x6593, 0x65bc, 0x65c6, 0x65c4,
02733 0x65c3, 0x65cc, 0x65ce, 0x65d2, 0x65d6, 0x7080, 0x709c, 0x7096,
02734 0x709d, 0x70bb, 0x70c0, 0x70b7, 0x70ab, 0x70b1, 0x70e8, 0x70ca,
02735 0x7110, 0x7113, 0x7116, 0x712f, 0x7131, 0x7173, 0x715c, 0x7168,
02736 0x7145, 0x7172, 0x714a, 0x7178, 0x717a, 0x7198, 0x71b3, 0x71b5,
02737 0x71a8, 0x71a0, 0x71e0, 0x71d4, 0x71e7, 0x71f9, 0x721d, 0x7228,
02738 0x706c, 0x7118, 0x7166, 0x71b9, 0x623e, 0x623d, 0x6243, 0x6248,
02739 0x6249, 0x793b, 0x7940, 0x7946, 0x7949, 0x795b, 0x795c, 0x7953,
02740 0x795a, 0x7962, 0x7957, 0x7960, 0x796f, 0x7967, 0x797a, 0x7985,
02741 0x798a, 0x799a, 0x79a7, 0x79b3, 0x5fd1, 0x5fd0,
02742 /* 0xed */
02743 0x979e, 0x979f, 0x97a1, 0x97a2, 0x97a4, 0x97a5, 0x97a6, 0x97a7,
02744 0x97a8, 0x97a9, 0x97aa, 0x97ac, 0x97ae, 0x97b0, 0x97b1, 0x97b3,
02745 0x97b5, 0x97b6, 0x97b7, 0x97b8, 0x97b9, 0x97ba, 0x97bb, 0x97bc,
02746 0x97bd, 0x97be, 0x97bf, 0x97c0, 0x97c1, 0x97c2, 0x97c3, 0x97c4,
02747 0x97c5, 0x97c6, 0x97c7, 0x97c8, 0x97c9, 0x97ca, 0x97cb, 0x97cc,
02748 0x97cd, 0x97ce, 0x97cf, 0x97d0, 0x97d1, 0x97d2, 0x97d3, 0x97d4,
02749 0x97d5, 0x97d6, 0x97d7, 0x97d8, 0x97d9, 0x97da, 0x97db, 0x97dc,
02750 0x97dd, 0x97de, 0x97df, 0x97e0, 0x97e1, 0x97e2, 0x97e3, 0x97e4,
02751 0x97e5, 0x97e8, 0x97ee, 0x97ef, 0x97f0, 0x97f1, 0x97f2, 0x97f4,
02752 0x97f7, 0x97f8, 0x97f9, 0x97fa, 0x97fb, 0x97fc, 0x97fd, 0x97fe,
02753 0x97ff, 0x9800, 0x9801, 0x9802, 0x9803, 0x9804, 0x9805, 0x9806,
02754 0x9807, 0x9808, 0x9809, 0x980a, 0x980b, 0x980c, 0x980d, 0x980e,
02755 0x603c, 0x605d, 0x605a, 0x6067, 0x6041, 0x6059, 0x6063, 0x60ab,
02756 0x6106, 0x610d, 0x615d, 0x61a9, 0x619d, 0x61cb, 0x61d1, 0x6206,
02757 0x8080, 0x807f, 0x6c93, 0x6cf6, 0x6dfc, 0x77f6, 0x77f8, 0x7800,
02758 0x7809, 0x7817, 0x7818, 0x7811, 0x65ab, 0x782d, 0x781c, 0x781d,
02759 0x7839, 0x783a, 0x783b, 0x781f, 0x783c, 0x7825, 0x782c, 0x7823,
02760 0x7829, 0x784e, 0x786d, 0x7856, 0x7857, 0x7826, 0x7850, 0x7847,
02761 0x784c, 0x786a, 0x789b, 0x7893, 0x789a, 0x7887, 0x789c, 0x78a1,
02762 0x78a3, 0x78b2, 0x78b9, 0x78a5, 0x78d4, 0x78d9, 0x78c9, 0x78ec,
02763 0x78f2, 0x7905, 0x78f4, 0x7913, 0x7924, 0x791e, 0x7934, 0x9f9b,
02764 0x9ef9, 0x9efb, 0x9efc, 0x76f1, 0x7704, 0x770d, 0x76f9, 0x7707,
02765 0x7708, 0x771a, 0x7722, 0x7719, 0x772d, 0x7726, 0x7735, 0x7738,
02766 0x7750, 0x7751, 0x7747, 0x7743, 0x775a, 0x7768,
02767 /* 0xee */
02768 0x980f, 0x9810, 0x9811, 0x9812, 0x9813, 0x9814, 0x9815, 0x9816,
02769 0x9817, 0x9818, 0x9819, 0x981a, 0x981b, 0x981c, 0x981d, 0x981e,
02770 0x981f, 0x9820, 0x9821, 0x9822, 0x9823, 0x9824, 0x9825, 0x9826,
02771 0x9827, 0x9828, 0x9829, 0x982a, 0x982b, 0x982c, 0x982d, 0x982e,
02772 0x982f, 0x9830, 0x9831, 0x9832, 0x9833, 0x9834, 0x9835, 0x9836,
02773 0x9837, 0x9838, 0x9839, 0x983a, 0x983b, 0x983c, 0x983d, 0x983e,
02774 0x983f, 0x9840, 0x9841, 0x9842, 0x9843, 0x9844, 0x9845, 0x9846,
02775 0x9847, 0x9848, 0x9849, 0x984a, 0x984b, 0x984c, 0x984d, 0x984e,
02776 0x984f, 0x9850, 0x9851, 0x9852, 0x9853, 0x9854, 0x9855, 0x9856,
02777 0x9857, 0x9858, 0x9859, 0x985a, 0x985b, 0x985c, 0x985d, 0x985e,
02778 0x985f, 0x9860, 0x9861, 0x9862, 0x9863, 0x9864, 0x9865, 0x9866,
02779 0x9867, 0x9868, 0x9869, 0x986a, 0x986b, 0x986c, 0x986d, 0x986e,
02780 0x7762, 0x7765, 0x777f, 0x778d, 0x777d, 0x7780, 0x778c, 0x7791,
02781 0x779f, 0x77a0, 0x77b0, 0x77b5, 0x77bd, 0x753a, 0x7540, 0x754e,
02782 0x754b, 0x7548, 0x755b, 0x7572, 0x7579, 0x7583, 0x7f58, 0x7f61,
02783 0x7f5f, 0x8a48, 0x7f68, 0x7f74, 0x7f71, 0x7f79, 0x7f81, 0x7f7e,
02784 0x76cd, 0x76e5, 0x8832, 0x9485, 0x9486, 0x9487, 0x948b, 0x948a,
02785 0x948c, 0x948d, 0x948f, 0x9490, 0x949a, 0x9497, 0x9495, 0x949a,
02786 0x949b, 0x949c, 0x94a3, 0x94a4, 0x94ab, 0x94aa, 0x94ad, 0x94ac,
02787 0x94af, 0x94b0, 0x94b2, 0x94b4, 0x94b6, 0x94b7, 0x94b8, 0x94b9,
02788 0x94ba, 0x94bc, 0x94bd, 0x94bf, 0x94c4, 0x94c8, 0x94c9, 0x94ca,
02789 0x94cb, 0x94cc, 0x94cd, 0x94ce, 0x94d0, 0x94d1, 0x94d2, 0x94d5,
```

```

02790 0x94d6, 0x94d7, 0x94d9, 0x94d8, 0x94db, 0x94de, 0x94df, 0x94e0,
02791 0x94e2, 0x94e4, 0x94e5, 0x94e7, 0x94e8, 0x94ea,
02792 /* 0xef */
02793 0x986f, 0x9870, 0x9871, 0x9872, 0x9873, 0x9874, 0x988b, 0x988e,
02794 0x9892, 0x9895, 0x9899, 0x98a3, 0x98a8, 0x98a9, 0x98aa, 0x98ab,
02795 0x98ac, 0x98ad, 0x98ae, 0x98af, 0x98b0, 0x98b1, 0x98b2, 0x98b3,
02796 0x98b4, 0x98b5, 0x98b6, 0x98b7, 0x98b8, 0x98b9, 0x98ba, 0x98bb,
02797 0x98bc, 0x98bd, 0x98be, 0x98bf, 0x98c0, 0x98c1, 0x98c2, 0x98c3,
02798 0x98c4, 0x98c5, 0x98c6, 0x98c7, 0x98c8, 0x98c9, 0x98ca, 0x98cb,
02799 0x98cc, 0x98cd, 0x98cf, 0x98d0, 0x98d4, 0x98d6, 0x98d7, 0x98db,
02800 0x98dc, 0x98dd, 0x98e0, 0x98e1, 0x98e2, 0x98e3, 0x98e4, 0x98e5,
02801 0x98e6, 0x98e9, 0x98ea, 0x98eb, 0x98ec, 0x98ed, 0x98ee, 0x98ef,
02802 0x98f0, 0x98f1, 0x98f2, 0x98f3, 0x98f4, 0x98f5, 0x98f6, 0x98f7,
02803 0x98f8, 0x98f9, 0x98fa, 0x98fb, 0x98fc, 0x98fd, 0x98fe, 0x98ff,
02804 0x9900, 0x9901, 0x9902, 0x9903, 0x9904, 0x9905, 0x9906, 0x9907,
02805 0x94e9, 0x94eb, 0x94ee, 0x94ef, 0x94f3, 0x94f4, 0x94f5, 0x94f7,
02806 0x94f9, 0x94fc, 0x94fd, 0x94ff, 0x9503, 0x9502, 0x9506, 0x9507,
02807 0x9509, 0x950a, 0x950d, 0x950e, 0x950f, 0x9512, 0x9513, 0x9514,
02808 0x9515, 0x9516, 0x9518, 0x951b, 0x951d, 0x951e, 0x951f, 0x9522,
02809 0x952a, 0x952b, 0x9529, 0x952c, 0x9531, 0x9532, 0x9534, 0x9536,
02810 0x9537, 0x9538, 0x953c, 0x953e, 0x953f, 0x9542, 0x9535, 0x9544,
02811 0x9545, 0x9546, 0x9549, 0x954c, 0x954e, 0x954f, 0x9552, 0x9553,
02812 0x9554, 0x9556, 0x9557, 0x9558, 0x9559, 0x955b, 0x955e, 0x955f,
02813 0x955d, 0x9561, 0x9562, 0x9564, 0x9565, 0x9566, 0x9567, 0x9568,
02814 0x9569, 0x956a, 0x956b, 0x956c, 0x956f, 0x9571, 0x9572, 0x9573,
02815 0x953a, 0x77e7, 0x77ec, 0x96c9, 0x79d5, 0x79ed, 0x79e3, 0x79eb,
02816 0x7a06, 0x5d47, 0x7a03, 0x7a02, 0x7a1e, 0x7a14,
02817 /* 0xf0 */
02818 0x9908, 0x9909, 0x990a, 0x990b, 0x990c, 0x990e, 0x990f, 0x9911,
02819 0x9912, 0x9913, 0x9914, 0x9915, 0x9916, 0x9917, 0x9918, 0x9919,
02820 0x991a, 0x991b, 0x991c, 0x991d, 0x991e, 0x991f, 0x9920, 0x9921,
02821 0x9922, 0x9923, 0x9924, 0x9925, 0x9926, 0x9927, 0x9928, 0x9929,
02822 0x992a, 0x992b, 0x992c, 0x992d, 0x992f, 0x9930, 0x9931, 0x9932,
02823 0x9933, 0x9934, 0x9935, 0x9936, 0x9937, 0x9938, 0x9939, 0x993a,
02824 0x993b, 0x993c, 0x993d, 0x993e, 0x993f, 0x9940, 0x9941, 0x9942,
02825 0x9943, 0x9944, 0x9945, 0x9946, 0x9947, 0x9948, 0x9949, 0x994a,
02826 0x994b, 0x994c, 0x994d, 0x994e, 0x994f, 0x9950, 0x9951, 0x9952,
02827 0x9953, 0x9956, 0x9957, 0x9958, 0x9959, 0x995a, 0x995b, 0x995c,
02828 0x995d, 0x995e, 0x995f, 0x9960, 0x9961, 0x9962, 0x9964, 0x9966,
02829 0x9973, 0x9978, 0x9979, 0x997b, 0x997e, 0x9982, 0x9983, 0x9989,
02830 0x7a39, 0x7a37, 0x7a51, 0x9ecf, 0x99a5, 0x7a70, 0x7688, 0x768e,
02831 0x7693, 0x7699, 0x76a4, 0x74de, 0x74e0, 0x752c, 0x9e20, 0x9e22,
02832 0x9e28, 0x9e29, 0x9e2a, 0x9e2b, 0x9e2c, 0x9e32, 0x9e31, 0x9e36,
02833 0x9e38, 0x9e37, 0x9e39, 0x9e3a, 0x9e3e, 0x9e41, 0x9e42, 0x9e44,
02834 0x9e46, 0x9e47, 0x9e48, 0x9e49, 0x9e4b, 0x9e4c, 0x9e4e, 0x9e51,
02835 0x9e55, 0x9e57, 0x9e5a, 0x9e5b, 0x9e5c, 0x9e5e, 0x9e63, 0x9e66,
02836 0x9e67, 0x9e68, 0x9e69, 0x9e6a, 0x9e6b, 0x9e6c, 0x9e71, 0x9e6d,
02837 0x9e73, 0x7592, 0x7594, 0x7596, 0x75a0, 0x759d, 0x75ac, 0x75a3,
02838 0x75b3, 0x75b4, 0x75b8, 0x75c4, 0x75b1, 0x75b0, 0x75c3, 0x75c2,
02839 0x75d6, 0x75cd, 0x75e3, 0x75e8, 0x75e6, 0x75e4, 0x75eb, 0x75e7,
02840 0x7603, 0x75f1, 0x75fc, 0x75ff, 0x7610, 0x7600, 0x7605, 0x760c,
02841 0x7617, 0x760a, 0x7625, 0x7618, 0x7615, 0x7619,
02842 /* 0xf1 */
02843 0x998c, 0x998e, 0x999a, 0x999b, 0x999c, 0x999d, 0x999e, 0x999f,
02844 0x99a0, 0x99a1, 0x99a2, 0x99a3, 0x99a4, 0x99a6, 0x99a7, 0x99a9,
02845 0x99aa, 0x99ab, 0x99ac, 0x99ad, 0x99ae, 0x99af, 0x99b0, 0x99b1,
02846 0x99b2, 0x99b3, 0x99b4, 0x99b5, 0x99b6, 0x99b7, 0x99b8, 0x99b9,
02847 0x99ba, 0x99bb, 0x99bc, 0x99bd, 0x99be, 0x99bf, 0x99c0, 0x99c1,
02848 0x99c2, 0x99c3, 0x99c4, 0x99c5, 0x99c6, 0x99c7, 0x99c8, 0x99c9,
02849 0x99ca, 0x99cb, 0x99cc, 0x99cd, 0x99ce, 0x99cf, 0x99d0, 0x99d1,
02850 0x99d2, 0x99d3, 0x99d4, 0x99d5, 0x99d6, 0x99d7, 0x99d8, 0x99d9,
02851 0x99da, 0x99db, 0x99dc, 0x99dd, 0x99de, 0x99df, 0x99e0, 0x99e1,
02852 0x99e2, 0x99e3, 0x99e4, 0x99e5, 0x99e6, 0x99e7, 0x99e8, 0x99e9,
02853 0x99ea, 0x99eb, 0x99ec, 0x99ed, 0x99ee, 0x99ef, 0x99f0, 0x99f1,
02854 0x99f2, 0x99f3, 0x99f4, 0x99f5, 0x99f6, 0x99f7, 0x99f8, 0x99f9,
02855 0x761b, 0x763c, 0x7622, 0x7620, 0x7640, 0x762d, 0x7630, 0x763f,
02856 0x7635, 0x7643, 0x763e, 0x7633, 0x764d, 0x765e, 0x7654, 0x765c,
02857 0x7656, 0x766b, 0x766f, 0x7fca, 0x7ae6, 0x7a78, 0x7a79, 0x7a80,
02858 0x7a86, 0x7a88, 0x7a95, 0x7aa6, 0x7aa0, 0x7aac, 0x7aa8, 0x7aad,
02859 0x7ab3, 0x8864, 0x8869, 0x8872, 0x887d, 0x887f, 0x8882, 0x88a2,
02860 0x88c6, 0x88b7, 0x88bc, 0x88c9, 0x88e2, 0x88ce, 0x88e3, 0x88e5,
02861 0x88f1, 0x891a, 0x88fc, 0x88e8, 0x88fe, 0x88f0, 0x8921, 0x8919,
02862 0x8913, 0x891b, 0x890a, 0x8934, 0x892b, 0x8936, 0x8941, 0x8966,
02863 0x897b, 0x758b, 0x80e5, 0x76b2, 0x76b4, 0x77dc, 0x8012, 0x8014,
02864 0x8016, 0x801c, 0x8020, 0x8022, 0x8025, 0x8026, 0x8027, 0x8029,
02865 0x8028, 0x8031, 0x800b, 0x8035, 0x8043, 0x8046, 0x804d, 0x8052,
02866 0x8069, 0x8071, 0x8983, 0x9878, 0x9880, 0x9883,
02867 /* 0xf2 */
02868 0x99fa, 0x99fb, 0x99fc, 0x99fd, 0x99fe, 0x99ff, 0x9a00, 0x9a01,
02869 0x9a02, 0x9a03, 0x9a04, 0x9a05, 0x9a06, 0x9a07, 0x9a08, 0x9a09,
02870 0x9a0a, 0x9a0b, 0x9a0c, 0x9a0d, 0x9a0e, 0x9a0f, 0x9a10, 0x9a11,
02871 0x9a12, 0x9a13, 0x9a14, 0x9a15, 0x9a16, 0x9a17, 0x9a18, 0x9a19,
02872 0x9a1a, 0x9a1b, 0x9a1c, 0x9a1d, 0x9a1e, 0x9a1f, 0x9a20, 0x9a21,
02873 0x9a22, 0x9a23, 0x9a24, 0x9a25, 0x9a26, 0x9a27, 0x9a28, 0x9a29,
02874 0x9a2a, 0x9a2b, 0x9a2c, 0x9a2d, 0x9a2e, 0x9a2f, 0x9a30, 0x9a31,
02875 0x9a32, 0x9a33, 0x9a34, 0x9a35, 0x9a36, 0x9a37, 0x9a38, 0x9a39,
02876 0x9a3a, 0x9a3b, 0x9a3c, 0x9a3d, 0x9a3e, 0x9a3f, 0x9a40, 0x9a41,

```



```
02877 0x9a42, 0x9a43, 0x9a44, 0x9a45, 0x9a46, 0x9a47, 0x9a48, 0x9a49,
02878 0x9a4a, 0x9a4b, 0x9a4c, 0x9a4d, 0x9a4e, 0x9a4f, 0x9a50, 0x9a51,
02879 0x9a52, 0x9a53, 0x9a54, 0x9a55, 0x9a56, 0x9a57, 0x9a58, 0x9a59,
02880 0x9889, 0x988c, 0x988d, 0x988f, 0x9894, 0x989a, 0x989b, 0x989e,
02881 0x989f, 0x98a1, 0x98a2, 0x98a5, 0x98a6, 0x864d, 0x8654, 0x866c,
02882 0x866e, 0x867f, 0x867a, 0x867c, 0x867b, 0x86a8, 0x868d, 0x868b,
02883 0x86ac, 0x869d, 0x86a7, 0x86a3, 0x86aa, 0x8693, 0x86a9, 0x86b6,
02884 0x86c4, 0x86b5, 0x86ce, 0x86b0, 0x86ba, 0x86b1, 0x86af, 0x86c9,
02885 0x86cf, 0x86b4, 0x86e9, 0x86f1, 0x86f2, 0x86ed, 0x86f3, 0x86d0,
02886 0x8713, 0x86de, 0x86f4, 0x86df, 0x86d8, 0x86d1, 0x8703, 0x8707,
02887 0x86f8, 0x8708, 0x870a, 0x870d, 0x8709, 0x8723, 0x873b, 0x871e,
02888 0x8725, 0x872e, 0x871a, 0x873e, 0x8748, 0x8734, 0x8731, 0x8729,
02889 0x8737, 0x873f, 0x8782, 0x8722, 0x877d, 0x877e, 0x877b, 0x8760,
02890 0x8770, 0x874c, 0x876e, 0x878b, 0x8753, 0x8763, 0x877c, 0x8764,
02891 0x8759, 0x8765, 0x8793, 0x87af, 0x87a8, 0x87d2,
02892 /* 0xf3 */
02893 0x9a5a, 0x9a5b, 0x9a5c, 0x9a5d, 0x9a5e, 0x9a5f, 0x9a60, 0x9a61,
02894 0x9a62, 0x9a63, 0x9a64, 0x9a65, 0x9a66, 0x9a67, 0x9a68, 0x9a69,
02895 0x9a6a, 0x9a6b, 0x9a72, 0x9a83, 0x9a89, 0x9a8d, 0x9a8e, 0x9a94,
02896 0x9a95, 0x9a99, 0x9aa6, 0x9aa9, 0x9aaa, 0x9aab, 0x9aac, 0x9aad,
02897 0x9aae, 0x9aaf, 0x9ab2, 0x9ab3, 0x9ab4, 0x9ab5, 0x9abb,
02898 0x9abd, 0x9abe, 0x9abf, 0x9ac3, 0x9ac4, 0x9ac6, 0x9ac7, 0x9ac8,
02899 0x9ac9, 0x9aca, 0x9acd, 0x9ace, 0x9acf, 0x9ad0, 0x9ad2, 0x9ad4,
02900 0x9ad5, 0x9ad6, 0x9ad7, 0x9ad9, 0x9ada, 0x9adb, 0x9adc, 0x9add,
02901 0x9ade, 0x9ae0, 0x9ae2, 0x9ae3, 0x9ae4, 0x9ae5, 0x9ae7, 0x9ae8,
02902 0x9ae9, 0x9aea, 0x9aec, 0x9aee, 0x9af0, 0x9af1, 0x9af2, 0x9af3,
02903 0x9af4, 0x9af5, 0x9af6, 0x9af7, 0x9af8, 0x9afa, 0x9afc, 0x9afd,
02904 0x9afe, 0x9aff, 0x9b00, 0x9b01, 0x9b02, 0x9b04, 0x9b05, 0x9b06,
02905 0x87c6, 0x8788, 0x8785, 0x87ad, 0x8797, 0x8783, 0x87ab, 0x87e5,
02906 0x87ac, 0x87b5, 0x87b3, 0x87b3, 0x87cb, 0x87d3, 0x87bd, 0x87d1, 0x87c0,
02907 0x87ca, 0x87db, 0x87ea, 0x87e0, 0x87ee, 0x8816, 0x8813, 0x87fe,
02908 0x880a, 0x881b, 0x8821, 0x8839, 0x883c, 0x7f36, 0x7f42, 0x7f44,
02909 0x7f45, 0x8210, 0x7afa, 0x7afd, 0x7b08, 0x7b03, 0x7b04, 0x7b15,
02910 0x7b0a, 0x7b2b, 0x7b0f, 0x7b47, 0x7b38, 0x7b2a, 0x7b19, 0x7b2e,
02911 0x7b31, 0x7b20, 0x7b25, 0x7b24, 0x7b33, 0x7b3e, 0x7b1e, 0x7b58,
02912 0x7b5a, 0x7b45, 0x7b75, 0x7b4c, 0x7b5d, 0x7b60, 0x7b6e, 0x7b7b,
02913 0x7b62, 0x7b72, 0x7b71, 0x7b90, 0x7ba6, 0x7ba7, 0x7bb8, 0x7bac,
02914 0x7b9d, 0x7ba8, 0x7b85, 0x7baa, 0x7b9c, 0x7ba2, 0x7bab, 0x7bb4,
02915 0x7bd1, 0x7bcl, 0x7bcc, 0x7bdd, 0x7bda, 0x7be5, 0x7be6, 0x7bea,
02916 0x7c0c, 0x7bfe, 0x7bfc, 0x7c0f, 0x7c16, 0x7c0b,
02917 /* 0xf4 */
02918 0x9b07, 0x9b09, 0x9b0a, 0x9b0b, 0x9b0c, 0x9b0d, 0x9b0e, 0x9b10,
02919 0x9b11, 0x9b12, 0x9b14, 0x9b15, 0x9b16, 0x9b17, 0x9b18, 0x9b19,
02920 0x9b1a, 0x9b1b, 0x9b1c, 0x9b1d, 0x9b1e, 0x9b20, 0x9b21, 0x9b22,
02921 0x9b24, 0x9b25, 0x9b26, 0x9b27, 0x9b28, 0x9b29, 0x9b2a, 0x9b2b,
02922 0x9b2c, 0x9b2d, 0x9b2e, 0x9b30, 0x9b31, 0x9b33, 0x9b34, 0x9b35,
02923 0x9b36, 0x9b37, 0x9b38, 0x9b39, 0x9b3a, 0x9b3d, 0x9b3e, 0x9b3f,
02924 0x9b40, 0x9b46, 0x9b4a, 0x9b4b, 0x9b4c, 0x9b4e, 0x9b50, 0x9b52,
02925 0x9b53, 0x9b55, 0x9b56, 0x9b57, 0x9b58, 0x9b59, 0x9b5a, 0x9b5b,
02926 0x9b5c, 0x9b5d, 0x9b5e, 0x9b5f, 0x9b60, 0x9b61, 0x9b62, 0x9b63,
02927 0x9b64, 0x9b65, 0x9b66, 0x9b67, 0x9b68, 0x9b69, 0x9b6a, 0x9b6b,
02928 0x9b6c, 0x9b6d, 0x9b6e, 0x9b6f, 0x9b70, 0x9b71, 0x9b72, 0x9b73,
02929 0x9b74, 0x9b75, 0x9b76, 0x9b77, 0x9b78, 0x9b79, 0x9b7a, 0x9b7b,
02930 0x7c1f, 0x7c2a, 0x7c26, 0x7c38, 0x7c41, 0x7c40, 0x81fe, 0x8201,
02931 0x8202, 0x8204, 0x81ec, 0x8844, 0x8221, 0x8222, 0x8223, 0x822d,
02932 0x822f, 0x8228, 0x822b, 0x8238, 0x823b, 0x8233, 0x8234, 0x823e,
02933 0x8244, 0x8249, 0x824b, 0x824f, 0x825a, 0x825f, 0x8268, 0x887e,
02934 0x8885, 0x8888, 0x88d8, 0x88df, 0x895e, 0x7f9d, 0x7f9f, 0x7fa7,
02935 0x7faf, 0x7fb0, 0x7fb2, 0x7c7c, 0x6549, 0x7c91, 0x7c9d, 0x7c9c,
02936 0x7c9e, 0x7ca2, 0x7cb2, 0x7cb2, 0x7cb2, 0x7cc1, 0x7cc7, 0x7ccc,
02937 0x7ccd, 0x7cc8, 0x7cc5, 0x7cd7, 0x7ce8, 0x826e, 0x66a8, 0x7fbf,
02938 0x7fce, 0x7fd5, 0x7fe5, 0x7fe1, 0x7fe6, 0x7fe9, 0x7fee, 0x7ff3,
02939 0x7cf8, 0x7d77, 0x7da6, 0x7dae, 0x7e47, 0x7e9b, 0x9eb8, 0x9eb4,
02940 0x8d73, 0x8d84, 0x8d94, 0x8db1, 0x8d67, 0x8d6d, 0x8c47,
02941 0x8c49, 0x914a, 0x9150, 0x914e, 0x914f, 0x9164,
02942 /* 0xf5 */
02943 0x9b7c, 0x9b7d, 0x9b7e, 0x9b7f, 0x9b80, 0x9b81, 0x9b82, 0x9b83,
02944 0x9b84, 0x9b85, 0x9b86, 0x9b87, 0x9b88, 0x9b89, 0x9b8a, 0x9b8b,
02945 0x9b8c, 0x9b8d, 0x9b8e, 0x9b8f, 0x9b90, 0x9b91, 0x9b92, 0x9b93,
02946 0x9b94, 0x9b95, 0x9b96, 0x9b97, 0x9b98, 0x9b99, 0x9b9a, 0x9b9b,
02947 0x9b9c, 0x9b9d, 0x9b9e, 0x9b9f, 0x9ba0, 0x9ba1, 0x9ba2, 0x9ba3,
02948 0x9ba4, 0x9ba5, 0x9ba6, 0x9ba7, 0x9ba8, 0x9ba9, 0x9baa, 0x9bab,
02949 0x9bac, 0x9bad, 0x9bae, 0x9baf, 0x9bb0, 0x9bb1, 0x9bb2, 0x9bb3,
02950 0x9bb4, 0x9bb5, 0x9bb6, 0x9bb7, 0x9bb8, 0x9bb9, 0x9bba, 0x9bbb,
02951 0x9bbc, 0x9bbd, 0x9bbe, 0x9bbf, 0x9bc0, 0x9bc1, 0x9bc2, 0x9bc3,
02952 0x9bc4, 0x9bc5, 0x9bc6, 0x9bc7, 0x9bc8, 0x9bc9, 0x9bca, 0x9bcb,
02953 0x9bcc, 0x9bcd, 0x9bce, 0x9bcf, 0x9bd0, 0x9bd1, 0x9bd2, 0x9bd3,
02954 0x9bd4, 0x9bd5, 0x9bd6, 0x9bd7, 0x9bd8, 0x9bd9, 0x9bda, 0x9bdb,
02955 0x9162, 0x9161, 0x9170, 0x9169, 0x916f, 0x917d, 0x917e, 0x9172,
02956 0x9174, 0x9179, 0x918c, 0x9185, 0x9190, 0x918d, 0x9191, 0x91a2,
02957 0x91a3, 0x91aa, 0x91ad, 0x91ae, 0x91af, 0x91b5, 0x91b4, 0x91ba,
02958 0x8c55, 0x9e7e, 0x8db8, 0x8deb, 0x8e05, 0x8e59, 0x8e69, 0x8db5,
02959 0x8dbf, 0x8dbc, 0x8dba, 0x8dc4, 0x8dd6, 0x8dd7, 0x8dda, 0x8dde,
02960 0x8dce, 0x8dcf, 0x8ddb, 0x8dc6, 0x8dec, 0x8df7, 0x8df8, 0x8de3,
02961 0x8df9, 0x8dfb, 0x8de4, 0x8e09, 0x8dfd, 0x8e14, 0x8e1d, 0x8e1f,
02962 0x8e2c, 0x8e2e, 0x8e23, 0x8e2f, 0x8e3a, 0x8e40, 0x8e39, 0x8e35,
02963 0x8e3d, 0x8e31, 0x8e49, 0x8e41, 0x8e42, 0x8e51, 0x8e52, 0x8e4a,
```

```
02964 0x8e70, 0x8e76, 0x8e7c, 0x8e6f, 0x8e74, 0x8e85, 0x8e8f, 0x8e94,
02965 0x8e90, 0x8e9c, 0x8e9e, 0x8c78, 0x8c82, 0x8c8a, 0x8c85, 0x8c98,
02966 0x8c94, 0x659b, 0x89d6, 0x89de, 0x89da, 0x89dc,
02967 /* 0xf6 */
02968 0x9bdc, 0x9bdd, 0x9bde, 0x9bdf, 0x9be0, 0x9be1, 0x9be2, 0x9be3,
02969 0x9be4, 0x9be5, 0x9be6, 0x9be7, 0x9be8, 0x9be9, 0x9bea, 0x9beb,
02970 0x9bec, 0x9bed, 0x9bee, 0x9bef, 0x9bf0, 0x9bf1, 0x9bf2, 0x9bf3,
02971 0x9bf4, 0x9bf5, 0x9bf6, 0x9bf7, 0x9bf8, 0x9bf9, 0x9bfa, 0x9bfb,
02972 0x9bfc, 0x9bfd, 0x9bfe, 0x9bff, 0x9c00, 0x9c01, 0x9c02, 0x9c03,
02973 0x9c04, 0x9c05, 0x9c06, 0x9c07, 0x9c08, 0x9c09, 0x9c0a, 0x9c0b,
02974 0x9c0c, 0x9c0d, 0x9c0e, 0x9c0f, 0x9c10, 0x9c11, 0x9c12, 0x9c13,
02975 0x9c14, 0x9c15, 0x9c16, 0x9c17, 0x9c18, 0x9c19, 0x9c1a, 0x9c1b,
02976 0x9c1c, 0x9c1d, 0x9c1e, 0x9c1f, 0x9c20, 0x9c21, 0x9c22, 0x9c23,
02977 0x9c24, 0x9c25, 0x9c26, 0x9c27, 0x9c28, 0x9c29, 0x9c2a, 0x9c2b,
02978 0x9c2c, 0x9c2d, 0x9c2e, 0x9c2f, 0x9c30, 0x9c31, 0x9c32, 0x9c33,
02979 0x9c34, 0x9c35, 0x9c36, 0x9c37, 0x9c38, 0x9c39, 0x9c3a, 0x9c3b,
02980 0x89e5, 0x89eb, 0x89ef, 0x8a3e, 0x8b26, 0x9753, 0x96e9, 0x96f3,
02981 0x96ef, 0x9706, 0x970f, 0x9708, 0x970f, 0x970e, 0x972a, 0x972d,
02982 0x9730, 0x973e, 0x9f80, 0x9f83, 0x9f85, 0x9f86, 0x9f87, 0x9f88,
02983 0x9f89, 0x9f8a, 0x9f8c, 0x9efe, 0x9f0b, 0x9f0d, 0x96b9, 0x96bc,
02984 0x96bd, 0x96ce, 0x96d2, 0x77bf, 0x96e0, 0x928e, 0x92ae, 0x92c8,
02985 0x933e, 0x936a, 0x93ca, 0x938f, 0x943e, 0x946b, 0x9c7f, 0x9c82,
02986 0x9c85, 0x9c86, 0x9c87, 0x9c88, 0x7a23, 0x9c8b, 0x9c8e, 0x9c90,
02987 0x9c91, 0x9c92, 0x9c94, 0x9c95, 0x9c9a, 0x9c9b, 0x9c9e, 0x9c9f,
02988 0x9ca0, 0x9ca1, 0x9ca2, 0x9ca3, 0x9ca5, 0x9ca6, 0x9ca7, 0x9ca8,
02989 0x9ca9, 0x9cab, 0x9cad, 0x9cae, 0x9cb0, 0x9cb1, 0x9cb2, 0x9cb3,
02990 0x9cb4, 0x9cb5, 0x9cb6, 0x9cb7, 0x9cba, 0x9cbb, 0x9cbc, 0x9cbd,
02991 0x9cc4, 0x9cc5, 0x9cc6, 0x9cc7, 0x9cca, 0x9ccb,
02992 /* 0xf7 */
02993 0x9c3c, 0x9c3d, 0x9c3e, 0x9c3f, 0x9c40, 0x9c41, 0x9c42, 0x9c43,
02994 0x9c44, 0x9c45, 0x9c46, 0x9c47, 0x9c48, 0x9c49, 0x9c4a, 0x9c4b,
02995 0x9c4c, 0x9c4d, 0x9c4e, 0x9c4f, 0x9c50, 0x9c51, 0x9c52, 0x9c53,
02996 0x9c54, 0x9c55, 0x9c56, 0x9c57, 0x9c58, 0x9c59, 0x9c5a, 0x9c5b,
02997 0x9c5c, 0x9c5d, 0x9c5e, 0x9c5f, 0x9c60, 0x9c61, 0x9c62, 0x9c63,
02998 0x9c64, 0x9c65, 0x9c66, 0x9c67, 0x9c68, 0x9c69, 0x9c6a, 0x9c6b,
02999 0x9c6c, 0x9c6d, 0x9c6e, 0x9c6f, 0x9c70, 0x9c71, 0x9c72, 0x9c73,
03000 0x9c74, 0x9c75, 0x9c76, 0x9c77, 0x9c78, 0x9c79, 0x9c7a, 0x9c7b,
03001 0x9c7d, 0x9c7e, 0x9c80, 0x9c83, 0x9c84, 0x9c89, 0x9c8a, 0x9c8c,
03002 0x9c8f, 0x9c93, 0x9c96, 0x9c97, 0x9c99, 0x9c9d, 0x9caa,
03003 0x9cac, 0x9caf, 0x9cb9, 0x9cbe, 0x9cbf, 0x9cc0, 0x9cc1, 0x9cc2,
03004 0x9cc8, 0x9cc9, 0x9cd1, 0x9cd2, 0x9cda, 0x9cdb, 0x9ce0, 0x9ce1,
03005 0x9ccc, 0x9ccd, 0x9cce, 0x9ccf, 0x9cd0, 0x9cd3, 0x9cd4, 0x9cd5,
03006 0x9cd7, 0x9cd8, 0x9cd9, 0x9cdc, 0x9cdd, 0x9cdf, 0x9ce2, 0x977c,
03007 0x9785, 0x9791, 0x9792, 0x9794, 0x97af, 0x97ab, 0x97a3, 0x97b2,
03008 0x97b4, 0x97ab1, 0x97ab0, 0x97ab7, 0x97ab6, 0x97aba, 0x97abc,
03009 0x9ac1, 0x9ac0, 0x9ac5, 0x9ac2, 0x9acb, 0x9acc, 0x9ad1, 0x9b45,
03010 0x9b43, 0x9b47, 0x9b49, 0x9b48, 0x9b4d, 0x9b51, 0x98e8, 0x990d,
03011 0x992e, 0x9954, 0x995d, 0x995f, 0x9954, 0x995d, 0x995f, 0x9954,
03012 0x995f, 0x995d, 0x995f, 0x9954, 0x995d, 0x995f, 0x9954,
03013 0x995f, 0x995d, 0x995f, 0x9954, 0x995d, 0x995f, 0x9954,
03014 0x995f, 0x995d, 0x995f, 0x9954, 0x995d, 0x995f, 0x9954,
03015 0x995f, 0x995d, 0x995f, 0x9954, 0x995d, 0x995f, 0x9954,
03016 0x995f, 0x995d, 0x995f, 0x9954, 0x995d, 0x995f, 0x9954,
03017 /* 0xf8 */
03018 0x9ce3, 0x9ce4, 0x9ce5, 0x9ce6, 0x9ce7, 0x9ce8, 0x9ce9, 0x9cea,
03019 0x9ceb, 0x9cec, 0x9ced, 0x9cee, 0x9cef, 0x9cf0, 0x9cf1, 0x9cf2,
03020 0x9cf3, 0x9cf4, 0x9cf5, 0x9cf6, 0x9cf7, 0x9cf8, 0x9cf9, 0x9cfa,
03021 0x9cfb, 0x9cfc, 0x9cfd, 0x9cfe, 0x9cff, 0x9d00, 0x9d01, 0x9d02,
03022 0x9d03, 0x9d04, 0x9d05, 0x9d06, 0x9d07, 0x9d08, 0x9d09, 0x9d0a,
03023 0x9d0b, 0x9d0c, 0x9d0d, 0x9d0e, 0x9d0f, 0x9d10, 0x9d11, 0x9d12,
03024 0x9d13, 0x9d14, 0x9d15, 0x9d16, 0x9d17, 0x9d18, 0x9d19, 0x9d1a,
03025 0x9d1b, 0x9d1c, 0x9d1d, 0x9d1e, 0x9d1f, 0x9d20, 0x9d21, 0x9d22,
03026 0x9d23, 0x9d24, 0x9d25, 0x9d26, 0x9d27, 0x9d28, 0x9d29, 0x9d2a,
03027 0x9d2b, 0x9d2c, 0x9d2d, 0x9d2e, 0x9d2f, 0x9d30, 0x9d31, 0x9d32,
03028 0x9d33, 0x9d34, 0x9d35, 0x9d36, 0x9d37, 0x9d38, 0x9d39, 0x9d3a,
03029 0x9d3b, 0x9d3c, 0x9d3d, 0x9d3e, 0x9d3f, 0x9d40, 0x9d41, 0x9d42,
03030 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03031 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03032 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03033 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03034 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03035 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03036 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03037 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03038 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03039 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03040 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03041 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
03042 /* 0xf9 */
03043 0x9d43, 0x9d44, 0x9d45, 0x9d46, 0x9d47, 0x9d48, 0x9d49, 0x9d4a,
03044 0x9d4b, 0x9d4c, 0x9d4d, 0x9d4e, 0x9d4f, 0x9d50, 0x9d51, 0x9d52,
03045 0x9d53, 0x9d54, 0x9d55, 0x9d56, 0x9d57, 0x9d58, 0x9d59, 0x9d5a,
03046 0x9d5b, 0x9d5c, 0x9d5d, 0x9d5e, 0x9d5f, 0x9d60, 0x9d61, 0x9d62,
03047 0x9d63, 0x9d64, 0x9d65, 0x9d66, 0x9d67, 0x9d68, 0x9d69, 0x9d6a,
03048 0x9d6b, 0x9d6c, 0x9d6d, 0x9d6e, 0x9d6f, 0x9d70, 0x9d71, 0x9d72,
03049 0x9d73, 0x9d74, 0x9d75, 0x9d76, 0x9d77, 0x9d78, 0x9d79, 0x9d7a,
03050 0x9d7b, 0x9d7c, 0x9d7d, 0x9d7e, 0x9d7f, 0x9d80, 0x9d81, 0x9d82,
```

```
03051 0x9d83, 0x9d84, 0x9d85, 0x9d86, 0x9d87, 0x9d88, 0x9d89, 0x9d8a,
03052 0x9d8b, 0x9d8c, 0x9d8d, 0x9d8e, 0x9d8f, 0x9d90, 0x9d91, 0x9d92,
03053 0x9d93, 0x9d94, 0x9d95, 0x9d96, 0x9d97, 0x9d98, 0x9d99, 0x9da0,
03054 0x9da1, 0x9da2, 0x9da3, 0x9da4, 0x9da5, 0x9da6, 0x9da7, 0x9da8, 0x9da9, 0x9daa,
03055 0x9dab, 0x9dac, 0x9dad, 0x9dae, 0x9daf, 0x9db0, 0x9db1, 0x9db2,
03056 0x9db3, 0x9db4, 0x9db5, 0x9db6, 0x9db7, 0x9db8, 0x9db9, 0x9dba,
03057 0x9dbb, 0x9dbc, 0x9dbd, 0x9dbe, 0x9dbf, 0x9dc0, 0x9dc1, 0x9dc2,
03058 0x9dc3, 0x9dc4, 0x9dc5, 0x9dc6, 0x9dc7, 0x9dc8, 0x9dc9, 0x9dca,
03059 0x9dcb, 0x9dcc, 0x9dcd, 0x9dce, 0x9dcf, 0x9dd0, 0x9dd1, 0x9dd2,
03060 0x9dd3, 0x9dd4, 0x9dd5, 0x9dd6, 0x9dd7, 0x9dd8, 0x9dd9, 0x9dda,
03061 0x9ddb, 0x9ddc, 0x9ddd, 0x9dde, 0x9ddf, 0x9de0, 0x9de1, 0x9de2,
03062 0x9de3, 0x9de4, 0x9de5, 0x9de6, 0x9de7, 0x9de8, 0x9de9, 0x9dea,
03063 0x9deb, 0x9dec, 0x9ded, 0x9dee, 0x9def, 0x9df0, 0x9df1, 0x9df2,
03064 0x9df3, 0x9df4, 0x9df5, 0x9df6, 0x9df7, 0x9df8, 0x9df9, 0x9dfa,
03065 0x9dfb, 0x9dfc, 0x9dfd, 0x9dfe, 0x9dff, 0x9e00, 0x9e01, 0x9e02,
03066 0x9e03, 0x9e04, 0x9e05, 0x9e06, 0x9e07, 0x9e08, 0x9e09, 0x9e0a,
03067 /* 0xfa */
03068 0x9e0b, 0x9e0c, 0x9e0d, 0x9e0e, 0x9e0f, 0x9e10, 0x9e11, 0x9e12,
03069 0x9e13, 0x9e14, 0x9e15, 0x9e16, 0x9e17, 0x9e18, 0x9e19, 0x9e1a,
03070 0x9e1b, 0x9e1c, 0x9e1d, 0x9e1e, 0x9e1f, 0x9e20, 0x9e21, 0x9e22,
03071 0x9e23, 0x9e24, 0x9e25, 0x9e26, 0x9e27, 0x9e28, 0x9e29, 0x9e2a,
03072 0x9e2b, 0x9e2c, 0x9e2d, 0x9e2e, 0x9e2f, 0x9e30, 0x9e31, 0x9e32,
03073 0x9e33, 0x9e34, 0x9e35, 0x9e36, 0x9e37, 0x9e38, 0x9e39, 0x9e3a,
03074 0x9e3b, 0x9e3c, 0x9e3d, 0x9e3e, 0x9e3f, 0x9e40, 0x9e41, 0x9e42,
03075 0x9e43, 0x9e44, 0x9e45, 0x9e46, 0x9e47, 0x9e48, 0x9e49, 0x9e4a,
03076 0x9e4b, 0x9e4c, 0x9e4d, 0x9e4e, 0x9e4f, 0x9e50, 0x9e51, 0x9e52,
03077 0x9e53, 0x9e54, 0x9e55, 0x9e56, 0x9e57, 0x9e58, 0x9e59, 0x9e5a,
03078 0x9e5b, 0x9e5c, 0x9e5d, 0x9e5e, 0x9e5f, 0x9e60, 0x9e61, 0x9e62,
03079 0x9e63, 0x9e64, 0x9e65, 0x9e66, 0x9e67, 0x9e68, 0x9e69, 0x9e6a,
03080 0x9e6b, 0x9e6c, 0x9e6d, 0x9e6e, 0x9e6f, 0x9e70, 0x9e71, 0x9e72,
03081 0x9e73, 0x9e74, 0x9e75, 0x9e76, 0x9e77, 0x9e78, 0x9e79, 0x9e7a,
03082 0x9e7b, 0x9e7c, 0x9e7d, 0x9e7e, 0x9e7f, 0x9e80, 0x9e81, 0x9e82,
03083 0x9e83, 0x9e84, 0x9e85, 0x9e86, 0x9e87, 0x9e88, 0x9e89, 0x9e8a,
03084 0x9e8b, 0x9e8c, 0x9e8d, 0x9e8e, 0x9e8f, 0x9e90, 0x9e91, 0x9e92,
03085 0x9e93, 0x9e94, 0x9e95, 0x9e96, 0x9e97, 0x9e98, 0x9e99, 0x9e9a,
03086 0x9e9b, 0x9e9c, 0x9e9d, 0x9e9e, 0x9e9f, 0x9ea0, 0x9ea1, 0x9ea2,
03087 0x9ea3, 0x9ea4, 0x9ea5, 0x9ea6, 0x9ea7, 0x9ea8, 0x9ea9, 0x9eaa,
03088 0x9eab, 0x9eac, 0x9ead, 0x9eae, 0x9eaf, 0x9eb0, 0x9eb1, 0x9eb2,
03089 0x9eb3, 0x9eb4, 0x9eb5, 0x9eb6, 0x9eb7, 0x9eb8, 0x9eb9, 0x9eba,
03090 0x9ebb, 0x9ebc, 0x9ebd, 0x9ebe, 0x9ebf, 0x9ec0, 0x9ec1, 0x9ec2,
03091 0x9ec3, 0x9ec4, 0x9ec5, 0x9ec6, 0x9ec7, 0x9ec8, 0x9ec9, 0x9eca,
03092 0x9ecb, 0x9ecc, 0x9ecd, 0x9ece, 0x9ecf, 0x9ed0, 0x9ed1, 0x9ed2,
03093 0x9ed3, 0x9ed4, 0x9ed5, 0x9ed6, 0x9ed7, 0x9ed8, 0x9ed9, 0x9eda,
03094 0x9edb, 0x9edc, 0x9ede, 0x9eed, 0x9eee, 0x9eef, 0x9ef0, 0x9ef1,
03095 0x9ef2, 0x9ef3, 0x9ef4, 0x9ef5, 0x9ef6, 0x9ef7, 0x9ef8, 0x9ef9, 0x9efa,
03096 0x9efb, 0x9efc, 0x9efd, 0x9efe, 0x9eff, 0x9f00, 0x9f01, 0x9f02,
03097 0x9f03, 0x9f04, 0x9f05, 0x9f06, 0x9f07, 0x9f08, 0x9f09, 0x9f0a,
03098 0x9f0b, 0x9f0c, 0x9f0d, 0x9f0e, 0x9f0f, 0x9f10, 0x9f11, 0x9f12,
03099 0x9f13, 0x9f14, 0x9f15, 0x9f16, 0x9f17, 0x9f18, 0x9f19, 0x9f1a,
03100 0x9f1b, 0x9f1c, 0x9f1d, 0x9f1e, 0x9f1f, 0x9f20, 0x9f21, 0x9f22,
03101 0x9f23, 0x9f24, 0x9f25, 0x9f26, 0x9f27, 0x9f28, 0x9f29, 0x9f2a,
03102 0x9f2b, 0x9f2c, 0x9f2d, 0x9f2e, 0x9f2f, 0x9f30, 0x9f31, 0x9f32,
03103 0x9f33, 0x9f34, 0x9f35, 0x9f36, 0x9f37, 0x9f38, 0x9f39, 0x9f3a,
03104 0x9f3b, 0x9f3c, 0x9f3d, 0x9f3e, 0x9f3f, 0x9f40, 0x9f41, 0x9f42,
03105 0x9f43, 0x9f44, 0x9f45, 0x9f46, 0x9f47, 0x9f48, 0x9f49, 0x9f4a,
03106 0x9f4b, 0x9f4c, 0x9f4d, 0x9f4e, 0x9f4f, 0x9f50, 0x9f51, 0x9f52,
03107 0x9f53, 0x9f54, 0x9f55, 0x9f56, 0x9f57, 0x9f58, 0x9f59, 0x9f5a,
03108 0x9f5b, 0x9f5c, 0x9f5d, 0x9f5e, 0x9f5f, 0x9f60, 0x9f61, 0x9f62,
03109 0x9f63, 0x9f64, 0x9f65, 0x9f66, 0x9f67, 0x9f68, 0x9f69, 0x9f6a,
03110 0x9f6b, 0x9f6c, 0x9f6d, 0x9f6e, 0x9f6f, 0x9f70, 0x9f71, 0x9f72,
03111 0x9f73, 0x9f74, 0x9f75, 0x9f76, 0x9f77, 0x9f78, 0x9f79, 0x9f7a,
03112 0x9f7b, 0x9f7c, 0x9f7d, 0x9f7e, 0x9f7f, 0x9f80, 0x9f81, 0x9f82,
03113 0x9f83, 0x9f84, 0x9f85, 0x9f86, 0x9f87, 0x9f88, 0x9f89, 0x9f8a,
03114 0x9f8b, 0x9f8c, 0x9f8d, 0x9f8e, 0x9f8f, 0x9f90, 0x9f91, 0x9f92,
03115 0x9f93, 0x9f94, 0x9f95, 0x9f96, 0x9f97, 0x9f98, 0x9f99, 0x9f9a,
03116 0x9f9b, 0x9f9c, 0x9f9d, 0x9f9e, 0x9f9f, 0x9fa0, 0x9fa1, 0x9fa2,
03117 /* 0xfc */
03118 0x9fab, 0x9fac, 0x9fad, 0x9fae, 0x9faf, 0x9fb0, 0x9fb1, 0x9fb2,
03119 0x9fb3, 0x9fb4, 0x9fb5, 0x9fb6, 0x9fb7, 0x9fb8, 0x9fb9, 0x9fba,
03120 0x9fbb, 0x9fbc, 0x9fbd, 0x9fbe, 0x9fbf, 0x9fc0, 0x9fc1, 0x9fc2,
03121 0x9fc3, 0x9fc4, 0x9fc5, 0x9fc6, 0x9fc7, 0x9fc8, 0x9fc9, 0x9fca,
03122 0x9fcb, 0x9fcc, 0x9fcd, 0x9fce, 0x9fcf, 0x9fd0, 0x9fd1, 0x9fd2,
03123 0x9fd3, 0x9fd4, 0x9fd5, 0x9fd6, 0x9fd7, 0x9fd8, 0x9fd9, 0x9fda,
03124 0x9fdb, 0x9fdc, 0x9fde, 0x9fdd, 0x9fde, 0x9fed, 0x9fee, 0x9fef,
03125 0x9ff0, 0x9ff1, 0x9ff2, 0x9ff3, 0x9ff4, 0x9ff5, 0x9ff6, 0x9ff7,
03126 0x9ff8, 0x9ff9, 0x9ffa, 0x9ffb, 0x9ffc, 0x9ffd, 0x9ffe, 0x9fff,
03127 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03128 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03129 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03130 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03131 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03132 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03133 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03134 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03135 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03136 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
03137 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0x9fff,
```

```

03138 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03139 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03140 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03141 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03142 /* 0xfd */
03143 0x9f32, 0x9f33, 0x9f34, 0x9f35, 0x9f36, 0x9f38, 0x9f3a, 0x9f3c,
03144 0x9f3f, 0x9f40, 0x9f41, 0x9f42, 0x9f43, 0x9f45, 0x9f46, 0x9f47,
03145 0x9f48, 0x9f49, 0x9f4a, 0x9f4b, 0x9f4c, 0x9f4d, 0x9f4e, 0x9f4f,
03146 0x9f52, 0x9f53, 0x9f54, 0x9f55, 0x9f56, 0x9f57, 0x9f58, 0x9f59,
03147 0x9f5a, 0x9f5b, 0x9f5c, 0x9f5d, 0x9f5e, 0x9f5f, 0x9f60, 0x9f61,
03148 0x9f62, 0x9f63, 0x9f64, 0x9f65, 0x9f66, 0x9f67, 0x9f68, 0x9f69,
03149 0x9f6a, 0x9f6b, 0x9f6c, 0x9f6d, 0x9f6e, 0x9f6f, 0x9f70, 0x9f71,
03150 0x9f72, 0x9f73, 0x9f74, 0x9f75, 0x9f76, 0x9f77, 0x9f78, 0x9f79,
03151 0x9f7a, 0x9f7b, 0x9f7c, 0x9f7d, 0x9f7e, 0x9f81, 0x9f82, 0x9f8d,
03152 0x9f8e, 0x9f8f, 0x9f90, 0x9f91, 0x9f92, 0x9f93, 0x9f94, 0x9f95,
03153 0x9f96, 0x9f97, 0x9f98, 0x9f9c, 0x9f9d, 0x9f9e, 0x9fa1, 0x9fa2,
03154 0x9fa3, 0x9fa4, 0x9fa5, 0xf92c, 0xf979, 0xf995, 0xf9e7, 0xf9f1,
03155 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03156 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03157 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03158 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03159 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03160 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03161 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03162 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03163 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03164 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03165 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03166 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
03167 /* 0xfe */
03168 0xfa0c, 0xfa0d, 0xfa0e, 0xfa0f, 0xfa11, 0xfa13, 0xfa14, 0xfa18,
03169 0xfa1f, 0xfa20, 0xfa21, 0xfa23, 0xfa24, 0xfa27, 0xfa28, 0xfa29,
03170 };
03171
03172 static int
03173 cp936ext_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
03174 {
03175     unsigned char c1 = s[0];
03176     if ((c1 >= 0x81 && c1 <= 0xfe)) {
03177         if (n >= 2) {
03178             unsigned char c2 = s[1];
03179             if ((c2 >= 0x40 && c2 < 0x7f) || (c2 >= 0x80 && c2 < 0xff)) {
03180                 unsigned int i = 190 * (c1 - 0x81) + (c2 - (c2 >= 0x80 ? 0x41 : 0x40));
03181                 unsigned short wc = 0xffffd;
03182                 {
03183                     if (i < 23766)
03184                         wc = cp936ext_2uni_page81[i];
03185                 }
03186                 if (wc != 0xffffd) {
03187                     *pwc = (ucs4_t) wc;
03188                     return 2;
03189                 }
03190             }
03191             return RET_ILSEQ;
03192         }
03193         return RET_TOOFEW(0);
03194     }
03195     return RET_ILSEQ;
03196 }
03197 #endif /* NEED_TOWC */
03198
03199 #ifndef NEED_TOMB
03200
03201 static const unsigned short cp936ext_page0014[208] = {
03202     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xa0-0xa7*/
03203     0xa1a7, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xa8-0xaf*/
03204     0xa1e3, 0xa1c0, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xb0-0xb7*/
03205     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xb8-0xbf*/
03206     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xc0-0xc7*/
03207     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xc8-0xcf*/
03208     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xd0-0xd7*/
03209     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xd8-0xdf*/
03210     0xa8a4, 0xa8a2, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xe0-0xe7*/
03211     0xa8a8, 0xa8a6, 0xa8ba, 0x0000, 0xa8ac, 0xa8aa, 0x0000, 0x0000, /*0xe8-0xef*/
03212     0x0000, 0x0000, 0xa8b0, 0xa8ae, 0x0000, 0x0000, 0x0000, 0xa1c2, /*0xf0-0xf7*/
03213     0x0000, 0xa8b4, 0xa8b2, 0x0000, 0xa8b9, 0x0000, 0x0000, 0x0000, /*0xf8-0xff*/
03214     /* 0x0100 */
03215     0x0000, 0xa8a1, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x00-0x07*/
03216     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x08-0x0f*/
03217     0x0000, 0x0000, 0x0000, 0x0000, 0xa8a5, 0x0000, 0x0000, 0x0000, /*0x10-0x17*/
03218     0x0000, 0x0000, 0x0000, 0xa8a7, 0x0000, 0x0000, 0x0000, 0x0000, /*0x18-0x1f*/
03219     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x20-0x27*/
03220     0x0000, 0x0000, 0x0000, 0x0000, 0xa8a9, 0x0000, 0x0000, 0x0000, /*0x28-0x2f*/
03221     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x30-0x37*/
03222     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x38-0x3f*/
03223     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa8bd, 0x0000, 0x0000, /*0x40-0x47*/
03224     0xa8be, 0x0000, 0x0000, 0x0000, 0x0000, 0xa8ad, 0x0000, 0x0000, /*0x48-0x4f*/

```

```
03225 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x50-0x57*/
03226 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x58-0x5f*/
03227 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x60-0x67*/
03228 0x0000, 0x0000, 0x0000, 0xa8b1, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x68-0x6f*/
03229 };
03230 static const unsigned short cp936ext_page0039[24] = {
03231 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa8a3, 0x0000, /*0xc8-0xcf*/
03232 0xa8ab, 0x0000, 0xa8af, 0x0000, 0xa8b3, 0x0000, 0xa8b5, 0x0000, /*0xd0-0xd7*/
03233 0xa8b6, 0x0000, 0xa8b7, 0x0000, 0xa8b8, 0x0000, 0x0000, 0x0000, /*0xd8-0xdf*/
03234 };
03235 static const unsigned short cp936ext_page004a[24] = {
03236 0x0000, 0xa8bb, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x50-0x57*/
03237 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x58-0x5f*/
03238 0x0000, 0xa8c0, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x60-0x67*/
03239 };
03240 static const unsigned short cp936ext_page0058[32] = {
03241 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa1a6, /*0xc0-0xc7*/
03242 0x0000, 0xa1a5, 0xa840, 0xa841, 0x0000, 0x0000, 0x0000, 0x0000, /*0xc8-0xcf*/
03243 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xd0-0xd7*/
03244 0x0000, 0xa842, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xd8-0xdf*/
03245 };
03246 static const unsigned short cp936ext_page0072[64] = {
03247 0x0000, 0xa6a1, 0xa6a2, 0xa6a3, 0xa6a4, 0xa6a5, 0xa6a6, 0xa6a7, /*0x90-0x97*/
03248 0xa6a8, 0xa6a9, 0xa6aa, 0xa6ab, 0xa6ac, 0xa6ad, 0xa6ae, 0xa6af, /*0x98-0x9f*/
03249 0xa6b0, 0xa6b1, 0x0000, 0xa6b2, 0xa6b3, 0xa6b4, 0xa6b5, 0xa6b6, /*0xa0-0xa7*/
03250 0xa6b7, 0xa6b8, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xa8-0xaf*/
03251 0x0000, 0xa6c1, 0xa6c2, 0xa6c3, 0xa6c4, 0xa6c5, 0xa6c6, 0xa6c7, /*0xb0-0xb7*/
03252 0xa6c8, 0xa6c9, 0xa6ca, 0xa6cb, 0xa6cc, 0xa6cd, 0xa6ce, 0xa6cf, /*0xb8-0xbf*/
03253 0xa6d0, 0xa6d1, 0x0000, 0xa6d2, 0xa6d3, 0xa6d4, 0xa6d5, 0xa6d6, /*0xc0-0xc7*/
03254 0xa6d7, 0xa6d8, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xc8-0xcf*/
03255 };
03256 static const unsigned short cp936ext_page0080[88] = {
03257 0x0000, 0xa7a7, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x00-0x07*/
03258 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x08-0x0f*/
03259 0xa7a1, 0xa7a2, 0xa7a3, 0xa7a4, 0xa7a5, 0xa7a6, 0xa7a7, 0xa7a8, /*0x10-0x17*/
03260 0xa7aa, 0xa7ab, 0xa7ac, 0xa7ad, 0xa7ae, 0xa7af, 0xa7b0, 0xa7b1, /*0x18-0x1f*/
03261 0xa7b2, 0xa7b3, 0xa7b4, 0xa7b5, 0xa7b6, 0xa7b7, 0xa7b8, 0xa7b9, /*0x20-0x27*/
03262 0xa7ba, 0xa7bb, 0xa7bc, 0xa7bd, 0xa7be, 0xa7bf, 0xa7c0, 0xa7c1, /*0x28-0x2f*/
03263 0xa7d1, 0xa7d2, 0xa7d3, 0xa7d4, 0xa7d5, 0xa7d6, 0xa7d7, 0xa7d8, /*0x30-0x37*/
03264 0xa7da, 0xa7db, 0xa7dc, 0xa7dd, 0xa7de, 0xa7df, 0xa7e0, 0xa7e1, /*0x38-0x3f*/
03265 0xa7e2, 0xa7e3, 0xa7e4, 0xa7e5, 0xa7e6, 0xa7e7, 0xa7e8, 0xa7e9, /*0x40-0x47*/
03266 0xa7ea, 0xa7eb, 0xa7ec, 0xa7ed, 0xa7ee, 0xa7ef, 0xa7f0, 0xa7f1, /*0x48-0x4f*/
03267 0x0000, 0xa7d7, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x50-0x57*/
03268 };
03269 static const unsigned short cp936ext_page0402[48] = {
03270 0xa95c, 0x0000, 0x0000, 0xa843, 0xa1aa, 0xa844, 0xa1ac, 0x0000, /*0x10-0x17*/
03271 0xa1ae, 0xa1af, 0x0000, 0x0000, 0xa1b0, 0xa1b1, 0x0000, 0x0000, /*0x18-0x1f*/
03272 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa845, 0xa1ad, 0x0000, /*0x20-0x27*/
03273 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x28-0x2f*/
03274 0xa1eb, 0x0000, 0xa1e4, 0xa1e5, 0x0000, 0xa846, 0x0000, 0x0000, /*0x30-0x37*/
03275 0x0000, 0x0000, 0x0000, 0x0000, 0xa1f9, 0x0000, 0x0000, 0x0000, /*0x38-0x3f*/
03276 };
03277 static const unsigned short cp936ext_page0420[160] = {
03278 0x0000, 0x0000, 0x0000, 0xa1e6, 0x0000, 0xa847, 0x0000, 0x0000, /*0x00-0x07*/
03279 0x0000, 0xa848, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x08-0x0f*/
03280 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa1ed, 0x0000, /*0x10-0x17*/
03281 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x18-0x1f*/
03282 0x0000, 0xa959, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x20-0x27*/
03283 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x28-0x2f*/
03284 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x30-0x37*/
03285 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x38-0x3f*/
03286 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x40-0x47*/
03287 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x48-0x4f*/
03288 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x50-0x57*/
03289 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x58-0x5f*/
03290 0xa2f1, 0xa2f2, 0xa2f3, 0xa2f4, 0xa2f5, 0xa2f6, 0xa2f7, 0xa2f8, /*0x60-0x67*/
03291 0xa2f9, 0xa2fa, 0xa2fb, 0xa2fc, 0x0000, 0x0000, 0x0000, 0x0000, /*0x68-0x6f*/
03292 0xa2a1, 0xa2a2, 0xa2a3, 0xa2a4, 0xa2a5, 0xa2a6, 0xa2a7, 0xa2a8, /*0x70-0x77*/
03293 0xa2a9, 0xa2aa, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x78-0x7f*/
03294 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x80-0x87*/
03295 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x88-0x8f*/
03296 0xa1fb, 0xa1fc, 0xa1fa, 0xa1fd, 0x0000, 0x0000, 0xa849, 0xa84a, /*0x90-0x97*/
03297 0xa84b, 0xa84c, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x98-0x9f*/
03298 };
03299 static const unsigned short cp936ext_page0441[184] = {
03300 0xa1ca, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa1c7, /*0x08-0x0f*/
03301 0x0000, 0xa1c6, 0x0000, 0x0000, 0x0000, 0xa84d, 0x0000, 0x0000, /*0x10-0x17*/
03302 0x0000, 0x0000, 0xa1cc, 0x0000, 0x0000, 0xa1d8, 0xa1de, 0xa84e, /*0x18-0x1f*/
03303 0xa1cf, 0x0000, 0x0000, 0xa84f, 0x0000, 0xa1ce, 0x0000, 0xa1c4, /*0x20-0x27*/
03304 0xa1c5, 0xa1c9, 0xa1c8, 0xa1d2, 0x0000, 0x0000, 0xa1d3, 0x0000, /*0x28-0x2f*/
03305 0x0000, 0x0000, 0x0000, 0x0000, 0xa1e0, 0xa1df, 0xa1c3, 0xa1cb, /*0x30-0x37*/
03306 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa1d7, 0x0000, 0x0000, /*0x38-0x3f*/
03307 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x40-0x47*/
03308 0xa1d6, 0x0000, 0x0000, 0x0000, 0xa1d5, 0x0000, 0x0000, 0x0000, /*0x48-0x4f*/
03309 0x0000, 0x0000, 0xa850, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x50-0x57*/
03310 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x58-0x5f*/
03311 0xa1d9, 0xa1d4, 0x0000, 0x0000, 0xa1dc, 0xa1dd, 0xa851, 0xa852, /*0x60-0x67*/
```

Generated by Doxygen


```
03399 0xa5b0, 0xa5b1, 0xa5b2, 0xa5b3, 0xa5b4, 0xa5b5, 0xa5b6, 0xa5b7, /*0xb0-0xb7*/
03400 0xa5b8, 0xa5b9, 0xa5ba, 0xa5bb, 0xa5bc, 0xa5bd, 0xa5be, 0xa5bf, /*0xb8-0xbf*/
03401 0xa5c0, 0xa5c1, 0xa5c2, 0xa5c3, 0xa5c4, 0xa5c5, 0xa5c6, 0xa5c7, /*0xc0-0xc7*/
03402 0xa5c8, 0xa5c9, 0xa5ca, 0xa5cb, 0xa5cc, 0xa5cd, 0xa5ce, 0xa5cf, /*0xc8-0xcf*/
03403 0xa5d0, 0xa5d1, 0xa5d2, 0xa5d3, 0xa5d4, 0xa5d5, 0xa5d6, 0xa5d7, /*0xd0-0xd7*/
03404 0xa5d8, 0xa5d9, 0xa5da, 0xa5db, 0xa5dc, 0xa5dd, 0xa5de, 0xa5df, /*0xd8-0xdf*/
03405 0xa5e0, 0xa5e1, 0xa5e2, 0xa5e3, 0xa5e4, 0xa5e5, 0xa5e6, 0xa5e7, /*0xe0-0xe7*/
03406 0xa5e8, 0xa5e9, 0xa5ea, 0xa5eb, 0xa5ec, 0xa5ed, 0xa5ee, 0xa5ef, /*0xe8-0xef*/
03407 0xa5f0, 0xa5f1, 0xa5f2, 0xa5f3, 0xa5f4, 0xa5f5, 0xa5f6, 0x0000, /*0xf0-0xff*/
03408 0x0000, 0x0000, 0x0000, 0x0000, 0xa960, 0xa963, 0xa964, 0x0000, /*0xf8-0xff*/
03409 /* 0x3100 */
03410 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa8c5, 0xa8c6, 0xa8c7, /*0x00-0x07*/
03411 0xa8c8, 0xa8c9, 0xa8ca, 0xa8cb, 0xa8cc, 0xa8cd, 0xa8ce, 0xa8cf, /*0x08-0x0f*/
03412 0xa8d0, 0xa8d1, 0xa8d2, 0xa8d3, 0xa8d4, 0xa8d5, 0xa8d6, 0xa8d7, /*0x10-0x17*/
03413 0xa8d8, 0xa8d9, 0xa8da, 0xa8db, 0xa8dc, 0xa8dd, 0xa8de, 0xa8df, /*0x18-0x1f*/
03414 0xa8e0, 0xa8e1, 0xa8e2, 0xa8e3, 0xa8e4, 0xa8e5, 0xa8e6, 0xa8e7, /*0x20-0x27*/
03415 0xa8e8, 0xa8e9, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x28-0x2f*/
03416 };
03417 static const unsigned short cp936ext_page0644[24] = {
03418 0xa2e5, 0xa2e6, 0xa2e7, 0xa2e8, 0xa2e9, 0xa2ea, 0xa2eb, 0xa2ec, /*0x20-0x27*/
03419 0xa2ed, 0xa2ee, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x28-0x2f*/
03420 0x0000, 0xa95a, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x30-0x37*/
03421 };
03422 static const unsigned short cp936ext_page0671[80] = {
03423 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa94a, 0xa94b, /*0x88-0x8f*/
03424 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x90-0x97*/
03425 0x0000, 0x0000, 0x0000, 0x0000, 0xa94c, 0xa94d, 0xa94e, 0x0000, /*0x98-0x9f*/
03426 0x0000, 0xa94f, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xa0-0xa7*/
03427 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xa8-0xaf*/
03428 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xb0-0xb7*/
03429 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xb8-0xbf*/
03430 0x0000, 0x0000, 0x0000, 0x0000, 0xa950, 0x0000, 0x0000, 0x0000, /*0xc0-0xc7*/
03431 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa951, 0x0000, /*0xc8-0xcf*/
03432 0x0000, 0xa952, 0xa953, 0x0000, 0x0000, 0xa954, 0x0000, 0x0000, /*0xd0-0xd7*/
03433 };
03434 static const unsigned short cp936ext_page09c0[20904] = {
03435 0xd2bb, 0xb6a1, 0x8140, 0xc6df, 0x8141, 0x8142, 0x8143, 0xcdf2, /*0x00-0x07*/
03436 0xd5c9, 0xc8fd, 0xc9cf, 0xcfc2, 0xd8a2, 0xb2bb, 0xd3eb, 0x8144, /*0x08-0x0f*/
03437 0xd8a4, 0xb3f3, 0x8145, 0xd7a8, 0xc7d2, 0xd8a7, 0xcac0, 0x8146, /*0x10-0x17*/
03438 0xc7f0, 0xb1fb, 0xd2b5, 0xb4d4, 0xb6ab, 0xcbbf, 0xd8a9, 0x8147, /*0x18-0x1f*/
03439 0x8148, 0x8149, 0xb6aa, 0x814a, 0xc1bd, 0xd1cf, 0x814b, 0xc9a5, /*0x20-0x27*/
03440 0xd8ad, 0x814c, 0xb8f6, 0xd1be, 0xe3dc, 0xd6d0, 0x814d, 0x814e, /*0x28-0x2f*/
03441 0xb7e1, 0x814f, 0xb4ae, 0x8150, 0xc1d9, 0x8151, 0xd8bc, 0x8152, /*0x30-0x37*/
03442 0xcde8, 0xb5a4, 0xceaa, 0xd6f7, 0x8153, 0xc0f6, 0xbed9, 0xd8af, /*0x38-0x3f*/
03443 0x8154, 0x8155, 0x8156, 0xc4cb, 0x8157, 0xbec3, 0x8158, 0xd8b1, /*0x40-0x47*/
03444 0xc3b4, 0xd2e5, 0x8159, 0xd6ae, 0xceda, 0xd5a7, 0xbaf5, 0xb7a6, /*0x48-0x4f*/
03445 0xc0d6, 0x815a, 0xc6b9, 0xc5d2, 0xc7c7, 0x815b, 0xb9d4, 0x815c, /*0x50-0x57*/
03446 0xb3cb, 0xd2d2, 0x815d, 0x815e, 0xd8bf, 0xbec5, 0xc6f2, 0xd2b2, /*0x58-0x5f*/
03447 0xcfb0, 0xcfe7, 0x815f, 0x8160, 0x8161, 0x8162, 0xcae9, 0x8163, /*0x60-0x67*/
03448 0x8164, 0xd8c0, 0x8165, 0x8166, 0x8167, 0x8168, 0x8169, 0x816a, /*0x68-0x6f*/
03449 0xc2f2, 0xc2d2, 0x816b, 0xc8e9, 0x816c, 0x816d, 0x816e, 0x816f, /*0x70-0x77*/
03450 0x8170, 0x8171, 0x8172, 0x8173, 0x8174, 0x8175, 0xc7ac, 0x8176, /*0x78-0x7f*/
03451 0x8177, 0x8178, 0x8179, 0x817a, 0x817b, 0x817c, 0xc1cb, 0x817d, /*0x80-0x87*/
03452 0xd3e8, 0xd5f9, 0x817e, 0xcac2, 0xb6fe, 0xd8a1, 0xd3da, 0xbff7, /*0x88-0x8f*/
03453 0x8180, 0xd4c6, 0xbba5, 0xd8c1, 0xcee5, 0xbeae, 0x8181, 0x8182, /*0x90-0x97*/
03454 0xd8a8, 0x8183, 0xd1c7, 0xd0a9, 0x8184, 0x8185, 0x8186, 0xd8bd, /*0x98-0x9f*/
03455 0xd9ef, 0xcdf6, 0xbfba, 0x8187, 0xbdbb, 0xbaa5, 0xd2e0, 0xb2fa, /*0xa0-0xaf*/
03456 0xbae0, 0xc4b6, 0x8188, 0xcfed, 0xbea9, 0xcda4, 0xc1c1, 0x8189, /*0xa8-0xaf*/
03457 0x818a, 0x818b, 0xc7d7, 0xd9f1, 0x818c, 0xd9f4, 0x818d, 0x818e, /*0xb0-0xb7*/
03458 0x818f, 0x8190, 0xc8cb, 0xd8e9, 0x8191, 0x8192, 0x8193, 0xd2da, /*0xb8-0xbf*/
03459 0xcab2, 0xc8ca, 0xd8ec, 0xd8ea, 0xd8c6, 0xbdf6, 0xc6cd, 0xb3f0, /*0xc0-0xc7*/
03460 0x8194, 0xd8eb, 0xbdf1, 0xbde9, 0x8195, 0xc8d4, 0xb4d3, 0x8196, /*0xc8-0xcf*/
03461 0x8197, 0xc2d8, 0x8198, 0xb2d6, 0xd7d0, 0xcacb, 0xcbbf, 0xd5cc, /*0xd0-0xd7*/
03462 0xb8b6, 0xcfc9, 0x8199, 0x819a, 0x819b, 0xd9da, 0xd8f0, 0xc7aa, /*0xd8-0xdf*/
03463 0x819c, 0xd8ee, 0x819d, 0xb4fa, 0xc1ee, 0xd2d4, 0x819e, 0x819f, /*0xe0-0xe7*/
03464 0xd8ed, 0x81a0, 0xd2c7, 0xd8ef, 0xc3c7, 0x81a1, 0x81a2, 0x81a3, /*0xe8-0xef*/
03465 0xd1f6, 0x81a4, 0xd6d9, 0xd8f2, 0x81a5, 0xd8f5, 0xbcf6, 0xbcd5, /*0xf0-0xff*/
03466 0x81a6, 0x81a7, 0x81a8, 0xc8ce, 0x81a9, 0xb7dd, 0x81aa, 0xb7c2, /*0xf8-0xff*/
03467 /* 0x4f00 */
03468 0x81ab, 0xc6f3, 0x81ac, 0x81ad, 0x81ae, 0x81af, 0x81b0, 0x81b1, /*0x00-0x07*/
03469 0x81b2, 0xd8f8, 0xd2c1, 0x81b3, 0x81b4, 0xcee9, 0xbcbf, 0xb7fc, /*0x08-0x0f*/
03470 0xb7a5, 0xd0dd, 0x81b5, 0x81b6, 0x81b7, 0x81b8, 0x81b9, 0xd6da, /*0x10-0x17*/
03471 0xd3c5, 0xbbbf, 0xbbe1, 0xd8f1, 0x81ba, 0x81bb, 0xc9a1, 0xceb0, /*0x18-0x1f*/
03472 0xb4ab, 0x81bc, 0xd8f3, 0x81bd, 0xc9cb, 0xd8f6, 0xc2d7, 0xd8f7, /*0x20-0x27*/
03473 0x81be, 0x81bf, 0xceb1, 0xd8f9, 0x81c0, 0x81c1, 0x81c2, 0xb2ae, /*0x28-0x2f*/
03474 0xb9c0, 0x81c3, 0xd9a3, 0x81c4, 0xb0e9, 0x81c5, 0xc1e6, 0x81c6, /*0x30-0x37*/
03475 0xc9ec, 0x81c7, 0xcbc5, 0x81c8, 0xcbc6, 0xd9a4, 0x81c9, 0x81ca, /*0x38-0x3f*/
03476 0x81cb, 0x81cc, 0x81cd, 0xb5e8, 0x81ce, 0x81cf, 0xb5ab, 0x81d0, /*0x40-0x47*/
03477 0x81d1, 0x81d2, 0x81d3, 0x81d4, 0x81d5, 0xcebb, 0xb5cd, 0xd7a1, /*0x48-0x4f*/
03478 0xd7f4, 0xd3d3, 0x81d6, 0xcce5, 0x81d7, 0xbace, 0x81d8, 0xd9a2, /*0x50-0x57*/
03479 0xd9dc, 0xd3e0, 0xd8fd, 0xb7f0, 0xd7f7, 0xd8fe, 0xd8fa, 0xd9a1, /*0x58-0x5f*/
03480 0xc4e3, 0x81d9, 0x81da, 0xd3b6, 0xd8f4, 0xd9dd, 0x81db, 0xd8fb, /*0x60-0x67*/
03481 0x81dc, 0xc5e5, 0x81dd, 0x81de, 0xc0d0, 0x81df, 0x81e0, 0xd1f0, /*0x68-0x6f*/
03482 0xb0db, 0x81e1, 0x81e2, 0xbcd1, 0xd9a6, 0x81e3, 0xd9a5, 0x81e4, /*0x70-0x77*/
03483 0x81e5, 0x81e6, 0x81e7, 0xd9ac, 0xd9ae, 0x81e8, 0xd9ad, 0xcab9, /*0x78-0x7f*/
03484 0x81e9, 0x81ea, 0x81eb, 0xd9a9, 0xd6b6, 0x81ec, 0x81ed, 0x81ee, /*0x80-0x87*/
03485 0xb3de, 0xd9a8, 0x81ef, 0xc0fd, 0x81f0, 0xcacc, 0x81f1, 0xd9aa, /*0x88-0x8f*/
```

```

03486 0x81f2, 0xd9a7, 0x81f3, 0x81f4, 0xd9b0, 0x81f5, 0x81f6, 0xb6b1, /*0x90-0x97*/
03487 0x81f7, 0x81f8, 0x81f9, 0xb9a9, 0x81fa, 0xd2c0, 0x81fb, 0x81fc, /*0x98-0x9f*/
03488 0xcfc0, 0x81fd, 0x81fe, 0xc2c2, 0x8240, 0xbdc4, 0xd5ec, 0xb2e0, /*0xa0-0xa7*/
03489 0xc7c8, 0xbfeb, 0xd9ad, 0x8241, 0xd9af, 0x8242, 0xceea, 0xbaee, /*0xa8-0xaf*/
03490 0x8243, 0x8244, 0x8245, 0x8246, 0x8247, 0xc7d6, 0x8248, 0x8249, /*0xb0-0xb7*/
03491 0x824a, 0x824b, 0x824c, 0x824d, 0x824e, 0x824f, 0x8250, 0xb1e3, /*0xb8-0xbf*/
03492 0x8251, 0x8252, 0x8253, 0xb4d9, 0xb6ed, 0xd9b4, 0x8254, 0x8255, /*0xc0-0xc7*/
03493 0x8256, 0x8257, 0xbfa1, 0x8258, 0x8259, 0x825a, 0xd9de, 0xc7ce, /*0xc8-0xcf*/
03494 0xc0fe, 0xd9b8, 0x825b, 0x825c, 0x825d, 0x825e, 0x825f, 0xcdb7, /*0xd0-0xd7*/
03495 0xb7fd, 0x8260, 0xd9b5, 0x8261, 0xd9b7, 0xb1a3, 0xd3e1, 0xd9b9, /*0xd8-0xdf*/
03496 0x8262, 0xd0c5, 0x8263, 0xd9b6, 0x8264, 0x8265, 0xd9b1, 0x8266, /*0xe0-0xef*/
03497 0xd9b2, 0xc1a9, 0xd9b3, 0x8267, 0x8268, 0xbcf3, 0xd0de, 0xb8a9, /*0xe8-0xef*/
03498 0x8269, 0xbbee3, 0x826a, 0xd9bd, 0x826b, 0x826c, 0x826d, 0x826e, /*0xf0-0xf7*/
03499 0xd9ba, 0x826f, 0xb0b3, 0x8270, 0x8271, 0x8272, 0xd9c2, 0x8273, /*0xf8-0xff*/
03500 /* 0x5000 */
03501 0x8274, 0x8275, 0x8276, 0x8277, 0x8278, 0x8279, 0x827a, 0x827b, /*0x00-0x07*/
03502 0x827c, 0x827d, 0x827e, 0x8280, 0xd9c4, 0xb1b6, 0x8281, 0xd9bf, /*0x08-0x0f*/
03503 0x8282, 0x8283, 0xb5b9, 0x8284, 0xbef3, 0x8285, 0x8286, 0x8287, /*0x10-0x17*/
03504 0xccc8, 0xbaf2, 0xd2d0, 0x8288, 0xd9c3, 0x8289, 0x828a, 0xbde8, /*0x18-0x1f*/
03505 0x828b, 0xb3ab, 0x828c, 0x828d, 0x828e, 0xd9c5, 0xbbeb, 0x828f, /*0x20-0x27*/
03506 0xd9c6, 0xd9bb, 0xc4df, 0x8290, 0xd9be, 0xd9c1, 0xd9c0, 0x8291, /*0x28-0x2f*/
03507 0x8292, 0x8293, 0x8294, 0x8295, 0x8296, 0x8297, 0x8298, 0x8299, /*0x30-0x37*/
03508 0x829a, 0x829b, 0xd5ae, 0x829c, 0xd6b5, 0x829d, 0xc7e3, 0x829e, /*0x38-0x3f*/
03509 0x829f, 0x82a0, 0x82a1, 0xd9c8, 0x82a2, 0x82a3, 0x82a4, 0xbcd9, /*0x40-0x47*/
03510 0xd9ca, 0x82a5, 0x82a6, 0x82a7, 0xd9bc, 0x82a8, 0xd9cb, 0xc6ab, /*0x48-0x4f*/
03511 0x82a9, 0x82aa, 0x82ab, 0x82ac, 0x82ad, 0xd9c9, 0x82ae, 0x82af, /*0x50-0x57*/
03512 0x82b0, 0x82b1, 0xd7f6, 0x82b2, 0xcda3, 0x82b3, 0x82b4, 0x82b5, /*0x58-0x5f*/
03513 0x82b6, 0x82b7, 0x82b8, 0x82b9, 0x82ba, 0xbda1, 0x82bb, 0x82bc, /*0x60-0x67*/
03514 0x82bd, 0x82be, 0x82bf, 0x82c0, 0xd9cc, 0x82c1, 0x82c2, 0x82c3, /*0x68-0x6f*/
03515 0x82c4, 0x82c5, 0x82c6, 0x82c7, 0x82c8, 0x82c9, 0xc5bc, 0xcdb5, /*0x70-0x77*/
03516 0x82ca, 0x82cb, 0x82cc, 0xd9cd, 0x82cd, 0x82ce, 0xd9c7, 0xb3a5, /*0x78-0x7f*/
03517 0xbffe, 0x82cf, 0x82d0, 0x82d1, 0x82d2, 0xb8b5, 0x82d3, 0x82d4, /*0x80-0x87*/
03518 0xc0fc, 0x82d5, 0x82d6, 0x82d7, 0x82d8, 0xb0f8, 0x82d9, 0x82da, /*0x88-0x8f*/
03519 0x82db, 0x82dc, 0x82dd, 0x82de, 0x82df, 0x82e0, 0x82e1, 0x82e2, /*0x90-0x97*/
03520 0x82e3, 0x82e4, 0x82e5, 0x82e6, 0x82e7, 0x82e8, 0x82e9, 0x82ea, /*0x98-0x9f*/
03521 0x82eb, 0x82ec, 0x82ed, 0xb4f6, 0x82ee, 0xd9ce, 0x82ef, 0xd9cf, /*0xa0-0xaf*/
03522 0xb4a2, 0xd9d0, 0x82f0, 0x82f1, 0xb4df, 0x82f2, 0x82f3, 0x82f4, /*0xaa-0xaf*/
03523 0x82f5, 0x82f6, 0xb0c1, 0x82f7, 0x82f8, 0x82f9, 0x82fa, 0x82fb, /*0xb0-0xb7*/
03524 0x82fc, 0x82fd, 0xd9d1, 0xc9b5, 0x82fe, 0x8340, 0x8341, 0x8342, /*0xb8-0xbf*/
03525 0x8343, 0x8344, 0x8345, 0x8346, 0x8347, 0x8348, 0x8349, 0x834a, /*0xc0-0xc7*/
03526 0x834b, 0x834c, 0x834d, 0x834e, 0x834f, 0x8350, 0x8351, 0xcff1, /*0xc8-0xcf*/
03527 0x8352, 0x8353, 0x8354, 0x8355, 0x8356, 0x8357, 0xd9d2, 0x8358, /*0xd0-0xd7*/
03528 0x8359, 0x835a, 0xc1c5, 0x835b, 0x835c, 0x835d, 0x835e, 0x835f, /*0xd8-0xdf*/
03529 0x8360, 0x8361, 0x8362, 0x8363, 0x8364, 0x8365, 0xd9d6, 0xc9ae, /*0xe0-0xef*/
03530 0x8366, 0x8367, 0x8368, 0x8369, 0xd9d5, 0xd9d4, 0xd9d7, 0x836a, /*0xe8-0xef*/
03531 0x836b, 0x836c, 0x836d, 0xcdbd, 0x836e, 0xbda9, 0x836f, 0x8370, /*0xf0-0xf7*/
03532 0x8371, 0x8372, 0x8373, 0xc6a7, 0x8374, 0x8375, 0x8376, 0x8377, /*0xf8-0xff*/
03533 /* 0x5100 */
03534 0x8378, 0x8379, 0x837a, 0x837b, 0x837c, 0x837d, 0xd9d3, 0xd9d8, /*0x00-0x07*/
03535 0x837e, 0x8380, 0x8381, 0xd9d9, 0x8382, 0x8383, 0x8384, 0x8385, /*0x08-0x0f*/
03536 0x8386, 0x8387, 0xc8e5, 0x8388, 0x8389, 0x838a, 0x838b, 0x838c, /*0x10-0x17*/
03537 0x838d, 0x838e, 0x838f, 0x8390, 0x8391, 0x8392, 0x8393, 0x8394, /*0x18-0x1f*/
03538 0x8395, 0xc0dc, 0x8396, 0x8397, 0x8398, 0x8399, 0x839a, 0x839b, /*0x20-0x27*/
03539 0x839c, 0x839e, 0x839f, 0x83a0, 0x83a1, 0x83a2, 0x83a3, 0x83a4, /*0x28-0x2f*/
03540 0x83a5, 0x83a6, 0x83a7, 0x83a8, 0x83a9, 0x83aa, 0x83ab, 0x83ac, /*0x30-0x37*/
03541 0x83ad, 0x83ae, 0x83af, 0x83b0, 0x83b1, 0x83b2, 0xb6f9, 0x83b3, /*0x38-0x3f*/
03542 0xd8a3, 0xd4ca, 0x83b3, 0xd4aa, 0xd0d6, 0xb3e4, 0xd5d7, 0x83b4, /*0x40-0x47*/
03543 0xcfc8, 0xb9e2, 0x83b5, 0xbfcf, 0x83b6, 0xc3e2, 0x83b7, 0x83b8, /*0x48-0x4f*/
03544 0x83b9, 0xb6d2, 0x83ba, 0x83bb, 0xc3c3, 0xd9ee, 0xd9f0, 0x83bc, /*0x50-0x57*/
03545 0x83bd, 0x83be, 0xb5b3, 0x83bf, 0xb6b5, 0x83c0, 0x83c1, 0x83c2, /*0x58-0x5f*/
03546 0x83c3, 0x83c4, 0xbea4, 0x83c5, 0x83c6, 0xc8eb, 0x83c7, 0x83c8, /*0x60-0x67*/
03547 0xc8ab, 0x83c9, 0x83ca, 0xb0cb, 0xb9ab, 0xc1f9, 0xd9e2, 0x83cb, /*0x68-0x6f*/
03548 0xc0bc, 0xb9b2, 0x83cc, 0xb9d8, 0xd0cb, 0xb1f8, 0xc6e4, 0xbef, /*0x70-0x77*/
03549 0xb5e4, 0xd7c8, 0x83cd, 0xd1f8, 0xbce6, 0xcade, 0x83ce, 0x83cf, /*0x78-0x7f*/
03550 0xbcbcd, 0xd9e6, 0xd8e7, 0x83d0, 0x83d1, 0xc4da, 0x83d2, 0x83d3, /*0x80-0x87*/
03551 0xb8d4, 0xc8bd, 0x83d4, 0x83d5, 0xb2e1, 0xd4d9, 0x83d6, 0x83d7, /*0x88-0x8f*/
03552 0x83d8, 0x83d9, 0xc3b0, 0x83da, 0x83db, 0xc3e1, 0xdaa2, 0xc8df, /*0x90-0x97*/
03553 0x83dc, 0xd0b4, 0x83dd, 0xbefc, 0xc5a9, 0x83de, 0x83df, 0x83e0, /*0x98-0x9f*/
03554 0xb9da, 0x83e1, 0xdaa3, 0x83e2, 0xd4a9, 0xdaa4, 0x83e3, 0x83e4, /*0xa0-0xaf*/
03555 0x83e5, 0x83e6, 0x83e7, 0xd9fb, 0xb6ac, 0x83e8, 0x83e9, 0xb7eb, /*0xaa-0xaf*/
03556 0xb1f9, 0xd9fc, 0xb3e5, 0xbef6, 0x83ea, 0xbff6, 0xd2b1, 0xc0e4, /*0xb0-0xb7*/
03557 0x83eb, 0x83ec, 0x83ed, 0xb6b3, 0xd9fe, 0xd9fd, 0x83ee, 0x83ef, /*0xb8-0xbf*/
03558 0xbebb, 0x83f0, 0x83f1, 0x83f2, 0xc6e0, 0x83f3, 0xd7bc, 0xdaa1, /*0xc0-0xcf*/
03559 0x83f4, 0xc1b9, 0x83f5, 0xb5f2, 0xc1e8, 0x83f6, 0x83f7, 0xbcf5, /*0xc8-0xcf*/
03560 0x83f8, 0xb4d5, 0x83f9, 0x83fa, 0x83fb, 0x83fc, 0x83fd, 0x83fe, /*0xd0-0xd7*/
03561 0x8440, 0x8441, 0x8442, 0xc1dd, 0x8443, 0xc4fd, 0x8444, 0x8445, /*0xd8-0xdf*/
03562 0xbcb8, 0xb7b2, 0x8446, 0x8447, 0x8448, 0x8449, 0x844a, 0x844b, /*0xe0-0xef*/
03563 0x844c, 0x844d, 0x844e, 0xd9ec, 0x844f, 0xc6be, 0x844f, 0xbfad, /*0xe8-0xef*/
03564 0xbcbcb, 0x8450, 0x8451, 0xb5ca, 0x8452, 0xdbcb, 0xd0d7, 0x8453, /*0xf0-0xf7*/
03565 0xcdb9, 0xb0bc, 0xb3f6, 0xbbf7, 0xdabc, 0xbaaf, 0x8454, 0xd4e4, /*0xf8-0xff*/
03566 /* 0x5200 */
03567 0xb5b6, 0xb5f3, 0xd8d6, 0xc8d0, 0x8455, 0x8456, 0xb7d6, 0xc7d0, /*0x00-0x07*/
03568 0xd8d7, 0x8457, 0xbfaf, 0x8458, 0x8459, 0xdbbb, 0xd8d8, 0x845a, /*0x08-0x0f*/
03569 0x845b, 0xd0cc, 0xbbae, 0x845c, 0x845d, 0x845e, 0xebbe, 0xc1d0, /*0x10-0x17*/
03570 0xc1f5, 0xd4f2, 0xb8d5, 0xb4b4, 0x845f, 0xb3f5, 0x8460, 0x8461, /*0x18-0x1f*/
03571 0xc9be, 0x8462, 0x8463, 0x8464, 0xc5d0, 0x8465, 0x8466, 0x8467, /*0x20-0x27*/
03572 0xc5d9, 0xc0fb, 0x8468, 0xb1f0, 0x8469, 0xd8d9, 0xb9ce, 0x846a, /*0x28-0x2f*/

```



```
03573 0xb5bd, 0x846b, 0x846c, 0xd8da, 0x846d, 0x846e, 0xd6c6, 0xcba2, /*0x30-0x37*/
03574 0xc8af, 0xc9b2, 0xb4cc, 0xbfcc, 0x846f, 0xb9f4, 0x8470, 0xd8db, /*0x38-0x3f*/
03575 0xd8dc, 0xb6e7, 0xbcc1, 0xccea, 0x8471, 0x8472, 0x8473, 0x8474, /*0x40-0x47*/
03576 0x8475, 0x8476, 0xcff7, 0x8477, 0xd8dd, 0xc7b0, 0x8478, 0x8479, /*0x48-0x4f*/
03577 0xb9d0, 0xbda3, 0x847a, 0x847b, 0xccde, 0x847c, 0xc6ca, 0x847d, /*0x50-0x57*/
03578 0x847e, 0x8480, 0x8481, 0x8482, 0xd8e0, 0x8483, 0xd8de, 0x8484, /*0x58-0x5f*/
03579 0x8485, 0xd8df, 0x8486, 0x8487, 0x8488, 0xb0fe, 0x8489, 0xbee7, /*0x60-0x67*/
03580 0x848a, 0xcaa3, 0xbcf4, 0x848b, 0x848c, 0x848d, 0x848e, 0xb8b1, /*0x68-0x6f*/
03581 0x848f, 0x8490, 0xb8ee, 0x8491, 0x8492, 0x8493, 0x8494, 0x8495, /*0x70-0x77*/
03582 0x8496, 0x8497, 0x8498, 0x8499, 0x849a, 0xd8e2, 0x849b, 0xbdcf, /*0x78-0x7f*/
03583 0x849c, 0xd8e4, 0xd8e3, 0x849d, 0x849e, 0x849f, 0x84a0, 0x84a1, /*0x80-0x87*/
03584 0xc5fc, 0x84a2, 0x84a3, 0x84a4, 0x84a5, 0x84a6, 0x84a7, 0x84a8, /*0x88-0x8f*/
03585 0xd8e5, 0x84a9, 0x84aa, 0xd8e6, 0x84ab, 0x84ac, 0x84ad, 0x84ae, /*0x90-0x97*/
03586 0x84af, 0x84b0, 0x84b1, 0xc1a6, 0x84b2, 0xc8b0, 0xb0ec, 0xb9a6, /*0x98-0x9f*/
03587 0xbcd3, 0xcfc1, 0xd8bd, 0xc1d3, 0x84b3, 0x84b4, 0x84b5, 0x84b6, /*0xa0-0xaf*/
03588 0xb6af, 0xd6fa, 0xc5ac, 0xbdd9, 0xdbbe, 0xdbbf, 0x84b7, 0x84b8, /*0xa8-0xaf*/
03589 0x84b9, 0xc0f8, 0xbea2, 0xc0cd, 0x84ba, 0x84bb, 0x84bc, 0x84bd, /*0xb0-0xb7*/
03590 0x84be, 0x84bf, 0x84c0, 0x84c1, 0x84c2, 0x84c3, 0xdbcc, 0xcac6, /*0xb8-0xbf*/
03591 0x84c4, 0x84c5, 0x84c6, 0xb2aa, 0x84c7, 0x84c8, 0x84c9, 0xd3c2, /*0xc0-0xc7*/
03592 0x84ca, 0xc3e3, 0x84cb, 0xd1ab, 0x84cc, 0x84cd, 0x84ce, 0x84cf, /*0xc8-0xcf*/
03593 0xdbcc, 0x84d0, 0xc0d5, 0x84d1, 0x84d2, 0x84d3, 0xdbcc, 0x84d4, /*0xd0-0xd7*/
03594 0xbfb1, 0x84d5, 0x84d6, 0x84d7, 0x84d8, 0x84d9, 0x84da, 0xc4bc, /*0xd8-0xdf*/
03595 0x84db, 0x84dc, 0x84dd, 0x84de, 0xc7da, 0x84df, 0x84e0, 0x84e1, /*0xe0-0xe7*/
03596 0x84e2, 0x84e3, 0x84e4, 0x84e5, 0x84e6, 0x84e7, 0x84e8, 0x84e9, /*0xe8-0xef*/
03597 0xdbc4, 0x84ea, 0x84eb, 0x84ec, 0x84ed, 0x84ee, 0x84ef, 0x84f0, /*0xf0-0xf7*/
03598 0x84f1, 0x84f2, 0xc9d7, 0x84f3, 0x84f4, 0xb9b4, 0xcef0, /*0xf8-0xff*/
03599 /* 0x5300 */
03600 0xd4c8, 0x84f5, 0x84f6, 0x84f7, 0x84f8, 0xb0fc, 0xb4d2, 0x84f9, /*0x00-0x07*/
03601 0xd0d9, 0x84fa, 0x84fb, 0x84fc, 0x84fd, 0xd9e9, 0x84fe, 0xdecf, /*0x08-0x0f*/
03602 0xd9eb, 0x8540, 0x8541, 0x8542, 0x8543, 0xd8b0, 0xbbaa, 0xb1b1, /*0x10-0x17*/
03603 0x8544, 0xb3d7, 0xd8ce, 0x8545, 0x8546, 0xd4d1, 0x8547, 0x8548, /*0x18-0x1f*/
03604 0xbdb3, 0xbffe, 0x8549, 0xcffb, 0x854a, 0x854b, 0xd8d0, 0x854c, /*0x20-0x27*/
03605 0x854d, 0x854e, 0xb7cb, 0x854f, 0x8550, 0x8551, 0xd8d1, 0x8552, /*0x28-0x2f*/
03606 0x8553, 0x8554, 0x8555, 0x8556, 0x8557, 0x8558, 0x8559, 0x855a, /*0x30-0x37*/
03607 0x855b, 0xc6a5, 0xc7f8, 0xd2bd, 0x855c, 0x855d, 0xd8d2, 0xc4e4, /*0x38-0x3f*/
03608 0x855e, 0xc6aa, 0x855f, 0xc7a7, 0x8560, 0xd8a6, 0x8561, 0xc9fd, /*0x40-0x47*/
03609 0xcce7, 0xbdbd, 0xb0eb, 0x8562, 0x8563, 0x8564, 0xbbaa, 0xd0ad, /*0x48-0x4f*/
03610 0x8565, 0xb1b0, 0xd7e4, 0xd7bf, 0x8566, 0xb5a5, 0xc2f4, 0xc4cf, /*0x50-0x57*/
03611 0x8567, 0x8568, 0xb2a9, 0x8569, 0xb2b7, 0x856a, 0xb1e5, 0xdfb2, /*0x58-0x5f*/
03612 0xd5bc, 0xbfa8, 0xc2ac, 0xd8d5, 0xc2b1, 0x856b, 0xd8d4, 0xcdc4, /*0x60-0x67*/
03613 0x856c, 0xdae0, 0x856d, 0xccec, 0x856e, 0x856f, 0xd8b4, 0xc3ae, /*0x68-0x6f*/
03614 0xd3a1, 0xc6a3, 0x8570, 0xbcb4, 0xc2d1, 0x8571, 0xbed, /*0x70-0x77*/
03615 0xd0b6, 0x8572, 0xdae1, 0x8573, 0x8574, 0x8575, 0x8576, 0xc7e4, /*0x78-0x7f*/
03616 0x8577, 0x8578, 0xb3a7, 0x8579, 0xb6f2, 0xc0fc, 0xc0fa, 0x857a, /*0x80-0x87*/
03617 0x857b, 0xc0f7, 0x857c, 0xd1b9, 0xd1e1, 0xd8c7, 0x857d, 0x857e, /*0x88-0x8f*/
03618 0x8580, 0x8581, 0x8582, 0x8583, 0x8584, 0xb2de, 0x8585, 0x8586, /*0x90-0x97*/
03619 0xc0e5, 0x8587, 0xbaf1, 0x8588, 0x8589, 0xd8c8, 0x858a, 0xd4ad, /*0x98-0x9f*/
03620 0x858b, 0x858c, 0xcfe1, 0xd8c9, 0x858d, 0xd8ca, 0xcfc3, 0x858e, /*0xa0-0xaf*/
03621 0xb3f8, 0xbec7, 0x858f, 0x8590, 0x8591, 0x8592, 0xd8cb, 0x8593, /*0xa8-0xaf*/
03622 0x8594, 0x8595, 0x8596, 0x8597, 0x8598, 0x8599, 0xdbcc, 0x859a, /*0xb0-0xb7*/
03623 0x859b, 0x859c, 0x859d, 0xc8a5, 0x859e, 0x859f, 0x85a0, 0xcfd8, /*0xb8-0xbf*/
03624 0x85a1, 0xc8fe, 0xb2ce, 0x85a2, 0x85a3, 0x85a4, 0x85a5, 0x85a6, /*0xc0-0xc7*/
03625 0xd3d6, 0xb2e6, 0xbcb0, 0xd3d1, 0xcbaa, 0xb7b4, 0x85a7, 0x85a8, /*0xc8-0xcf*/
03626 0x85a9, 0xb7a2, 0x85aa, 0x85ab, 0xcae5, 0x85ac, 0xc8a1, 0xcadc, /*0xd0-0xd7*/
03627 0xb1e4, 0xd0f0, 0x85ad, 0xc5d1, 0x85ae, 0x85af, 0x85b0, 0xdbcc, /*0xd8-0xdf*/
03628 0xb5fe, 0x85b1, 0x85b2, 0xbfd4, 0xb9c5, 0xbec4, 0xc1ed, 0x85b3, /*0xe0-0xe7*/
03629 0xdfb6, 0xdfb5, 0xd6bb, 0xbdd0, 0xd5d9, 0xb0c8, 0xb6a3, 0xbfc9, /*0xe8-0xef*/
03630 0xc6ca, 0xdfb3, 0xcab7, 0xd3d2, 0x85b4, 0xd8cf, 0xd2b6, 0xbac5, /*0xf0-0xf7*/
03631 0xcbb6, 0xcbb6, 0x85b5, 0xdfb7, 0xb5f0, 0xdfb4, 0x85b6, 0x85b7, /*0xf8-0xff*/
03632 /* 0x5400 */
03633 0x85b8, 0xd3f5, 0x85b9, 0xb3d4, 0xb8f7, 0x85ba, 0xdfba, 0x85bb, /*0x00-0x07*/
03634 0xbacf, 0xbcaa, 0xb5f5, 0x85bc, 0xcdac, 0xc3fb, 0xbaf3, 0xc0f4, /*0x08-0x0f*/
03635 0xc6dc, 0xcff2, 0xdfb8, 0xcfc5, 0x85bd, 0xc2c0, 0xdfb9, 0xc2f0, /*0x10-0x17*/
03636 0x85be, 0x85bf, 0x85c0, 0xbefd, 0x85c1, 0xc1df, 0xcdcc, 0xd2f7, /*0x18-0x1f*/
03637 0xb7cd, 0xdfc1, 0x85c2, 0xdfc4, 0x85c3, 0x85c4, 0xb7f1, 0xb0c9, /*0x20-0x27*/
03638 0xb6d6, 0xb7d4, 0x85c5, 0xbaac, 0xc0cf, 0xbfd4, 0xcbb1, 0xc6f4, /*0x28-0x2f*/
03639 0x85c6, 0xd6a8, 0xdfc5, 0x85c7, 0xcce2, 0xb3b3, 0x85c8, 0x85c9, /*0x30-0x37*/
03640 0xcce7, 0xb4b5, 0x85ca, 0xccec, 0xbaf0, 0x85cb, 0xcce1, 0x85cc, /*0x38-0x3f*/
03641 0xd1bd, 0x85cd, 0x85ce, 0xdfc0, 0x85cf, 0x85d0, 0xb4f4, 0x85d1, /*0x40-0x47*/
03642 0xb3ca, 0x85d2, 0xb8e6, 0xdfb6, 0x85d3, 0x85d4, 0x85d5, 0x85d6, /*0x48-0x4f*/
03643 0xc4c5, 0x85d7, 0xdfbc, 0xdfbd, 0xdfbe, 0xc5bb, 0xdfbf, 0xdfc2, /*0x50-0x57*/
03644 0xd4b1, 0xdfc3, 0x85d8, 0xc7ba, 0xcdc8, 0x85d9, 0x85da, 0x85db, /*0x58-0x5f*/
03645 0x85dc, 0x85dd, 0xc4d8, 0x85de, 0xdfca, 0x85df, 0xdfcf, 0x85e0, /*0x60-0x67*/
03646 0xd6dc, 0x85e1, 0x85e2, 0x85e3, 0x85e4, 0x85e5, 0x85e6, 0x85e7, /*0x68-0x6f*/
03647 0x85e8, 0xdfc9, 0xdfda, 0xcceb, 0x85e9, 0xbac7, 0xdfce, 0xdfc8, /*0x70-0x77*/
03648 0xc5de, 0x85ea, 0x85eb, 0xc9eb, 0xbaf4, 0xc3fc, 0x85ec, 0x85ed, /*0x78-0x7f*/
03649 0xbcd7, 0x85ee, 0xdfc6, 0x85ef, 0xdfcd, 0x85f0, 0xc5d8, 0x85f1, /*0x80-0x87*/
03650 0x85f2, 0x85f3, 0x85f4, 0xd5a6, 0xbacd, 0x85f5, 0xbecf, 0xd3bd, /*0x88-0x8f*/
03651 0xb8c0, 0x85f6, 0xd6e4, 0x85f7, 0xdfc7, 0xb9be, 0xbfa7, 0x85f8, /*0x90-0x97*/
03652 0x85f9, 0xc1fc, 0xdfcb, 0xdfcc, 0x85fa, 0xdfd0, 0x85fb, 0x85fc, /*0x98-0x9f*/
03653 0x85fd, 0x85fe, 0x8600, 0xdfdb, 0xdfde, 0x8601, 0xdfd7, 0xdfd6, /*0xa0-0xaf*/
03654 0xd7c9, 0xdfec, 0xdfed, 0xe5eb, 0xd2a7, 0xdfd2, 0x8602, 0xbfa9, /*0xa8-0xaf*/
03655 0x8603, 0xd4db, 0x8604, 0xbfc8, 0xdfd4, 0x8605, 0x8606, 0x8607, 0x8608, /*0xb0-0xbf*/
03656 0xcfc6, 0x8609, 0x860a, 0xdfdd, 0x860b, 0xd1ca, 0x860c, 0xdfde, /*0xb8-0xbf*/
03657 0xb0a7, 0xc6b7, 0xdfd3, 0x860d, 0x860e, 0x860f, 0xb6d4, 0xcddb, /*0xc0-0xc7*/
03658 0xb9fe, 0xd4d5, 0x860f, 0xdfdf, 0xdfdf, 0xcfec, 0xb0a5, 0xdfef, /*0xc8-0xcf*/
03659 0xdfd1, 0xd1c6, 0xdfd5, 0xdfd8, 0xdfd9, 0xdfdc, 0x8650, 0xbba9, /*0xd0-0xd7*/
```

```

03660 0x8651, 0xdf0, 0xdf1, 0x8652, 0xdf2, 0xdf6, 0xdf8, 0xd3b4, /*0xd8-0xdf*/
03661 0x8653, 0x8654, 0x8655, 0x8656, 0x8657, 0xb8e7, 0xc5b6, 0xdf0, /*0xe0-0xe7*/
03662 0xc9da, 0xc1a8, 0xc4c4, 0x8658, 0x8659, 0xbfd0, 0xcff8, 0x865a, /*0xe8-0xef*/
03663 0x865b, 0x865c, 0xd5dc, 0xdfee, 0x865d, 0x865e, 0x865f, 0x8660, /*0xf0-0xf7*/
03664 0x8661, 0x8662, 0xb2b8, 0x8663, 0xbadf, 0xdfec, 0x8664, 0xdbc1, /*0xf8-0xff*/
03665 /* 0x5500 */
03666 0x8665, 0xd1e4, 0x8666, 0x8667, 0x8668, 0x8669, 0xcbf4, 0xb4bd, /*0x00-0x07*/
03667 0x866a, 0xb0a6, 0x866b, 0x866c, 0x866d, 0x866e, 0x866f, 0xdf1, /*0x08-0x0f*/
03668 0xcc6, 0xdf2, 0x8670, 0x8671, 0xdfed, 0x8672, 0x8673, 0x8674, /*0x10-0x17*/
03669 0x8675, 0x8676, 0x8677, 0xdf9, 0x8678, 0x8679, 0x867a, 0x867b, /*0x18-0x1f*/
03670 0xdf0, 0x867c, 0xdfef, 0xdf0, 0xbbbd, 0x867d, 0x867e, 0xdf3, /*0x20-0x27*/
03671 0x8680, 0x8681, 0xdf4, 0x8682, 0xbba3, 0x8683, 0xcadb, 0x8684, /*0x28-0x2f*/
03672 0xe0a7, 0xb3aa, 0x8684, 0xe0a6, 0x8685, 0x8686, 0x8687, 0xe0a1, /*0x30-0x37*/
03673 0x8688, 0x8689, 0x868a, 0x868b, 0xdf0, 0x868c, 0xcdd9, 0xdfc, /*0x38-0x3f*/
03674 0x868d, 0xdfa, 0x868e, 0xbfd0, 0xd7c4, 0x868f, 0xc9cc, 0x8690, /*0x40-0x47*/
03675 0x8691, 0xdf8, 0xb0a1, 0x8692, 0x8693, 0x8694, 0x8695, 0x8696, /*0x48-0x4f*/
03676 0xdfd, 0x8697, 0x8698, 0x8699, 0x869a, 0xdfb, 0xe0a2, 0x869b, /*0x50-0x57*/
03677 0x869c, 0x869d, 0x869e, 0x869f, 0xe0a8, 0x86a0, 0x86a1, 0x86a2, /*0x58-0x5f*/
03678 0x86a3, 0xb7c8, 0x86a4, 0x86a5, 0xc6a1, 0xc9b6, 0xc0b2, 0xdf5, /*0x60-0x67*/
03679 0x86a6, 0x86a7, 0xc5be, 0x86a8, 0xd8c4, 0xdf9, 0xc4f6, 0x86a9, /*0x68-0x6f*/
03680 0x86aa, 0x86ab, 0x86ac, 0x86ad, 0x86ae, 0xe0a3, 0xe0a4, 0xe0a5, /*0x70-0x77*/
03681 0xd0a5, 0x86af, 0x86b0, 0xe0b4, 0xcce4, 0x86b1, 0xe0b1, 0x86b2, /*0x78-0x7f*/
03682 0xbfa6, 0xe0af, 0xcce9, 0xe0ab, 0xc9c6, 0x86b3, 0x86b4, 0xc0ae, /*0x80-0x87*/
03683 0xe0ae, 0xbad, 0xbab0, 0xe0a9, 0x86b5, 0x86b6, 0x86b7, 0xdf6, /*0x88-0x8f*/
03684 0x86b8, 0xe0b3, 0x86b9, 0x86ba, 0xe0b8, 0x86bb, 0x86bc, 0x86bd, /*0x90-0x97*/
03685 0xb4ad, 0xe0b9, 0x86be, 0x86bf, 0xcfb2, 0xbac8, 0x86c0, 0xe0b0, /*0x98-0x9f*/
03686 0x86c1, 0x86c2, 0x86c3, 0x86c4, 0x86c5, 0x86c6, 0x86c7, 0xd0fa, /*0xa0-0xaf*/
03687 0x86c8, 0x86c9, 0x86ca, 0x86cb, 0x86cc, 0x86cd, 0x86ce, 0x86cf, /*0xab-0xaf*/
03688 0x86d0, 0xe0ac, 0x86d1, 0xd4fb, 0x86d2, 0xdf7, 0x86d3, 0xc5e7, /*0xb0-0xb7*/
03689 0x86d4, 0xe0ad, 0x86d5, 0xd3f7, 0x86d6, 0xe0b6, 0xe0b7, 0x86d7, /*0xb8-0xbf*/
03690 0x86d8, 0x86d9, 0x86da, 0x86db, 0xe0c4, 0xd0e1, 0x86dc, 0x86dd, /*0xc0-0xc7*/
03691 0x86de, 0xe0bc, 0x86df, 0x86e0, 0xe0c9, 0xe0ca, 0x86e1, 0x86e2, /*0xc8-0xcf*/
03692 0x86e3, 0xe0be, 0xe0aa, 0xc9a4, 0xe0c1, 0x86e4, 0xe0b2, 0x86e5, /*0xd0-0xd7*/
03693 0x86e6, 0x86e7, 0x86e8, 0x86e9, 0xcac8, 0xe0c3, 0x86ea, 0xe0b5, /*0xd8-0xdf*/
03694 0x86eb, 0xcce, 0x86ec, 0xcbe3, 0xe0cd, 0xe0c6, 0xe0c2, 0x86ed, /*0xe0-0xe7*/
03695 0xe0cb, 0x86ee, 0xe0ba, 0xe0bf, 0xe0c0, 0x86ef, 0x86f0, 0xe0c5, /*0xe8-0xef*/
03696 0x86f1, 0x86f2, 0xe0c7, 0xe0c8, 0x86f3, 0xe0cc, 0x86f4, 0xe0bb, /*0xf0-0xf7*/
03697 0x86f5, 0x86f6, 0x86f7, 0x86f8, 0x86f9, 0xcdb4, 0xe0d5, 0x86fa, /*0xf8-0xff*/
03698 /* 0x5600 */
03699 0xe0d6, 0xe0d2, 0x86fb, 0x86fc, 0x86fd, 0x86fe, 0x8740, 0x8741, /*0x00-0x07*/
03700 0xe0d0, 0xbcc, 0x8742, 0x8743, 0xe0d1, 0x8744, 0xb8c2, 0xd8c5, /*0x08-0x0f*/
03701 0x8745, 0x8746, 0x8747, 0x8748, 0x8749, 0x874a, 0x874b, 0x874c, /*0x10-0x17*/
03702 0xd0ea, 0x874d, 0x874e, 0xc2ef, 0x874f, 0x8750, 0xe0cf, 0xe0bd, /*0x18-0x1f*/
03703 0x8751, 0x8752, 0x8753, 0xe0d4, 0xe0d3, 0x8754, 0x8755, 0xe0d7, /*0x20-0x27*/
03704 0x8756, 0x8757, 0x8758, 0x8759, 0xe0dc, 0xe0d8, 0x875a, 0x875b, /*0x28-0x2f*/
03705 0x875c, 0xd6f6, 0xb3b0, 0x875d, 0xd7ec, 0x875e, 0xcbbb, 0x875f, /*0x30-0x37*/
03706 0x8760, 0xe0da, 0x8761, 0xcce, 0x8762, 0x8763, 0x8764, 0xbad9, /*0x38-0x3f*/
03707 0x8765, 0x8766, 0x8767, 0x8768, 0x8769, 0x876a, 0x876b, 0x876c, /*0x40-0x47*/
03708 0x876d, 0x876e, 0x876f, 0x8770, 0xe0e1, 0xe0dd, 0xd2ad, 0x8771, /*0x48-0x4f*/
03709 0x8772, 0x8773, 0x8774, 0x8775, 0xe0e2, 0x8776, 0x8777, 0xe0db, /*0x50-0x57*/
03710 0xe0d9, 0xe0df, 0x8778, 0x8779, 0xe0e0, 0x877a, 0x877b, 0x877c, /*0x58-0x5f*/
03711 0x877d, 0x877e, 0xe0de, 0x8780, 0xe0e4, 0x8781, 0x8782, 0x8783, /*0x60-0x67*/
03712 0xc6f7, 0xd8ac, 0xd4eb, 0xe0e6, 0xcac9, 0x8784, 0x8785, 0x8786, /*0x68-0x6f*/
03713 0x8787, 0xe0e5, 0x8788, 0x8789, 0x878a, 0x878b, 0xb8c1, 0x878c, /*0x70-0x77*/
03714 0x878d, 0x878e, 0x878f, 0xe0e7, 0xe0e8, 0x8790, 0x8791, 0x8792, /*0x78-0x7f*/
03715 0x8793, 0x8794, 0x8795, 0x8796, 0x8797, 0xe0e9, 0xe0e3, 0x8798, /*0x80-0x87*/
03716 0x8799, 0x879a, 0x879b, 0x879c, 0x879d, 0x879e, 0xbabf, 0xcce, /*0x88-0x8f*/
03717 0x879f, 0x87a0, 0x87a1, 0xe0ea, 0x87a2, 0x87a3, 0x87a4, 0x87a5, /*0x90-0x97*/
03718 0x87a6, 0x87a7, 0x87a8, 0x87a9, 0x87aa, 0x87ab, 0x87ac, 0x87ad, /*0x98-0x9f*/
03719 0x87ae, 0x87af, 0x87b0, 0xcff9, 0x87b1, 0x87b2, 0x87b3, 0x87b4, /*0xa0-0xaf*/
03720 0x87b5, 0x87b6, 0x87b7, 0x87b8, 0x87b9, 0x87ba, 0x87bb, 0xe0eb, /*0xab-0xaf*/
03721 0x87bc, 0x87bd, 0x87be, 0x87bf, 0x87c0, 0x87c1, 0x87c2, 0xc8c2, /*0xb0-0xb7*/
03722 0x87c3, 0x87c4, 0x87c5, 0x87c6, 0xbdc0, 0x87c7, 0x87c8, 0x87c9, /*0xb8-0xbf*/
03723 0x87ca, 0x87cb, 0x87cc, 0x87cd, 0x87ce, 0x87cf, 0x87d0, 0x87d1, /*0xc0-0xc7*/
03724 0x87d2, 0x87d3, 0xc4d2, 0x87d4, 0x87d5, 0x87d6, 0x87d7, 0x87d8, /*0xc8-0xcf*/
03725 0x87d9, 0x87da, 0x87db, 0x87dc, 0xe0ec, 0x87dd, 0x87de, 0xe0ed, /*0xd0-0xd7*/
03726 0x87df, 0x87e0, 0xc7f4, 0xcbe4, 0x87e1, 0xe0ee, 0xbdbd, 0xd8b6, /*0xd8-0xdf*/
03727 0xd2f2, 0xe0ef, 0xcde5, 0x87e2, 0xb6da, 0x87e3, 0x87e4, 0x87e5, /*0xe0-0xe7*/
03728 0x87e6, 0x87e7, 0x87e8, 0xe0f1, 0x87e9, 0xd4b0, 0x87ea, 0x87eb, /*0xe8-0xef*/
03729 0xc0a7, 0xb4d1, 0x87ec, 0x87ed, 0xc8a7, 0xe0f0, 0x87ee, 0x87ef, /*0xf0-0xf7*/
03730 0x87f0, 0xe0f2, 0xb9cc, 0x87f1, 0x87f2, 0xb9fa, 0xcdbc, 0xe0f3, /*0xf8-0xff*/
03731 /* 0x5700 */
03732 0x87f3, 0x87f4, 0x87f5, 0xc6d4, 0xe0f4, 0x87f6, 0xd4b2, 0x87f7, /*0x00-0x07*/
03733 0xc8a6, 0xe0f6, 0xe0f5, 0x87f8, 0x87f9, 0x87fa, 0x87fb, 0x87fc, /*0x08-0x0f*/
03734 0x87fd, 0x87fe, 0x8840, 0x8841, 0x8842, 0x8843, 0x8844, 0x8845, /*0x10-0x17*/
03735 0x8846, 0x8847, 0x8848, 0x8849, 0xe0f7, 0x884a, 0x884b, 0xcde1, /*0x18-0x1f*/
03736 0x884c, 0x884d, 0x884e, 0x884f, 0x8850, 0x8851, 0x8852, 0x8853, /*0x20-0x27*/
03737 0xd4da, 0xbdbd, 0xbdb9, 0x8853, 0xbdbd, 0xb9e7, 0xbdc, 0xbdd, /*0x28-0x2f*/
03738 0xb5d8, 0x8854, 0x8855, 0xbdbd, 0x8856, 0x8857, 0x8858, 0x8859, /*0x30-0x37*/
03739 0x885a, 0xbdbd, 0xb3a1, 0xbdbf, 0x885b, 0x885c, 0xbbf8, 0x885d, /*0x38-0x3f*/
03740 0xd6b7, 0x885e, 0xbde0, 0x885f, 0x8860, 0x8861, 0x8862, 0xbef9, /*0x40-0x47*/
03741 0x8863, 0x8864, 0xb7bb, 0x8865, 0xbdbd, 0xc8ae, 0xbfb2, 0xbbb5, /*0x48-0x4f*/
03742 0xd7f8, 0xbfd3, 0x8866, 0x8867, 0x8868, 0x8869, 0x886a, 0xbfe9, /*0x50-0x57*/
03743 0x886b, 0x886c, 0xbce1, 0xcbe3, 0xbde, 0xb0d3, 0xcce, 0xb7d8, /*0x58-0x5f*/
03744 0xd7b9, 0xc6c2, 0x886d, 0x886e, 0xc0a4, 0x886f, 0xcbe9, 0x8870, /*0x60-0x67*/
03745 0xbde7, 0xbde1, 0xc6ba, 0xbde3, 0x8871, 0xbde8, 0x8872, 0xc5f7, /*0x68-0x6f*/
03746 0x8873, 0x8874, 0x8875, 0xbdea, 0x8876, 0x8877, 0xbde9, 0xbfc0, /*0x70-0x77*/

```

```
03747 0x8878, 0x8879, 0x887a, 0xdbe6, 0xdbe5, 0x887b, 0x887c, 0x887d, /*0x78-0x7f*/
03748 0x887e, 0x8880, 0xb4b9, 0xc0ac, 0xc2a2, 0xdbe2, 0xdbe4, 0x8881, /*0x80-0x87*/
03749 0x8882, 0x8883, 0x8884, 0xd0cd, 0xdbed, 0x8885, 0x8886, 0x8887, /*0x88-0x8f*/
03750 0x8888, 0x8889, 0xc0dd, 0xdbf2, 0x888a, 0x888b, 0x888c, 0x888d, /*0x90-0x97*/
03751 0x888e, 0x888f, 0x8890, 0xb6e2, 0x8891, 0x8892, 0x8893, 0x8894, /*0x98-0x9f*/
03752 0xdbf3, 0xdbd2, 0xb9b8, 0xd4ab, 0xdbec, 0x8895, 0xbfd1, 0xdbf0, /*0xa0-0xa7*/
03753 0x8896, 0xdbd1, 0x8897, 0xb5e6, 0x8898, 0xdbeb, 0xbfe5, 0x8899, /*0xa8-0xaf*/
03754 0x889a, 0x889b, 0xdbee, 0x889c, 0xdbf1, 0x889d, 0x889e, 0x889f, /*0xb0-0xb7*/
03755 0xdbf9, 0x88a0, 0x88a1, 0x88a2, 0x88a3, 0x88a4, 0x88a5, 0x88a6, /*0xb8-0xbf*/
03756 0x88a7, 0x88a8, 0xb9a1, 0xb0a3, 0x88a9, 0x88aa, 0x88ab, 0x88ac, /*0xc0-0xc7*/
03757 0x88ad, 0x88ae, 0x88af, 0xc2f1, 0x88b0, 0x88b1, 0xb3c7, 0xdbef, /*0xc8-0xcf*/
03758 0x88b2, 0x88b3, 0xdbf8, 0x88b4, 0xc6d2, 0xdbf4, 0x88b5, 0x88b6, /*0xd0-0xd7*/
03759 0xdbf5, 0xdbf7, 0xdbf6, 0x88b7, 0x88b8, 0xdbfe, 0x88b9, 0xd3f2, /*0xd8-0xdf*/
03760 0xb2ba, 0x88ba, 0x88bb, 0x88bc, 0xdbfd, 0x88bd, 0x88be, 0x88bf, /*0xe0-0xef*/
03761 0x88c0, 0x88c1, 0x88c2, 0x88c3, 0x88c4, 0xdc4a, 0x88c5, 0xdbfb, /*0xe8-0xef*/
03762 0x88c6, 0x88c7, 0x88c8, 0x88c9, 0xdbfa, 0x88ca, 0x88cb, 0x88cc, /*0xf0-0xf7*/
03763 0xdbfc, 0xc5e0, 0xbbf9, 0x88cd, 0x88ce, 0xdc43, 0x88cf, 0x88d0, /*0xf8-0xff*/
03764 /* 0x5800 */
03765 0xdc45, 0x88d1, 0xccc3, 0x88d2, 0x88d3, 0x88d4, 0xb6d1, 0xddc0, /*0x00-0x07*/
03766 0x88d5, 0x88d6, 0x88d7, 0xdc41, 0x88d8, 0xdc42, 0x88d9, 0x88da, /*0x08-0x0f*/
03767 0x88db, 0xc7b5, 0x88dc, 0x88dd, 0x88de, 0xb6e9, 0x88df, 0x88e0, /*0x10-0x17*/
03768 0x88e1, 0xdc47, 0x88e2, 0x88e3, 0x88e4, 0x88e5, 0xdc46, 0x88e6, /*0x18-0x1f*/
03769 0xdc49, 0xb1a4, 0x88e7, 0x88e8, 0xb5cc, 0x88e9, 0x88ea, 0x88eb, /*0x20-0x27*/
03770 0x88ec, 0x88ed, 0xbfb0, 0x88ee, 0x88ef, 0x88f0, 0x88f1, 0x88f2, /*0x28-0x2f*/
03771 0xd1df, 0x88f3, 0x88f4, 0x88f5, 0x88f6, 0xb6c2, 0x88f7, 0x88f8, /*0x30-0x37*/
03772 0x88f9, 0x88fa, 0x88fb, 0x88fc, 0x88fd, 0x88fe, 0x8940, 0x8941, /*0x38-0x3f*/
03773 0x8942, 0x8943, 0x8944, 0x8945, 0xdc48, 0x8946, 0x8947, 0x8948, /*0x40-0x47*/
03774 0x8949, 0x894a, 0x894b, 0x894c, 0xcdfa, 0xebf3, 0x894d, 0x894e, /*0x48-0x4f*/
03775 0x894f, 0xcdbd, 0x8950, 0x8951, 0xcdbf, 0x8952, 0x8953, 0x8954, /*0x50-0x57*/
03776 0xcccl, 0x8955, 0x8956, 0x8957, 0x8958, 0x8959, 0xc8fb, 0x895a, /*0x58-0x5f*/
03777 0x895b, 0x895c, 0x895d, 0x895e, 0x895f, 0xdcaa, 0x8960, 0x8961, /*0x60-0x67*/
03778 0x8962, 0x8963, 0x8964, 0xccee, 0xdcab, 0x8965, 0x8966, 0x8967, /*0x68-0x6f*/
03779 0x8968, 0x8969, 0x896a, 0x896b, 0x896c, 0x896d, 0x896e, 0x896f, /*0x70-0x77*/
03780 0x8970, 0x8971, 0x8972, 0x8973, 0x8974, 0x8975, 0xdbd3, 0x8976, /*0x78-0x7f*/
03781 0xdc4f, 0xdcac, 0x8977, 0xbdb3, 0x8978, 0xcabf, 0x8979, 0x897a, /*0x80-0x87*/
03782 0x897b, 0xdcad, 0x897c, 0x897d, 0x897e, 0x8980, 0x8981, 0x8982, /*0x88-0x8f*/
03783 0x8983, 0x8984, 0xc9ca, 0xc4b9, 0x8985, 0x8986, 0x8987, 0x8988, /*0x90-0x97*/
03784 0x8989, 0xc7bd, 0xdcae, 0x898a, 0x898b, 0x898c, 0xd4f6, 0xd0e6, /*0x98-0x9f*/
03785 0x898d, 0x898e, 0xdcad, 0x898f, 0x8990, 0x8991, 0x8992, 0x8993, 0x8994, /*0xa0-0xa7*/
03786 0xc4ab, 0xb6d5, 0x8995, 0x8996, 0x8997, 0x8998, 0x8999, 0x899a, /*0xa8-0xaf*/
03787 0x899b, 0x899c, 0x899d, 0x899e, 0x899f, 0x89a0, 0x89a1, 0x89a2, /*0xb0-0xb7*/
03788 0x89a3, 0x89a4, 0x89a5, 0x89a6, 0xdbd4, 0x89a7, 0x89a8, 0x89a9, /*0xb8-0xbf*/
03789 0x89aa, 0xb1da, 0x89ab, 0x89ac, 0x89ad, 0xdbd5, 0x89ae, 0x89af, /*0xc0-0xc7*/
03790 0x89b0, 0x89b1, 0x89b2, 0x89b3, 0x89b4, 0x89b5, 0x89b6, 0x89b7, /*0xc8-0xcf*/
03791 0x89b8, 0xdbd6, 0x89b9, 0x89ba, 0x89bb, 0xbabe, 0x89bc, 0x89bd, /*0xd0-0xd7*/
03792 0x89be, 0x89bf, 0x89c0, 0x89c1, 0x89c2, 0x89c3, 0x89c4, 0x89c5, /*0xd8-0xdf*/
03793 0x89c6, 0x89c7, 0x89c8, 0x89c9, 0xc8c0, 0x89ca, 0x89cb, 0x89cc, /*0xe0-0xef*/
03794 0x89cd, 0x89ce, 0x89cf, 0xcabf, 0xc8c9, 0x89d0, 0xd7b3, 0x89d1, /*0xe8-0xef*/
03795 0xc9f9, 0x89d2, 0x89d3, 0xbfc7, 0x89d4, 0x89d5, 0xbaf8, 0x89d6, /*0xf0-0xf7*/
03796 0x89d7, 0xd2bc, 0x89d8, 0x89d9, 0x89da, 0x89db, 0x89dc, 0x89dd, /*0xf8-0xff*/
03797 /* 0x5900 */
03798 0x89de, 0x89df, 0xe2ba, 0x89e0, 0xb4a6, 0x89e1, 0x89e2, 0xb1b8, /*0x00-0x07*/
03799 0x89e3, 0x89e4, 0x89e5, 0x89e6, 0x89e7, 0xb8b4, 0x89e8, 0xcfc4, /*0x08-0x0f*/
03800 0x89e9, 0x89ea, 0x89eb, 0x89ec, 0xd9e7, 0xcfa6, 0xcde2, 0x89ed, /*0x10-0x17*/
03801 0x89ee, 0xd9ed, 0xb6e0, 0x89ef, 0xd2b9, 0x89f0, 0x89f1, 0xb9bb, /*0x18-0x1f*/
03802 0x89f2, 0x89f3, 0x89f4, 0x89f5, 0xe2b9, 0xe2b7, 0x89f6, 0xb4f3, /*0x20-0x27*/
03803 0x89f7, 0xccce, 0xcabf, 0xb7f2, 0x89f8, 0xd8b2, 0xd1eb, 0xbabb, /*0x28-0x2f*/
03804 0x89f9, 0xcaa7, 0x89fa, 0x89fb, 0xcdb7, 0x89fc, 0x89fd, 0xd2c4, /*0x30-0x37*/
03805 0xbfe4, 0xbcd0, 0xb6e1, 0x89fe, 0xdec5, 0x8a40, 0x8a41, 0x8a42, /*0x38-0x3f*/
03806 0x8a43, 0xdec6, 0xdbbc, 0x8a44, 0xd1d9, 0x8a45, 0x8a46, 0xc6e6, /*0x40-0x47*/
03807 0xc4ce, 0xb7ee, 0x8a47, 0xb7dc, 0x8a48, 0x8a49, 0xbffc, 0xd7e0, /*0x48-0x4f*/
03808 0x8a4a, 0xc6f5, 0x8a4b, 0x8a4c, 0xb1bc, 0xdce8, 0xbdb1, 0xccd7, /*0x50-0x57*/
03809 0xdec9, 0x8a4d, 0xdec9, 0x8a4e, 0x8a4f, 0x8a50, 0x8a51, 0x8a52, /*0x58-0x5f*/
03810 0xb5ec, 0x8a53, 0xc9dd, 0x8a54, 0x8a55, 0xb0c2, 0x8a56, 0x8a57, /*0x60-0x67*/
03811 0x8a58, 0x8a59, 0x8a5a, 0x8a5b, 0x8a5c, 0x8a5d, 0x8a5e, 0x8a5f, /*0x68-0x6f*/
03812 0x8a60, 0x8a61, 0x8a62, 0xc5ae, 0xc5ab, 0x8a63, 0xc4cc, 0x8a64, /*0x70-0x77*/
03813 0xbce9, 0xcbf9, 0x8a65, 0x8a66, 0x8a67, 0xbac3, 0x8a68, 0x8a69, /*0x78-0x7f*/
03814 0x8a6a, 0xe5f9, 0xc8e7, 0xe5fa, 0xcdfd, 0x8a6b, 0xd7b1, 0xb8be, /*0x80-0x87*/
03815 0xc2e8, 0x8a6c, 0xc8d1, 0x8a6d, 0x8a6e, 0xe5fb, 0x8a6f, 0x8a70, /*0x88-0x8f*/
03816 0x8a71, 0x8a72, 0xb6ca, 0xbccb, 0x8a73, 0x8a74, 0xd1fd, 0xe6a1, /*0x90-0x97*/
03817 0x8a75, 0xc3ee, 0x8a76, 0x8a77, 0x8a78, 0x8a79, 0xe6a4, 0x8a7a, /*0x98-0x9f*/
03818 0x8a7b, 0x8a7c, 0x8a7d, 0xe5fe, 0xe6a5, 0xcdd7, 0x8a7e, 0x8a80, /*0xa0-0xaf*/
03819 0xb7c1, 0xe5fc, 0xe5fd, 0xe6a3, 0x8a81, 0x8a82, 0xc4dd, 0xe6a8, /*0xa8-0xaf*/
03820 0x8a83, 0x8a84, 0xe6a7, 0x8a85, 0x8a86, 0x8a87, 0x8a88, 0x8a89, /*0xb0-0xbf*/
03821 0x8a8a, 0xc3c3, 0x8a8b, 0xc6de, 0x8a8c, 0x8a8d, 0xe6aa, 0x8a8e, /*0xb8-0xbf*/
03822 0x8a8f, 0x8a90, 0x8a91, 0x8a92, 0x8a93, 0x8a94, 0xc4b7, 0x8a95, /*0xc0-0xc7*/
03823 0x8a96, 0x8a97, 0xe6a2, 0xcabf, 0x8a98, 0x8a99, 0x8a9a, 0x8a9b, /*0xc8-0xcf*/
03824 0xbde3, 0xb9c3, 0xe6a6, 0xd0d5, 0xceaf, 0x8a9c, 0x8a9d, 0xe6a9, /*0xd0-0xd7*/
03825 0xe6b0, 0x8a9e, 0xd2a6, 0x8a9f, 0xbdaa, 0xe6ad, 0x8aa0, 0x8aa1, /*0xd8-0xdf*/
03826 0x8aa2, 0x8aa3, 0x8aa4, 0xe6af, 0x8aa5, 0xc0d1, 0x8aa6, 0x8aa7, /*0xe0-0xef*/
03827 0xd2cc, 0x8aa8, 0x8aa9, 0x8aaa, 0xbca7, 0x8aab, 0x8aac, 0x8aad, /*0xe8-0xef*/
03828 0x8aae, 0x8aaf, 0x8ab0, 0x8ab1, 0x8ab2, 0x8ab3, 0x8ab4, 0x8ab5, /*0xf0-0xf7*/
03829 0x8ab6, 0xe6b1, 0x8ab7, 0xd2f6, 0x8ab8, 0x8ab9, 0x8aba, 0xd7cb, /*0xf8-0xff*/
03830 /* 0x5a00 */
03831 0x8abb, 0xcdfe, 0x8abc, 0xcdde, 0xc2a6, 0xe6ab, 0xe6ac, 0xbdbf, /*0x00-0x07*/
03832 0xe6ae, 0xe6b3, 0x8abd, 0x8abe, 0xe6b2, 0x8abf, 0x8ac0, 0x8ac1, /*0x08-0x0f*/
03833 0x8ac2, 0xe6b6, 0x8ac3, 0xe6b8, 0x8ac4, 0x8ac5, 0x8ac6, 0x8ac7, /*0x10-0x17*/
```

```

03834 0xc4ef, 0x8ac8, 0x8ac9, 0x8aca, 0xc4c8, 0x8acb, 0x8acc, 0xbeea, /*x18-0x1f*/
03835 0xc9ef, 0x8acd, 0x8ace, 0xe6b7, 0x8acf, 0xb6f0, 0x8ad0, 0x8ad1, /*x20-0x27*/
03836 0x8ad2, 0xc3e4, 0x8ad3, 0x8ad4, 0x8ad5, 0x8ad6, 0x8ad7, 0x8ad8, /*x28-0x2f*/
03837 0x8ad9, 0xd3e9, 0xe6b4, 0x8ada, 0xe6b5, 0x8adb, 0xc8a2, 0x8adc, /*x30-0x37*/
03838 0x8add, 0x8ade, 0x8adf, 0x8ae0, 0xe6bd, 0x8ae1, 0x8ae2, 0x8ae3, /*x38-0x3f*/
03839 0xe6b9, 0x8ae4, 0x8ae5, 0x8ae6, 0x8ae7, 0x8ae8, 0xc6c5, 0x8ae9, /*x40-0x47*/
03840 0x8aea, 0xcdf1, 0xe6bb, 0x8aeb, 0x8aec, 0x8aed, 0x8aee, 0x8aef, /*x48-0x4f*/
03841 0x8af0, 0x8af1, 0x8af2, 0x8af3, 0x8af4, 0xe6bc, 0x8af5, 0x8af6, /*x50-0x57*/
03842 0x8af7, 0x8af8, 0xbbe9, 0x8af9, 0x8afa, 0x8afb, 0x8afc, 0x8afd, /*x58-0x5f*/
03843 0x8afe, 0x8b40, 0xe6be, 0x8b41, 0x8b42, 0x8b43, 0x8b44, 0xe6ba, /*x60-0x67*/
03844 0x8b45, 0x8b46, 0xc0b7, 0x8b47, 0x8b48, 0x8b49, 0x8b4a, 0x8b4b, /*x68-0x6f*/
03845 0x8b4c, 0x8b4d, 0x8b4e, 0x8b4f, 0xd3a4, 0xe6bf, 0xc9f4, 0xe6c3, /*x70-0x77*/
03846 0x8b50, 0x8b51, 0xe6c4, 0x8b52, 0x8b53, 0x8b54, 0x8b55, 0xd0f6, /*x78-0x7f*/
03847 0x8b56, 0x8b57, 0x8b58, 0x8b59, 0x8b5a, 0x8b5b, 0x8b5c, 0x8b5d, /*x80-0x87*/
03848 0x8b5e, 0x8b5f, 0x8b60, 0x8b61, 0x8b62, 0x8b63, 0x8b64, 0x8b65, /*x88-0x8f*/
03849 0x8b66, 0x8b67, 0xc3bd, 0x8b68, 0x8b69, 0x8b6a, 0x8b6b, 0x8b6c, /*x90-0x97*/
03850 0x8b6d, 0x8b6e, 0xc3c4, 0xe6c2, 0x8b6f, 0x8b70, 0x8b71, 0x8b72, /*x98-0x9f*/
03851 0x8b73, 0x8b74, 0x8b75, 0x8b76, 0x8b77, 0x8b78, 0x8b79, 0x8b7a, /*xa0-0xaf*/
03852 0x8b7b, 0x8b7c, 0xe6c1, 0x8b7d, 0x8b7e, 0x8b80, 0x8b81, 0x8b82, /*xa8-0xaf*/
03853 0x8b83, 0x8b84, 0xe6c7, 0xcfb1, 0x8b85, 0xebf4, 0x8b86, 0x8b87, /*xb0-0xb7*/
03854 0xe6ca, 0x8b88, 0x8b89, 0x8b8a, 0x8b8b, 0x8b8c, 0xe6c5, 0x8b8d, /*xb8-0xbf*/
03855 0x8b8e, 0xbcd, 0xc9a9, 0x8b8f, 0x8b90, 0x8b91, 0x8b92, 0x8b93, /*xc0-0xc7*/
03856 0x8b94, 0xbcb5, 0x8b95, 0x8b96, 0xcfd3, 0x8b97, 0x8b98, 0x8b99, /*xc8-0xcf*/
03857 0x8b9a, 0x8b9b, 0xe6c8, 0x8b9c, 0xe6c9, 0x8b9d, 0xe6ce, 0x8b9e, /*xd0-0xdf*/
03858 0xe6d0, 0x8b9f, 0x8ba0, 0x8ba1, 0xe6d1, 0x8ba2, 0x8ba3, 0x8ba4, /*xd8-0xdf*/
03859 0xe6cb, 0xb5d5, 0x8ba5, 0xe6cc, 0x8ba6, 0x8ba7, 0xe6cd, 0x8ba8, /*xe0-0xe7*/
03860 0x8ba9, 0xc4db, 0x8baa, 0xe6c6, 0x8bab, 0x8bac, 0x8bad, 0x8bae, /*xe8-0xef*/
03861 0x8baf, 0xe6cd, 0x8bb0, 0x8bb1, 0x8bb2, 0x8bb3, 0x8bb4, 0x8bb5, /*xf0-0xf7*/
03862 0x8bb6, 0x8bb7, 0x8bb8, 0x8bb9, 0x8bba, 0x8bbb, 0x8bbc, 0x8bbd, /*xf8-0xff*/
03863 /* 0x5b00 */
03864 0x8bbe, 0x8bbf, 0x8bc0, 0x8bc1, 0x8bc2, 0x8bc3, 0x8bc4, 0x8bc5, /*x00-0x07*/
03865 0x8bc6, 0xe6d2, 0x8bc7, 0x8bc8, 0x8bc9, 0x8bca, 0x8bcb, 0x8bcc, /*x08-0x0f*/
03866 0x8bcd, 0x8bce, 0x8bcf, 0x8bd0, 0x8bd1, 0x8bd2, 0xe6d4, 0xe6d3, /*x10-0x17*/
03867 0x8bd3, 0x8bd4, 0x8bd5, 0x8bd6, 0x8bd7, 0x8bd8, 0x8bd9, 0x8bda, /*x18-0x1f*/
03868 0x8bdb, 0x8bdc, 0x8bdd, 0x8bde, 0x8bdf, 0x8be0, 0x8be1, 0x8be2, /*x20-0x27*/
03869 0x8be3, 0x8be4, 0x8be5, 0x8be6, 0x8be7, 0x8be8, 0x8be9, 0x8bea, /*x28-0x2f*/
03870 0x8beb, 0x8bec, 0xe6d5, 0x8bed, 0xd9f8, 0x8bee, 0x8bef, 0xe6d6, /*x30-0x37*/
03871 0x8bf0, 0x8bf1, 0x8bf2, 0x8bf3, 0x8bf4, 0x8bf5, 0x8bf6, 0x8bf7, /*x38-0x3f*/
03872 0xe6d7, 0x8bf8, 0x8bf9, 0x8bfa, 0x8bfb, 0x8bfc, 0x8bfd, 0x8bfe, /*x40-0x47*/
03873 0x8c40, 0x8c41, 0x8c42, 0x8c43, 0x8c44, 0x8c45, 0x8c46, 0x8c47, /*x48-0x4f*/
03874 0xd7d3, 0xe6dd, 0x8c48, 0xe6de, 0xbfd7, 0xd4d0, 0x8c49, 0xd7d6, /*x50-0x57*/
03875 0xb4e6, 0xcbe, 0xe6da, 0xd8c3, 0xd7ce, 0xd0a2, 0x8c4a, 0xc3cf, /*x58-0x5f*/
03876 0x8c4b, 0x8c4c, 0xe6df, 0xbcb, 0xb9c2, 0xe6db, 0xd1a7, 0x8c4d, /*x60-0x67*/
03877 0x8c4e, 0xbaa2, 0xc2cf, 0x8c4f, 0xd8ab, 0x8c50, 0x8c51, 0x8c52, /*x68-0x6f*/
03878 0xcab, 0xe6ee, 0x8c53, 0xe6dc, 0x8c54, 0xb7f5, 0x8c55, 0x8c56, /*x70-0x77*/
03879 0x8c57, 0x8c58, 0xc8e6, 0x8c59, 0x8c5a, 0xc4f5, 0x8c5b, 0x8c5c, /*x78-0x7f*/
03880 0xe5b2, 0xc4fe, 0x8c5d, 0xcbf, 0xe5b3, 0xd5ac, 0x8c5e, 0xd3ee, /*x80-0x87*/
03881 0xcad8, 0xb0b2, 0x8c5f, 0xcbe, 0xcdea, 0x8c60, 0x8c61, 0xbaea, /*x88-0x8f*/
03882 0x8c62, 0x8c63, 0x8c64, 0xe5b5, 0x8c65, 0xe5b4, 0x8c66, 0xd7da, /*x90-0x97*/
03883 0xb9d9, 0xd6e6, 0xb6a8, 0xcdf0, 0xd2cb, 0xb1a6, 0xcab5, 0x8c67, /*x98-0x9f*/
03884 0xb3e8, 0xc9f3, 0xbfc, 0xd0fb, 0xcad2, 0xe5b6, 0xbbc2, 0x8c68, /*xa0-0xaf*/
03885 0x8c69, 0x8c6a, 0xcfdc, 0xb9ac, 0x8c6b, 0x8c6c, 0x8c6d, 0x8c6e, /*xa8-0xaf*/
03886 0xd4d7, 0x8c6f, 0x8c70, 0xbaa6, 0xd1e7, 0xcffc, 0xbcd2, 0x8c71, /*xb0-0xb7*/
03887 0xe5b7, 0xc8dd, 0x8c72, 0x8c73, 0x8c74, 0xbfed, 0xb1f6, 0xcbe, /*xb8-0xbf*/
03888 0x8c75, 0x8c76, 0xbcc5, 0x8c77, 0xbcc4, 0xd2fa, 0xc3dc, 0xbfdc, /*xc0-0xc7*/
03889 0x8c78, 0x8c79, 0x8c7a, 0x8c7b, 0xb8bb, 0x8c7c, 0x8c7d, 0x8c7e, /*xc8-0xcf*/
03890 0xc3c2, 0x8c80, 0xbaae, 0xd4a2, 0x8c81, 0x8c82, 0x8c83, 0x8c84, /*xd0-0xdf*/
03891 0x8c85, 0x8c86, 0x8c87, 0x8c88, 0x8c89, 0xc7de, 0xc4af, 0xb2ec, /*xd8-0xdf*/
03892 0x8c8a, 0xb9d1, 0x8c8b, 0x8c8c, 0xe5bb, 0xc1c8, 0x8c8d, 0x8c8e, /*xe0-0xe7*/
03893 0xd5af, 0xc8f8, 0x8c90, 0x8c91, 0x8c92, 0x8c93, 0xe5bc, 0x8c94, /*xe8-0xef*/
03894 0xe5be, 0x8c95, 0x8c96, 0x8c97, 0x8c98, 0x8c99, 0x8c9a, 0x8c9b, /*xf0-0xf7*/
03895 0xb4e7, 0xb6d4, 0xcbe2, 0xd1b0, 0xb5bc, 0x8c9c, 0x8c9d, 0xcad9, /*xf8-0xff*/
03896 /* 0x5c00 */
03897 0x8c9e, 0xb7e2, 0x8c9f, 0x8ca0, 0xc9e4, 0x8ca1, 0xbdab, 0x8ca2, /*x00-0x07*/
03898 0x8ca3, 0xcbe, 0xd7f0, 0x8ca4, 0x8ca5, 0x8ca6, 0x8ca7, 0xd0a1, /*x08-0x0f*/
03899 0x8ca8, 0xc9d9, 0x8caa, 0xb6fb, 0xe6d8, 0xbce2, 0x8cab, /*x10-0x17*/
03900 0xb3be, 0x8cac, 0xc9d0, 0x8cad, 0xe6d9, 0xb3a2, 0x8cae, 0x8caf, /*x18-0x1f*/
03901 0x8cb0, 0x8cb1, 0xdecc, 0x8cb2, 0xd3c8, 0xdec, 0x8cb3, 0xd2a2, /*x20-0x27*/
03902 0x8cb4, 0x8cb5, 0x8cb6, 0x8cb7, 0xdec, 0x8cb8, 0x8cb9, 0x8cba, /*x28-0x2f*/
03903 0x8cbb, 0xbec, 0x8cbc, 0x8cbd, 0xdec, 0x8cbe, 0x8cbf, 0x8cc0, /*x30-0x37*/
03904 0xcaac, 0xd2fc, 0xb3df, 0xe5ea, 0xc4e1, 0x8cbf, 0xcbe2, 0xc4f2, /*x38-0x3f*/
03905 0xbbed, 0xc6a8, 0xb2e3, 0x8cc1, 0x8cc2, 0xbbed, 0x8cc3, 0x8cc4, /*x40-0x47*/
03906 0xc7fc, 0xcce, 0xbdec, 0xcdd, 0x8cc5, 0x8cc6, 0xcaba, 0xc6c1, /*x48-0x4f*/
03907 0xe5ec, 0xd0bc, 0x8cc7, 0x8cc8, 0x8cc9, 0xd5b9, 0x8cca, 0x8ccb, /*x50-0x57*/
03908 0x8ccc, 0xe5ed, 0x8ccd, 0x8cce, 0x8ccf, 0x8cd0, 0xc4f4, 0x8cd1, /*x58-0x5f*/
03909 0xcdc0, 0xc2c5, 0x8cd2, 0xe5ef, 0x8cd3, 0xc2c4, 0xe5f0, 0x8cd4, /*x60-0x67*/
03910 0x8cd5, 0x8cd6, 0x8cd7, 0x8cd8, 0x8cd9, 0x8cda, 0xe5f8, 0xcdcd, /*x68-0x6f*/
03911 0x8cdb, 0xc9bd, 0x8cdc, 0x8cdd, 0x8cde, 0x8cdf, 0x8ce0, 0x8cel, /*x70-0x77*/
03912 0x8ce2, 0xd2d9, 0xela8, 0x8ce3, 0x8ce4, 0x8ce5, 0x8ce6, 0xd3ec, /*x78-0x7f*/
03913 0x8ce7, 0xcbea, 0xc6f1, 0x8ce8, 0x8ce9, 0x8cea, 0x8ceb, 0x8cec, /*x80-0x8f*/
03914 0xela, 0x8ced, 0x8cee, 0x8cef, 0xela7, 0xela9, 0xc4f0, 0x8cf1, /*x88-0x8f*/
03915 0xela, 0xela, 0x8cf2, 0x8cf3, 0xb2ed, 0x8cf4, 0xela, 0xb8da, /*x90-0x97*/
03916 0xela, 0xela, 0xela, 0xela, 0xela, 0xela, 0xela, 0xela, /*x98-0x9f*/
03917 0x8cf8, 0x8cf9, 0xela, 0xela, 0xela, 0xela, 0xela, 0xela, /*xa0-0xaf*/
03918 0x8cfe, 0xd1d2, 0xela, 0xela, 0xela, 0xela, 0xela, 0xela, /*xa8-0xaf*/
03919 0x8d43, 0xela, 0xela, 0xela, 0xela, 0xela, 0xela, 0xela, /*xb0-0xbf*/
03920 0xb0b6, 0x8d47, 0x8d48, 0x8d49, 0x8d4a, 0xela, 0xela, 0xbff9, /*xb8-0xbf*/

```

```
03921 0x8d4c, 0xe1b9, 0x8d4d, 0x8d4e, 0xe1bb, 0x8d4f, 0x8d50, 0x8d51, /*0xc0-0xc7*/
03922 0x8d52, 0x8d53, 0x8d54, 0xe1be, 0x8d55, 0x8d56, 0x8d57, 0x8d58, /*0xc8-0xcf*/
03923 0x8d59, 0x8d5a, 0xe1bc, 0x8d5b, 0x8d5c, 0x8d5d, 0x8d5e, 0x8d5f, /*0xd0-0xd7*/
03924 0x8d60, 0xd6c5, 0x8d61, 0x8d62, 0x8d63, 0x8d64, 0x8d65, 0x8d66, /*0xd8-0xdf*/
03925 0x8d67, 0xcfbf, 0x8d68, 0x8d69, 0xe1bd, 0xe1bf, 0xc2cd, 0x8d6a, /*0xe0-0xe7*/
03926 0xb6eb, 0x8d6b, 0xd3f8, 0x8d6c, 0xc7cd, 0x8d6e, 0x8d6f, /*0xe8-0xef*/
03927 0xb7e5, 0x8d70, 0x8d71, 0x8d72, 0x8d73, 0x8d74, 0x8d75, 0x8d76, /*0xf0-0xf7*/
03928 0x8d77, 0x8d78, 0x8d79, 0xbefe, 0x8d7a, 0x8d7b, 0x8d7c, 0x8d7d, /*0xf8-0xff*/
03929 /* 0x5d00 */
03930 0x8d7e, 0x8d80, 0xe1c0, 0xe1c1, 0x8d81, 0x8d82, 0xe1c7, 0xb3e7, /*0x00-0x07*/
03931 0x8d83, 0x8d84, 0x8d85, 0x8d86, 0x8d87, 0x8d88, 0xc6e9, 0x8d89, /*0x08-0x0f*/
03932 0x8d8a, 0x8d8b, 0x8d8c, 0x8d8d, 0xb4de, 0x8d8e, 0xd1c2, 0x8d8f, /*0x10-0x17*/
03933 0x8d90, 0x8d91, 0x8d92, 0xe1c8, 0x8d93, 0x8d94, 0xe1c6, 0x8d95, /*0x18-0x1f*/
03934 0x8d96, 0x8d97, 0x8d98, 0x8d99, 0xe1c5, 0x8d9a, 0xe1c3, 0xe1c2, /*0x20-0x27*/
03935 0x8d9b, 0xb1c0, 0x8d9c, 0x8d9d, 0x8d9e, 0xd5b8, 0xe1c4, 0x8d9f, /*0x28-0x2f*/
03936 0x8da0, 0x8da1, 0x8da2, 0x8da3, 0xe1cb, 0x8da4, 0x8da5, 0x8da6, /*0x30-0x37*/
03937 0x8da7, 0x8da8, 0x8da9, 0x8daa, 0x8dab, 0xe1cc, 0xe1ca, 0x8dac, /*0x38-0x3f*/
03938 0x8dad, 0x8dae, 0x8daf, 0x8db0, 0x8db1, 0x8db2, 0x8db3, 0xeffa, /*0x40-0x47*/
03939 0x8db4, 0x8db5, 0xe1d3, 0xe1d2, 0xc7b6, 0x8db6, 0x8db7, 0x8db8, /*0x48-0x4f*/
03940 0x8db9, 0x8dba, 0x8dbb, 0x8dbc, 0x8dbd, 0x8dbe, 0x8dbf, 0x8dc0, /*0x50-0x57*/
03941 0xe1c9, 0x8dc1, 0x8dc2, 0xe1ce, 0x8dc3, 0xe1d0, 0x8dc4, 0x8dc5, /*0x58-0x5f*/
03942 0x8dc6, 0x8dc7, 0x8dc8, 0x8dc9, 0x8dca, 0x8dcb, 0x8dcc, 0x8dcd, /*0x60-0x67*/
03943 0x8dce, 0xe1d4, 0x8dcf, 0xe1d1, 0xe1cd, 0x8dd0, 0x8dd1, 0xe1cf, /*0x68-0x6f*/
03944 0x8dd2, 0x8dd3, 0x8dd4, 0x8dd5, 0xe1d5, 0x8dd6, 0x8dd7, 0x8dd8, /*0x70-0x77*/
03945 0x8dd9, 0x8dda, 0x8ddb, 0x8ddc, 0x8ddd, 0x8dde, 0x8ddf, 0x8de0, /*0x78-0x7f*/
03946 0x8de1, 0x8de2, 0xe1d6, 0x8de3, 0x8de4, 0x8de5, 0x8de6, 0x8de7, /*0x80-0x87*/
03947 0x8de8, 0x8de9, 0x8dea, 0x8deb, 0x8dec, 0x8ded, 0x8dee, 0x8def, /*0x88-0x8f*/
03948 0x8df0, 0x8df1, 0x8df2, 0x8df3, 0x8df4, 0x8df5, 0x8df6, 0x8df7, /*0x90-0x97*/
03949 0x8df8, 0xe1d7, 0x8df9, 0x8dfa, 0x8dfb, 0xe1d8, 0x8dfc, 0x8dfd, /*0x98-0x9f*/
03950 0x8dfe, 0x8e40, 0x8e41, 0x8e42, 0x8e43, 0x8e44, 0x8e45, 0x8e46, /*0xa0-0xaf*/
03951 0x8e47, 0x8e48, 0x8e49, 0x8e4a, 0x8e4b, 0x8e4c, 0x8e4d, 0x8e4e, /*0xa8-0xaf*/
03952 0x8e4f, 0x8e50, 0x8e51, 0x8e52, 0x8e53, 0x8e54, 0x8e55, 0xe1da, /*0xb0-0xb7*/
03953 0x8e56, 0x8e57, 0x8e58, 0x8e59, 0x8e5a, 0x8e5b, 0x8e5c, 0x8e5d, /*0xb8-0xbf*/
03954 0x8e5e, 0x8e5f, 0x8e60, 0x8e61, 0x8e62, 0xe1db, 0x8e63, 0x8e64, /*0xc0-0xc7*/
03955 0x8e65, 0x8e66, 0x8e67, 0x8e68, 0x8e69, 0xcea1, 0x8e6a, 0x8e6b, /*0xc8-0xcf*/
03956 0x8e6c, 0x8e6d, 0x8e6e, 0x8e6f, 0x8e70, 0x8e71, 0x8e72, 0x8e73, /*0xd0-0xd7*/
03957 0x8e74, 0x8e75, 0x8e76, 0xe7dd, 0x8e77, 0xb4a8, 0xd6dd, 0x8e78, /*0xd8-0xdf*/
03958 0x8e79, 0xd1b2, 0xb3b2, 0x8e7a, 0x8e7b, 0xb9a4, 0xd7f3, 0xc7c9, /*0xe0-0xe7*/
03959 0xbded, 0xb9ae, 0x8e7c, 0xc7d7, 0x8e7e, 0xb2ee, 0xdbcf, /*0xe8-0xef*/
03960 0x8e80, 0xbcbba, 0xd2d1, 0xc7c8, 0xb0cd, 0x8e81, 0x8e82, 0xc7ef, /*0xf0-0xf7*/
03961 0x8e83, 0x8e84, 0x8e85, 0x8e86, 0x8e87, 0xd9e3, 0xbded, 0x8e88, /*0xf8-0xff*/
03962 /* 0x5e00 */
03963 0x8e89, 0xb1d2, 0xcad0, 0xb2bc, 0x8e8a, 0xcba7, 0xb7ab, 0x8e8b, /*0x00-0x07*/
03964 0xcaca, 0x8e8c, 0x8e8d, 0x8e8e, 0xcfa3, 0x8e8f, 0x8e90, 0xe0f8, /*0x08-0x0f*/
03965 0xd5ca, 0xe0fb, 0x8e91, 0x8e92, 0xe0fa, 0xc5c1, 0xc7fb, 0x8e93, /*0x10-0x17*/
03966 0xc1b1, 0xe0f9, 0xd6e3, 0xb2af, 0xd6c4, 0xb5db, 0x8e94, 0x8e95, /*0x18-0x1f*/
03967 0x8e96, 0x8e97, 0x8e98, 0x8e99, 0x8e9a, 0x8e9b, 0xb4f8, 0xd6a1, /*0x20-0x27*/
03968 0x8e9c, 0x8e9d, 0x8e9e, 0x8e9f, 0x8ea0, 0xc7af, 0xb0ef, 0x8ea1, /*0x28-0x2f*/
03969 0x8ea2, 0xe0fc, 0x8ea3, 0x8ea4, 0x8ea5, 0x8ea6, 0x8ea7, 0xe1a1, /*0x30-0x37*/
03970 0xb3a3, 0x8ea8, 0x8ea9, 0xe0fd, 0xe0fe, 0xc3b1, 0x8eaa, 0x8eab, /*0x38-0x3f*/
03971 0x8eac, 0x8ead, 0xc3dd, 0x8eae, 0xe1a2, 0xb7f9, 0x8eaf, 0x8eb0, /*0x40-0x47*/
03972 0x8eb1, 0x8eb2, 0x8eb3, 0x8eb4, 0xbbcf, 0x8eb5, 0x8eb6, 0x8eb7, /*0x48-0x4f*/
03973 0x8eb8, 0x8eb9, 0x8eba, 0x8ebb, 0xe1a3, 0xc4bb, 0x8ebc, 0x8ebd, /*0x50-0x57*/
03974 0x8ebe, 0x8ebf, 0x8ec0, 0xe1a4, 0x8ec1, 0x8ec2, 0xe1a5, 0x8ec3, /*0x58-0x5f*/
03975 0x8ec4, 0xe1a6, 0xb4b1, 0x8ec5, 0x8ec6, 0x8ec7, 0x8ec8, 0x8ec9, /*0x60-0x67*/
03976 0x8eca, 0x8ecb, 0x8ecc, 0x8ecd, 0x8ece, 0x8ecf, 0x8ed0, 0x8ed1, /*0x68-0x6f*/
03977 0x8ed2, 0x8ed3, 0x8edc, 0xc8c9, 0xc6bd, 0xc4ea, 0x8ed4, 0xb2a2, /*0x70-0x77*/
03978 0xd0d2, 0x8ed6, 0xe7db, 0xbbc3, 0xd3d7, 0xd3c4, 0x8ed7, 0xb9e3, /*0x78-0x7f*/
03979 0xe2cf, 0x8ed8, 0x8ed9, 0x8eda, 0xd7af, 0x8edb, 0xc7ec, 0xb1d3, /*0x80-0x87*/
03980 0x8edc, 0x8edd, 0xb4b2, 0xe2d1, 0x8ede, 0x8edf, 0x8ee0, 0xd0f2, /*0x88-0x8f*/
03981 0xc2ae, 0xe2d0, 0x8ee1, 0xbfe2, 0xd3a6, 0xb5d7, 0xe2d2, 0xb5ea, /*0x90-0x97*/
03982 0x8ee2, 0xc3ed, 0xb8fd, 0x8ee3, 0xb8ae, 0x8ee4, 0xc5d3, 0xb7cf, /*0x98-0x9f*/
03983 0xe2d4, 0x8ee5, 0x8ee6, 0x8ee7, 0x8ee8, 0xe2d3, 0xb6c8, 0xd7f9, /*0xa0-0xaf*/
03984 0x8ee9, 0x8eea, 0x8eeb, 0x8eec, 0x8eed, 0xcda5, 0x8eee, 0x8eef, /*0xa8-0xaf*/
03985 0x8ef0, 0x8ef1, 0x8ef2, 0xe2d8, 0x8ef3, 0xe2d6, 0xc7af, 0xbfb5, /*0xb0-0xb7*/
03986 0xd3b9, 0xe2d5, 0x8ef4, 0x8ef5, 0x8ef6, 0x8ef7, 0xe2d7, 0x8ef8, /*0xb8-0xbf*/
03987 0x8ef9, 0x8efa, 0x8efb, 0x8efc, 0x8efd, 0x8efe, 0x8f40, 0x8f41, /*0xc0-0xc7*/
03988 0x8f42, 0xc1ae, 0xc0c8, 0x8f43, 0x8f44, 0x8f45, 0x8f46, 0x8f47, /*0xc8-0xcf*/
03989 0x8f48, 0xe2db, 0xe2da, 0xc0aa, 0x8f49, 0x8f4a, 0xc1ce, 0x8f4b, /*0xd0-0xd7*/
03990 0x8f4c, 0x8f4d, 0x8f4e, 0xe2dc, 0x8f4f, 0x8f50, 0x8f51, 0x8f52, /*0xd8-0xdf*/
03991 0x8f53, 0x8f54, 0x8f55, 0x8f56, 0x8f57, 0x8f58, 0x8f59, 0x8f5a, /*0xe0-0xe7*/
03992 0xe2dd, 0x8f5b, 0xe2de, 0x8f5c, 0x8f5d, 0x8f5e, 0x8f5f, 0x8f60, /*0xe8-0xef*/
03993 0x8f61, 0x8f62, 0x8f63, 0x8f64, 0xdbcb, 0x8f65, 0xd1d3, 0xcda2, /*0xf0-0xf7*/
03994 0x8f66, 0x8f67, 0xbda8, 0x8f68, 0x8f69, 0x8f6a, 0xdec3, 0xd8a5, /*0xf8-0xff*/
03995 /* 0x5f00 */
03996 0xbfaa, 0xdbcd, 0xd2ec, 0xc6fa, 0xc5aa, 0x8f6b, 0x8f6c, 0x8f6d, /*0x00-0x07*/
03997 0xdec4, 0x8f6e, 0xb1d7, 0xdfae, 0x8f6f, 0x8f70, 0x8f71, 0xcabd, /*0x08-0x0f*/
03998 0x8f72, 0xdfb1, 0x8f73, 0xb9ad, 0x8f74, 0xd2fd, 0x8f75, 0xb8a5, /*0x10-0x17*/
03999 0xbabeb, 0x8f76, 0x8f77, 0xb3da, 0x8f78, 0x8f79, 0x8f7a, 0xb5dc, /*0x18-0x1f*/
04000 0xd5c5, 0x8f7b, 0x8f7c, 0x8f7d, 0x8f7e, 0xc3d6, 0xcfd2, 0xbba1, /*0x20-0x27*/
04001 0x8f80, 0xe5f3, 0xe5f2, 0x8f81, 0x8f82, 0xe5f4, 0x8f83, 0xcde4, /*0x28-0x2f*/
04002 0x8f84, 0xc8f5, 0x8f85, 0x8f86, 0x8f87, 0x8f88, 0x8f89, 0x8f8a, /*0x30-0x37*/
04003 0x8f8b, 0xb5af, 0xc7bf, 0x8f8c, 0xe5f6, 0x8f8d, 0x8f8e, 0x8f8f, /*0x38-0x3f*/
04004 0xecb0, 0x8f90, 0x8f91, 0x8f92, 0x8f93, 0x8f94, 0x8f95, 0x8f96, /*0x40-0x47*/
04005 0x8f97, 0x8f98, 0x8f99, 0x8f9a, 0x8f9b, 0x8f9c, 0x8f9d, 0x8f9e, /*0x48-0x4f*/
04006 0xe5e6, 0x8f9f, 0xb9e9, 0xb5b1, 0x8fa0, 0xc2bc, 0xe5e8, 0xe5e7, /*0x50-0x57*/
04007 0xe5e9, 0x8fa1, 0x8fa2, 0x8fa3, 0x8fa4, 0xd2cd, 0x8fa5, 0x8fa6, /*0x58-0x5f*/
```



```

04008 0x8fa7, 0xe1ea, 0xd0ce, 0x8fa8, 0xcdae, 0x8fa9, 0xd1e5, 0x8faa, /*0x60-0x67*/
04009 0x8fab, 0xb2ca, 0xb1eb, 0x8fac, 0xb1f2, 0xc5ed, 0x8fad, 0x8fae, /*0x68-0x6f*/
04010 0xd5c3, 0xd3b0, 0x8faf, 0x1dc, 0x8fb0, 0x8fb1, 0x8fb2, 0x1dd, /*0x70-0x77*/
04011 0x8fb3, 0xd2db, 0x8fb4, 0xb3b9, 0xb1cb, 0x8fb5, 0x8fb6, 0x8fb7, /*0x78-0x7f*/
04012 0xcdf9, 0xd5f7, 0x1de, 0x8fb8, 0xbeb6, 0xb4fd, 0x8fb9, 0x1df, /*0x80-0x87*/
04013 0xbadc, 0xe1e0, 0xbbb2, 0xc2c9, 0xe1e1, 0x8fba, 0x8bbb, 0x8fbc, /*0x88-0x8f*/
04014 0xd0ec, 0x8fbd, 0xcdbd, 0x8fbe, 0x8fbf, 0xe1e2, 0x8fc0, 0xb5c3, /*0x90-0x97*/
04015 0xc5c7, 0xe1e3, 0x8fc1, 0x8fc2, 0xe1e4, 0x8fc3, 0x8fc4, 0x8fc5, /*0x98-0x9f*/
04016 0x8fc6, 0xd3f9, 0x8fc7, 0x8fc8, 0x8fc9, 0x8fca, 0x8fcb, 0x8fcc, /*0xa0-0xa7*/
04017 0xe1e5, 0x8fcd, 0xd1ad, 0x8fce, 0x8fcf, 0xe1e6, 0xcea2, 0x8fd0, /*0xa8-0xaf*/
04018 0x8fd1, 0x8fd2, 0x8fd3, 0x8fd4, 0x8fd5, 0xe1e7, 0x8fd6, 0xb5c2, /*0xb0-0xb7*/
04019 0x8fd7, 0x8fd8, 0x8fd9, 0x8fda, 0xe1e8, 0xbbd5, 0x8fdb, 0x8fdc, /*0xb8-0xbf*/
04020 0x8fdd, 0x8fde, 0x8fdf, 0xd0c4, 0xe2e0, 0xb1d8, 0xd2e4, 0x8fe0, /*0xc0-0xc7*/
04021 0x8fe1, 0xe2e1, 0x8fe2, 0x8fe3, 0xbcc9, 0xc8cc, 0x8fe4, 0xe2e3, /*0xc8-0xcf*/
04022 0xecfe, 0xecfd, 0xdfaf, 0x8fe6, 0x8fe7, 0xe2e2, 0xd6be, /*0xd0-0xd7*/
04023 0xcdfc, 0xc3a6, 0x8fe8, 0x8fe9, 0x8fea, 0xe3c3, 0x8feb, 0x8fec, /*0xd8-0xdf*/
04024 0xd6d2, 0xe2e7, 0x8fed, 0x8fee, 0xe2e8, 0x8fef, 0x8ff0, 0xd3c7, /*0xe0-0xe7*/
04025 0x8ff1, 0x8ff2, 0xe2ec, 0xbfec, 0x8ff3, 0xe2ed, 0xe2e5, 0x8ffa, /*0xe8-0xef*/
04026 0x8ff5, 0xb3c0, 0x8ff6, 0x8ff7, 0x8ff8, 0xc4ee, 0x8ff9, 0x8ffa, /*0xf0-0xf7*/
04027 0xe2ee, 0x8ffb, 0x8ffc, 0xd0c3, 0x8ffd, 0xbaf6, 0xe2e9, 0xb7de, /*0xf8-0xff*/
04028 /* 0x6000 */
04029 0xbbb3, 0xccac, 0xcbbcb, 0xe2e4, 0xe2e6, 0xe2ea, 0xe2eb, 0x8ffe, /*0x00-0x07*/
04030 0x9040, 0x9041, 0xe2f7, 0x9042, 0x9043, 0xe2f4, 0xd4f5, 0xe2f3, /*0x08-0x0f*/
04031 0x9044, 0x9045, 0xc5ad, 0x9046, 0xd5fa, 0xc5c2, 0xb2c0, 0x9047, /*0x10-0x17*/
04032 0x9048, 0xe2ef, 0x9049, 0xe2f2, 0xc1af, 0xcbbc, 0x904a, 0x904b, /*0x18-0x1f*/
04033 0xb5a1, 0xe2f9, 0x904c, 0x904d, 0x904e, 0xbcb1, 0xe2f1, 0xd0d4, /*0x20-0x27*/
04034 0xd4b9, 0xe2f5, 0xb9d6, 0xe2f6, 0x904f, 0x9050, 0x9051, 0xc7d3, /*0x28-0x2f*/
04035 0x9052, 0x9053, 0x9054, 0x9055, 0x9056, 0xe2f0, 0x9057, 0x9058, /*0x30-0x37*/
04036 0x9059, 0x905a, 0x905b, 0xd7dc, 0xeda1, 0x905c, 0x905d, 0xe2f8, /*0x38-0x3f*/
04037 0x905e, 0xeda5, 0xe2fe, 0xcad1, 0x905f, 0x9060, 0x9061, 0x9062, /*0x40-0x47*/
04038 0x9063, 0x9064, 0x9065, 0xc1b5, 0x9066, 0xbbd0, 0x9067, 0x9068, /*0x48-0x4f*/
04039 0xbfd6, 0x9069, 0xbae3, 0x906a, 0x906b, 0xcba1, 0x906c, 0x906d, /*0x50-0x57*/
04040 0x906e, 0xeda6, 0xeda3, 0x906f, 0x9070, 0xeda2, 0x9071, 0x9072, /*0x58-0x5f*/
04041 0x9073, 0x9074, 0xbbd6, 0xeda7, 0xd0f4, 0x9075, 0x9076, 0xeda4, /*0x60-0x67*/
04042 0xbade, 0xb6f7, 0xe3a1, 0xb6b2, 0xccf1, 0xb9a7, 0x9077, 0xcfa2, /*0x68-0x6f*/
04043 0xc7a1, 0x9078, 0x9079, 0xbfd2, 0x907a, 0x907b, 0xb6f1, 0x907c, /*0x70-0x77*/
04044 0xe2fa, 0xe2fb, 0xe2fd, 0xe2fc, 0xc4d5, 0xe3a2, 0x907d, 0xd3c1, /*0x78-0x7f*/
04045 0x907e, 0x9080, 0x9081, 0xe3a7, 0xc7c4, 0x9082, 0x9083, 0x9084, /*0x80-0x87*/
04046 0x9085, 0xcfa4, 0x9086, 0x9087, 0xe3a9, 0xbab7, 0x9088, 0x9089, /*0x88-0x8f*/
04047 0x908a, 0x908b, 0xe3a8, 0x908c, 0xbbda, 0x908d, 0xe3a3, 0x908e, /*0x90-0x97*/
04048 0x908f, 0x9090, 0xe3a4, 0xe3aa, 0x9091, 0xe3a6, 0x9092, 0xcef2, /*0x98-0x9f*/
04049 0xd3c6, 0x9093, 0x9094, 0xbbbc, 0x9095, 0x9096, 0xd4c3, 0x9097, /*0xa0-0xaf*/
04050 0xc4fa, 0x9098, 0x9099, 0xeda8, 0xd0fc, 0xe3a5, 0x909a, 0xc3f5, /*0xa8-0xaf*/
04051 0x909b, 0xe3ad, 0xb1af, 0x909c, 0xe3b2, 0x909d, 0x909e, 0x909f, /*0xb0-0xb7*/
04052 0xbcc2, 0x90a0, 0x90a1, 0xe3ac, 0xb5bf, 0x90a2, 0x90a3, 0x90a4, /*0xb8-0xbf*/
04053 0x90a5, 0x90a6, 0x90a7, 0x90a8, 0x90a9, 0xc7e9, 0xe3b0, 0x90aa, /*0xc0-0xc7*/
04054 0x90ab, 0x90ac, 0xbeaa, 0xcdef, 0x90ad, 0x90ae, 0x90af, 0x90b0, /*0xc8-0xcf*/
04055 0x90b1, 0xbbf3, 0x90b2, 0x90b3, 0xcce8, 0x90b4, 0x90b5, 0x90b6, /*0xd0-0xd7*/
04056 0xe3af, 0x90b7, 0xe3b1, 0x90b8, 0xcfa7, 0xe3ae, 0x90b9, 0xcea9, /*0xd8-0xdf*/
04057 0xbbbd, 0x90ba, 0x90bb, 0x90bc, 0x90bd, 0x90be, 0xb5eb, 0xbee5, /*0xe0-0xe7*/
04058 0xb2d2, 0xb3cd, 0x90bf, 0xb1b9, 0xe3ab, 0xb2d1, 0xb5ac, 0xb9df, /*0xe8-0xef*/
04059 0xb6e8, 0x90c0, 0x90c1, 0xcfeb, 0xe3b7, 0x90c2, 0xbbcc, 0x90c3, /*0xf0-0xf7*/
04060 0x90c4, 0xc8c7, 0xd0ca, 0x90c5, 0x90c6, 0x90c7, 0x90c8, 0x90c9, /*0xf8-0xff*/
04061 /* 0x6100 */
04062 0xe3b8, 0xb3ee, 0x90ca, 0x90cb, 0x90cc, 0x90cd, 0xeda9, 0x90ce, /*0x00-0x07*/
04063 0xd3fa, 0xd3e4, 0x90cf, 0x90d0, 0x90d1, 0xeda, 0xe3b9, 0xd2e2, /*0x08-0x0f*/
04064 0x90d2, 0x90d3, 0x90d4, 0x90d5, 0xe3b5, 0x90d6, 0xe3b5, 0x90d7, /*0x10-0x17*/
04065 0x90d9, 0x90da, 0xd3de, 0x90db, 0x90dc, 0x90dd, 0x90de, 0xb8d0, /*0x18-0x1f*/
04066 0xe3b3, 0x90df, 0x90e0, 0xe3b6, 0xb7df, 0x90e1, 0xe3b4, 0xc0a2, /*0x20-0x27*/
04067 0x90e2, 0x90e3, 0x90e4, 0xe3ba, 0x90e5, 0x90e6, 0x90e7, 0x90e8, /*0x28-0x2f*/
04068 0x90e9, 0x90ea, 0x90eb, 0x90ec, 0x90ed, 0x90ee, 0x90ef, 0x90f0, /*0x30-0x37*/
04069 0x90f1, 0x90f2, 0x90f3, 0x90f4, 0x90f5, 0x90f6, 0x90f7, 0xd4b8, /*0x38-0x3f*/
04070 0x90f8, 0x90f9, 0x90fa, 0x90fb, 0x90fc, 0x90fd, 0x90fe, 0x9140, /*0x40-0x47*/
04071 0xb4c8, 0x9141, 0xe3bb, 0x9142, 0xbbc5, 0x9143, 0xc9f7, 0x9144, /*0x48-0x4f*/
04072 0x9145, 0xc9e5, 0x9146, 0x9147, 0x9148, 0xc4bd, 0x9149, 0x914a, /*0x50-0x57*/
04073 0x914b, 0x914c, 0x914d, 0x914e, 0x914f, 0xedab, 0x9150, 0x9151, /*0x58-0x5f*/
04074 0x9152, 0x9153, 0xc2fd, 0x9154, 0x9155, 0x9156, 0x9157, 0xbbdb, /*0x60-0x67*/
04075 0xbfae, 0x9158, 0x9159, 0x915a, 0x915b, 0x915c, 0x915d, 0x915e, /*0x68-0x6f*/
04076 0xcdbf, 0x915f, 0x9160, 0x9161, 0x9162, 0xe3bc, 0x9163, 0xbfb6, /*0x70-0x77*/
04077 0x9164, 0x9165, 0x9166, 0x9167, 0x9168, 0x9169, 0x916a, 0x916b, /*0x78-0x7f*/
04078 0x916c, 0x916d, 0x916e, 0x916f, 0x9170, 0x9171, 0x9172, 0x9173, /*0x80-0x87*/
04079 0x9174, 0x9175, 0x9176, 0xb1ef, 0x9177, 0x9178, 0xd4f7, 0x9179, /*0x88-0x8f*/
04080 0x917a, 0x917b, 0x917c, 0x917d, 0xe3be, 0x917e, 0x9180, 0x9181, /*0x90-0x97*/
04081 0x9182, 0x9183, 0x9184, 0x9185, 0x9186, 0xedad, 0x9187, 0x9188, /*0x98-0x9f*/
04082 0x9189, 0x918a, 0x918b, 0x918c, 0x918d, 0x918e, 0x918f, 0xe3bf, /*0xa0-0xaf*/
04083 0xbbaa, 0xedac, 0x9190, 0x9191, 0xe3bd, 0x9192, 0x9193, 0x9194, /*0xa8-0xaf*/
04084 0x9195, 0x9196, 0x9197, 0x9198, 0x9199, 0x919a, 0x919b, 0xe3c0, /*0xb0-0xbf*/
04085 0x919c, 0x919d, 0x919e, 0x919f, 0x91a0, 0x91a1, 0xbab6, 0x91a2, /*0xb8-0xbf*/
04086 0x91a3, 0x91a4, 0xb6ae, 0x91a5, 0x91a6, 0x91a7, 0x91a8, 0x91a9, /*0xc0-0xc7*/
04087 0xd0b8, 0x91aa, 0xb0c3, 0xedae, 0x91ab, 0x91ac, 0x91ad, 0x91ae, /*0xc8-0xcf*/
04088 0x91af, 0xeda, 0xc0c1, 0x91b0, 0xe3c1, 0x91b1, 0x91b2, 0x91b3, /*0xd0-0xd7*/
04089 0x91b4, 0x91b5, 0x91b6, 0x91b7, 0x91b8, 0x91b9, 0x91ba, 0x91bb, /*0xd8-0xdf*/
04090 0x91bc, 0x91bd, 0x91be, 0x91bf, 0x91c0, 0x91c1, 0xc5b3, 0x91c2, /*0xe0-0xe7*/
04091 0x91c3, 0x91c4, 0x91c5, 0x91c6, 0x91c7, 0x91c8, 0x91c9, 0x91ca, /*0xe8-0xef*/
04092 0x91cb, 0x91cc, 0x91cd, 0x91ce, 0x91cf, 0xe3c2, 0x91d0, 0x91d1, /*0xf0-0xf7*/
04093 0x91d2, 0x91d3, 0x91d4, 0x91d5, 0x91d6, 0x91d7, 0x91d8, 0xdcdb, /*0xf8-0xff*/
04094 /* 0x6200 */

```

```
04095 0x91d9, 0x91da, 0x91db, 0x91dc, 0x91dd, 0x91de, 0xedb0, 0x91df, /*x00-0x07*/
04096 0xb8ea, 0x91e0, 0xceec, 0xeaa7, 0xd0e7, 0xcaf9, 0xc8d6, 0xcf7b, /*x08-0x0f*/
04097 0xb3c9, 0xc2ed, 0xbde4, 0x91e1, 0x91e2, 0xe3de, 0xbbf2, 0xeaa8, /*x10-0x17*/
04098 0xd5bd, 0x91e3, 0xc6dd, 0xeaa9, 0x91e4, 0x91e5, 0x91e6, 0xeaaa, /*x18-0x1f*/
04099 0x91e7, 0xeaac, 0xeaa9, 0x91e8, 0xeaae, 0xeaa9, 0x91e9, 0x91ea, /*x20-0x27*/
04100 0x91eb, 0x91ec, 0xbdd8, 0x91ed, 0xeaa7, 0x91ee, 0xc2be, 0x91ef, /*x28-0x2f*/
04101 0x91f0, 0x91f1, 0x91f2, 0xb4c1, 0xb4f7, 0x91f3, 0x91f4, 0xbba7, /*x30-0x37*/
04102 0x91f5, 0x91f6, 0x91f7, 0x91f8, 0x91f9, 0xece6, 0xece5, 0xb7bf, /*x38-0x3f*/
04103 0xcbf9, 0xb1e2, 0x91fa, 0xece7, 0x91fb, 0x91fc, 0x91fd, 0xc9c8, /*x40-0x47*/
04104 0xece8, 0xece9, 0x91fe, 0xcad6, 0xded0, 0xb2c5, 0xd4fa, 0x9240, /*x48-0x4f*/
04105 0x9241, 0xc6cb, 0xb0c7, 0xb4f2, 0xc8d3, 0x9242, 0x9243, 0x9244, /*x50-0x57*/
04106 0xcdd0, 0x9245, 0x9246, 0xbfb8, 0x9247, 0x9248, 0x9249, 0x924a, /*x58-0x5f*/
04107 0x924b, 0x924c, 0x924d, 0xbfdb, 0x924e, 0x924f, 0xc7a4, 0xd6b4, /*x60-0x67*/
04108 0x9250, 0xc0a9, 0xded1, 0xc9a8, 0xd1ef, 0xc5a4, 0xb0e7, 0xb3b6, /*x68-0x6f*/
04109 0xc8c5, 0x9251, 0x9252, 0xbfb8, 0x9253, 0x9254, 0xb7f6, 0x9255, /*x70-0x77*/
04110 0x9256, 0xc5fa, 0x9257, 0x9258, 0xb6f3, 0x9259, 0xd5d2, 0xb3d0, /*x78-0x7f*/
04111 0xbcb9, 0x925a, 0x925b, 0x925c, 0xb3ad, 0x925d, 0x925e, 0x925f, /*x80-0x87*/
04112 0x9260, 0xbef1, 0xb0d1, 0x9261, 0x9262, 0x9263, 0x9264, 0x9265, /*x88-0x8f*/
04113 0x9266, 0xd2d6, 0xcae3, 0xd7a5, 0x9267, 0xcdb6, 0xb6b6, 0xbfb9, /*x90-0x97*/
04114 0xd5db, 0x9268, 0xb8a7, 0xc5d7, 0x9269, 0x926a, 0x926b, 0xded2, /*x98-0x9f*/
04115 0xbfd9, 0xc2d5, 0xc7c0, 0x926c, 0xbba4, 0xb1a8, 0x926d, 0x926e, /*xa0-0xaf*/
04116 0xc5ea, 0x926f, 0x9270, 0xc5fb, 0xc7ca, 0x9271, 0x9272, 0x9273, /*xa8-0xaf*/
04117 0x9274, 0xb1a7, 0x9275, 0x9276, 0x9277, 0xb5d6, 0x9278, 0x9279, /*xb0-0xb7*/
04118 0x927a, 0xc4a8, 0x927b, 0xded3, 0xd1ba, 0xb3e9, 0x927c, 0xc3f2, /*xb8-0xbf*/
04119 0x927d, 0x927e, 0xb7f7, 0x9280, 0xd6f4, 0xb5a3, 0xb2f0, 0xc4b4, /*xc0-0xc7*/
04120 0xc4e9, 0xc0ad, 0xded4, 0x9281, 0xb0e8, 0xc5c4, 0xc1e0, 0x9282, /*xc8-0xcf*/
04121 0xb9d5, 0x9282, 0xb9d5, 0xb9d5, 0xb9d5, 0xb9d5, 0xb9d5, 0xb9d5, /*xd0-0xdf*/
04122 0xb9d5, 0x9283, 0xc2d5, 0xc2d5, 0xc2d5, 0xc2d5, 0xc2d5, 0xc2d5, /*xe0-0xef*/
04123 0x9287, 0x9288, 0xc2a3, 0xbcf0, 0x9289, 0xd3b5, 0xc0b9, 0xc5a1, /*xe8-0xef*/
04124 0xb2a6, 0xd4f1, 0x928a, 0x928b, 0xc0a8, 0xcac3, 0xded7, 0xd5fc, /*xe8-0xef*/
04125 0x928c, 0xb9b0, 0x928d, 0xc8ad, 0xcba9, 0x928e, 0xded9, 0xbfbf, /*xf0-0xff*/
04126 0x928f, 0x9290, 0x9291, 0x9292, 0xc6b4, 0xd7a7, 0xcab0, 0xc4c3, /*xf8-0xff*/
04127 /* 0x6300 */
04128 0x9293, 0xb3d6, 0xb9d2, 0x9294, 0x9295, 0x9296, 0x9297, 0xd6b8, /*x00-0x07*/
04129 0xeafc, 0xb0b4, 0x9298, 0x9299, 0x929a, 0x929b, 0xbfe6, 0x929c, /*x08-0x0f*/
04130 0x929d, 0xc2f4, 0x929e, 0x929f, 0x92a0, 0x92a1, 0xcdda, 0x92a2, /*x10-0x17*/
04131 0x92a3, 0x92a4, 0xd6bf, 0xc2ce, 0x92a5, 0xcece, 0xc7ca, 0xd0ae, /*x18-0x1f*/
04132 0xc4d3, 0xb5b2, 0xded8, 0xd5f5, 0xbcb7, 0xbdb3, 0x92a6, 0x92a7, /*x20-0x27*/
04133 0xb0a4, 0x92a8, 0xc5b2, 0xb4ec, 0x92a9, 0x92aa, 0x92ab, 0xd5f1, /*x28-0x2f*/
04134 0x92ac, 0x92ad, 0xeafd, 0x92ae, 0x92af, 0x92b0, 0x92b1, 0x92b2, /*x30-0x37*/
04135 0x92b3, 0xdeda, 0xcda6, 0x92b4, 0x92b5, 0xcdec, 0x92b6, 0x92b7, /*x38-0x3f*/
04136 0x92b8, 0x92b9, 0xc2e6, 0xdedc, 0x92ba, 0xcdb1, 0xc0a6, 0x92bb, /*x40-0x47*/
04137 0x92bc, 0xd7bd, 0x92bd, 0xdedb, 0xb0c6, 0xbab4, 0xc9d3, 0xc4f3, /*x48-0x4f*/
04138 0xbbee, 0x92be, 0x92bf, 0x92c0, 0x92c1, 0xb2b6, 0x92c2, 0x92c3, /*x50-0x57*/
04139 0x92c4, 0x92c5, 0x92c6, 0x92c7, 0x92c8, 0x92c9, 0xc0cc, 0xcfb0, /*x58-0x5f*/
04140 0x92ca, 0xbcf1, 0xbbbb, 0xb5b7, 0x92cb, 0x92cc, 0x92cd, 0xc5f5, /*x60-0x67*/
04141 0x92ce, 0xdee6, 0x92cf, 0x92d0, 0x92d1, 0xdee3, 0xbedd, 0x92d2, /*x68-0x6f*/
04142 0x92d3, 0xdedf, 0x92d4, 0x92d5, 0x92d6, 0x92d7, 0xb4b7, 0xbddb, /*x70-0x77*/
04143 0x92d8, 0x92d9, 0xdee0, 0xc4ed, 0x92da, 0x92db, 0x92dc, 0x92dd, /*x78-0x7f*/
04144 0xcfc6, 0x92de, 0xb5e0, 0x92df, 0x92e0, 0x92e1, 0x92e2, 0xb6de, /*x80-0x87*/
04145 0xcada, 0xb5f4, 0xdee5, 0x92e3, 0xd5c6, 0x92e4, 0xdee1, 0xc2cd, /*x88-0x8f*/
04146 0xc6fe, 0x92e5, 0xc5c5, 0x92e6, 0x92e7, 0x92e8, 0xd2b4, 0x92e9, /*x90-0x97*/
04147 0xbef2, 0x92ea, 0x92eb, 0x92ec, 0x92ed, 0x92ee, 0x92ef, 0x92f0, /*x98-0x9f*/
04148 0xc2d3, 0x92f1, 0xc2bd, 0xb3b8, 0x92f2, 0xbdd3, 0x92f3, 0xbfd8, /*xa0-0xaf*/
04149 0xc2dc, 0xd1da, 0xb4eb, 0x92f4, 0xdee4, 0xdedd, 0xdee7, 0x92f5, /*xa8-0xaf*/
04150 0xeafe, 0x92f6, 0x92f7, 0xc2b0, 0xdee2, 0x92f8, 0x92f9, 0xd6c0, /*xb0-0xb7*/
04151 0xb5a7, 0x92fa, 0xb2f4, 0x92fb, 0xdee8, 0x92fc, 0xdedf, 0x92fd, /*xb8-0xbf*/
04152 0x92fe, 0x9340, 0x9341, 0x9342, 0xdeed, 0x9343, 0xdedf, 0x9344, /*xc0-0xcf*/
04153 0x9345, 0xc8e0, 0x9346, 0x9347, 0x9348, 0xd7e1, 0xdeef, 0xc3e8, /*xc8-0xcf*/
04154 0xcce1, 0x9349, 0xb2e5, 0x934a, 0x934b, 0x934c, 0xd2be, 0x934d, /*xd0-0xdf*/
04155 0x934e, 0x934f, 0x9350, 0x9351, 0x9352, 0x9353, 0xdeee, 0x9354, /*xd8-0xdf*/
04156 0xdeeb, 0xc2d5, 0x9355, 0xb4a7, 0x9356, 0x9357, 0x9358, 0x9359, /*xe0-0xef*/
04157 0x935a, 0xbfab, 0x935b, 0x935c, 0xbdd2, 0x935d, 0x935e, 0x935f, /*xe8-0xef*/
04158 0x935f, 0x9360, 0xdee9, 0x9361, 0xd4ae, 0x9362, 0xdede, 0x9363, /*xf0-0xff*/
04159 0xdeea, 0x9364, 0x9365, 0x9366, 0x9367, 0xc0bf, 0x9368, 0xdeec, /*xf8-0xff*/
04160 /* 0x6400 */
04161 0xb2f3, 0xb8e9, 0xc2a7, 0x9369, 0x936a, 0xbdc1, 0x936b, 0x936c, /*x00-0x07*/
04162 0x936d, 0x936e, 0x936f, 0xdedf, 0xdedf, 0x9370, 0x9371, 0xb2ab, /*x08-0x0f*/
04163 0xb4a4, 0x9372, 0x9373, 0xb4ea, 0xc9a6, 0x9374, 0x9375, 0x9376, /*x10-0x17*/
04164 0x9377, 0x9378, 0x9379, 0xdedf, 0xc2bd, 0x937a, 0xb8e3, 0x937b, /*x18-0x1f*/
04165 0xdedf, 0xdedf, 0x937c, 0x937d, 0x937e, 0x9380, 0xdedf, 0x9381, /*x20-0x27*/
04166 0x9382, 0x9383, 0xc2cd, 0x9384, 0xb0e1, 0xb4ee, 0x9385, 0x9386, /*x28-0x2f*/
04167 0x9387, 0x9388, 0x9389, 0x938a, 0xe5ba, 0x938b, 0x938c, 0x938d, /*x30-0x37*/
04168 0x938e, 0x938f, 0xd0af, 0x9390, 0x9391, 0xb2eb, 0x9392, 0xeba1, /*x38-0x3f*/
04169 0x9393, 0xdedf, 0x9394, 0x9395, 0xc9e3, 0xdedf, 0xb0da, 0xd2a1, /*x40-0x47*/
04170 0xb1f7, 0x9396, 0xc2af, 0x9397, 0x9398, 0x9399, 0x939a, 0x939b, /*x48-0x4f*/
04171 0x939c, 0x939d, 0xdedf, 0x939e, 0xcba4, 0x939f, 0x93a0, 0x93a1, /*x50-0x57*/
04172 0xd5aa, 0x93a2, 0x93a3, 0x93a4, 0x93a5, 0x93a6, 0xdedf, 0x93a7, /*x58-0x5f*/
04173 0x93a8, 0x93a9, 0x93aa, 0x93ab, 0x93ac, 0x93ad, 0x93ae, 0xb4dd, /*x60-0x67*/
04174 0x93af, 0xc4a6, 0x93b0, 0x93b1, 0x93b2, 0xdedf, 0x93b3, 0x93b4, /*x68-0x6f*/
04175 0x93b5, 0x93b6, 0x93b7, 0x93b8, 0x93b9, 0x93ba, 0x93bb, 0x93bc, /*x70-0x77*/
04176 0xc3fe, 0xc4a1, 0xdfa1, 0x93bd, 0x93be, 0x93bf, 0x93c0, 0x93c1, /*x78-0x7f*/
04177 0x93c2, 0x93c3, 0xc1cc, 0x93c4, 0xdedf, 0xbef, 0x93c5, 0xc6b2, /*x80-0x87*/
04178 0x93c6, 0x93c7, 0x93c8, 0x93c9, 0x93ca, 0x93cb, 0x93cc, 0x93cd, /*x88-0x8f*/
04179 0x93ce, 0xb3c5, 0xc8f6, 0x93cf, 0x93d0, 0xc2ba, 0xdedf, 0x93d1, /*x90-0x97*/
04180 0x93d2, 0xdfa4, 0x93d3, 0x93d4, 0x93d5, 0x93d6, 0xd7b2, 0x93d7, /*x98-0x9f*/
04181 0x93d8, 0x93d9, 0x93da, 0x93db, 0xb3b7, 0x93dc, 0x93dd, 0x93de, /*xa0-0xaf*/
```

```

04182 0x93df, 0xc1c3, 0x93e0, 0x93e1, 0xc7cb, 0xb2a5, 0xb4e9, 0x93e2, /*0xa8-0xaf*/
04183 0xd7ab, 0x93e3, 0x93e4, 0x93e5, 0x93e6, 0xc4ec, 0x93e7, 0xdfa2, /*0xb0-0xb7*/
04184 0xdfa3, 0x93e8, 0xdfa5, 0x93e9, 0xbab3, 0x93ea, 0x93eb, 0x93ec, /*0xb8-0xbf*/
04185 0xdfa6, 0x93ed, 0xc0de, 0x93ee, 0x93ef, 0xc9c3, 0x93f0, 0x93f1, /*0xc0-0xc7*/
04186 0x93f2, 0x93f3, 0x93f4, 0x93f5, 0x93f6, 0xb2d9, 0xc7e6, 0x93f7, /*0xc8-0xcf*/
04187 0xdfa7, 0x93f8, 0xc7dc, 0x93f9, 0x93fa, 0x93fb, 0x93fc, 0xdfa8, /*0xd0-0xd7*/
04188 0xeba2, 0x93fd, 0x93fe, 0x9440, 0x9441, 0x9442, 0xcdb3, 0x9443, /*0xd8-0xdf*/
04189 0x9444, 0x9445, 0xdfaa, 0x9446, 0xdfa9, 0x9447, 0xb2c1, 0x9448, /*0xe0-0xe7*/
04190 0x9449, 0x944a, 0x944b, 0x944c, 0x944d, 0x944e, 0x944f, 0x9450, /*0xe8-0xef*/
04191 0x9451, 0x9452, 0x9453, 0x9454, 0x9455, 0x9456, 0x9457, 0x9458, /*0xf0-0xf7*/
04192 0x9459, 0x945a, 0x945b, 0x945c, 0x945d, 0x945e, 0x945f, 0x9460, /*0xf8-0xff*/
04193 /* 0x6500 */
04194 0xc5ca, 0x9461, 0x9462, 0x9463, 0x9464, 0x9465, 0x9466, 0x9467, /*0x00-0x07*/
04195 0x9468, 0xdfab, 0x9469, 0x946a, 0x946b, 0x946c, 0x946d, 0x946e, /*0x08-0x0f*/
04196 0x946f, 0x9470, 0xd4dc, 0x9471, 0x9472, 0x9473, 0x9474, 0x9475, /*0x10-0x17*/
04197 0xc8c1, 0x9476, 0x9477, 0x9478, 0x9479, 0x947a, 0x947b, 0x947c, /*0x18-0x1f*/
04198 0x947d, 0x947e, 0x9480, 0x9481, 0x9482, 0xdfac, 0x9483, 0x9484, /*0x20-0x27*/
04199 0x9485, 0x9486, 0x9487, 0xbef0, 0x9488, 0x9489, 0xdfad, 0xd6a7, /*0x28-0x2f*/
04200 0x948a, 0x948b, 0x948c, 0x948d, 0xeab7, 0xebb6, 0xcad5, 0x948e, /*0x30-0x37*/
04201 0xd8fc, 0xb8c4, 0x948f, 0xb9a5, 0x9490, 0x9491, 0xb7c5, 0xd5fe, /*0x38-0x3f*/
04202 0x9492, 0x9493, 0x9494, 0x9495, 0x9496, 0xb9ca, 0x9497, 0x9498, /*0x40-0x47*/
04203 0xd0a7, 0xf4cd, 0x9499, 0x949a, 0xb5d0, 0x949b, 0x949c, 0xc3f4, /*0x48-0x4f*/
04204 0x949d, 0xbecb, 0x949e, 0x949f, 0x94a0, 0xebb7, 0xb0bd, 0x94a1, /*0x50-0x57*/
04205 0x94a2, 0xbdcc, 0x94a3, 0xc1b2, 0x94a4, 0xb1d6, 0xb3a8, 0x94a5, /*0x58-0x5f*/
04206 0x94a6, 0x94a7, 0xb8d2, 0xc9a2, 0x94a8, 0x94a9, 0xb6d8, 0x94aa, /*0x60-0x67*/
04207 0x94ab, 0x94ac, 0x94ad, 0xebb8, 0xbeb4, 0x94ae, 0x94af, 0x94b0, /*0x68-0x6f*/
04208 0xcacf, 0x94b1, 0xc7c3, 0x94b2, 0xd5fb, 0x94b3, 0x94b4, 0xb7f3, /*0x70-0x77*/
04209 0x94b5, 0x94b6, 0x94b7, 0x94b8, 0x94b9, 0x94ba, 0x94bb, 0x94bc, /*0x78-0x7f*/
04210 0x94bd, 0x94be, 0x94bf, 0x94c0, 0x94c1, 0x94c2, 0x94c3, 0xcce4, /*0x80-0x87*/
04211 0x94c4, 0x94c5, 0x94c6, 0xd5ab, 0xb1f3, 0x94c7, 0x94c8, 0x94c9, /*0x88-0x8f*/
04212 0xecb3, 0xb0df, 0x94ca, 0xecb5, 0x94cb, 0x94cc, 0x94cd, 0xb6b7, /*0x90-0x97*/
04213 0x94ce, 0xc1cf, 0x94cf, 0xf5fa, 0xd0b1, 0x94d0, 0x94d1, 0xd5e5, /*0x98-0x9f*/
04214 0x94d2, 0xcdd3, 0x94d3, 0x94d4, 0xbdef, 0xb3e2, 0x94d5, 0xb8ab, /*0xa0-0xaf*/
04215 0x94d6, 0xd5b6, 0x94d7, 0xedbd, 0x94d8, 0xb6cf, 0x94d9, 0xcbb9, /*0xa8-0xaf*/
04216 0xd0c2, 0x94da, 0x94db, 0x94dc, 0x94dd, 0x94de, 0x94df, 0x94e0, /*0xb0-0xb7*/
04217 0x94e1, 0xb7bd, 0x94e2, 0x94e3, 0xecb6, 0xcaa9, 0x94e4, 0x94e5, /*0xb8-0xbf*/
04218 0x94e6, 0xc5d4, 0x94e7, 0xecb9, 0xecb8, 0xc2c3, 0xecb7, 0x94e8, /*0xc0-0xc7*/
04219 0x94e9, 0x94ea, 0x94eb, 0xd0fd, 0xecba, 0x94ec, 0xecbb, 0xd7e5, /*0xc8-0xcf*/
04220 0x94ed, 0x94ee, 0xecbc, 0x94ef, 0x94f0, 0x94f1, 0xecbd, 0xc6ec, /*0xd0-0xd7*/
04221 0x94f2, 0x94f3, 0x94f4, 0x94f5, 0x94f6, 0x94f7, 0x94f8, 0x94f9, /*0xd8-0xdf*/
04222 0xcdec, 0x94fa, 0xbcc8, 0x94fb, 0x94fc, 0xc8d5, 0xb5a9, 0xbec9, /*0xe0-0xe7*/
04223 0xd6bc, 0xd4e7, 0x94fd, 0x94fe, 0xd1ae, 0xd0f1, 0xeab8, 0xeab9, /*0xe8-0xef*/
04224 0xeaba, 0xbab5, 0x9540, 0x9541, 0x9542, 0x9543, 0xcab1, 0xbff5, /*0xf0-0xf7*/
04225 0x9544, 0x9545, 0xcdfa, 0x9546, 0x9547, 0x9548, 0x9549, 0x954a, /*0xf8-0xff*/
04226 /* 0x6600 */
04227 0xeac0, 0x954b, 0xb0ba, 0xeabe, 0x954c, 0x954d, 0xc0a5, 0x954e, /*0x00-0x07*/
04228 0x954f, 0x9550, 0xeabb, 0x9551, 0xb2fd, 0x9552, 0xc3f7, 0xbbe8, /*0x08-0x0f*/
04229 0x9553, 0x9554, 0x9555, 0xd2d7, 0xcef4, 0xeabf, 0x9556, 0x9557, /*0x10-0x17*/
04230 0x9558, 0xeabc, 0x9559, 0x955a, 0x955b, 0xeac3, 0x955c, 0xd0c7, /*0x18-0x1f*/
04231 0xd3b3, 0x955d, 0x955e, 0x955f, 0x9560, 0xb4ba, 0x9561, 0xc3c1, /*0x20-0x27*/
04232 0xd7f2, 0x9562, 0x9563, 0x9564, 0x9565, 0xd5d1, 0x9566, 0xcac7, /*0x28-0x2f*/
04233 0x9567, 0xeac5, 0x9568, 0x9569, 0xeac4, 0xeac7, 0xeac6, 0x956a, /*0x30-0x37*/
04234 0x956b, 0x956c, 0x956d, 0x956e, 0xd6e7, 0x956f, 0xcfd4, 0x9570, /*0x38-0x3f*/
04235 0x9571, 0xeacb, 0x9572, 0xbbbe, 0x9573, 0x9574, 0x9575, 0x9576, /*0x40-0x47*/
04236 0x9577, 0x9578, 0x9579, 0xbdfa, 0xc9ce, 0x957a, 0x957b, 0xeacc, /*0x48-0x4f*/
04237 0x957c, 0x957d, 0xc9b9, 0xcffe, 0xeaca, 0xd4ce, 0xeacd, 0xeacf, /*0x50-0x57*/
04238 0x957e, 0x9580, 0xcdded, 0x9581, 0x9582, 0x9583, 0x9584, 0xeac9, /*0x58-0x5f*/
04239 0x9585, 0xeace, 0x9586, 0x9587, 0xccee, 0x9588, 0xbbbe, 0x9589, /*0x60-0x67*/
04240 0xb3bf, 0x958a, 0x958b, 0x958c, 0x958d, 0x958e, 0xc6d5, 0xbbe0, /*0x68-0x6f*/
04241 0xccea, 0x958f, 0x9590, 0xc7e7, 0x9591, 0xc7e7, 0x9592, 0x9593, /*0x70-0x77*/
04242 0x9593, 0x9594, 0xd6c7, 0x9595, 0x9596, 0x9597, 0xc1c0, 0x9598, /*0x78-0x7f*/
04243 0x9599, 0x959a, 0xd4dd, 0x959b, 0xeadd, 0x959c, 0x959d, 0xcfbe, /*0x80-0x87*/
04244 0x959e, 0x959f, 0x95a0, 0x95a1, 0xeadd, 0x95a2, 0x95a3, 0x95a4, /*0x88-0x8f*/
04245 0x95a5, 0xcaee, 0x95a6, 0x95a7, 0x95a8, 0x95a9, 0xc5af, 0xb0b5, /*0x90-0x97*/
04246 0x95aa, 0x95ab, 0x95ac, 0x95ad, 0x95ae, 0xeadd, 0x95af, 0x95b0, /*0x98-0x9f*/
04247 0x95b1, 0x95b2, 0x95b3, 0x95b4, 0x95b5, 0x95b6, 0x95b7, 0xeadd, /*0xa0-0xaf*/
04248 0xf4df, 0x95b8, 0x95b9, 0x95ba, 0x95bb, 0x95bc, 0xc4ba, 0x95bd, /*0xa8-0xaf*/
04249 0x95be, 0x95bf, 0x95c0, 0x95c1, 0xb1a9, 0x95c2, 0x95c3, 0x95c4, /*0xb0-0xb7*/
04250 0x95c5, 0xe5df, 0x95c6, 0x95c7, 0x95c8, 0x95c9, 0xeadd, 0x95ca, /*0xb8-0xbf*/
04251 0x95cb, 0x95cc, 0x95cd, 0x95ce, 0x95cf, 0x95d0, 0x95d1, 0x95d2, /*0xc0-0xc7*/
04252 0x95d3, 0x95d4, 0x95d5, 0x95d6, 0x95d7, 0x95d8, 0x95d9, 0x95da, /*0xc8-0xcf*/
04253 0x95db, 0x95dc, 0x95dd, 0x95de, 0x95df, 0x95e0, 0x95e1, 0x95e2, /*0xd0-0xd7*/
04254 0x95e3, 0xcaef, 0x95e4, 0xeadd, 0xeadd, 0xc6d8, 0x95e5, 0x95e6, /*0xd8-0xdf*/
04255 0x95e7, 0x95e8, 0x95e9, 0x95ea, 0x95eb, 0x95ec, 0xeadd, 0x95ed, /*0xe0-0xe7*/
04256 0x95ee, 0xeadd, 0x95ef, 0x95f0, 0x95f1, 0x95f2, 0x95f3, 0x95f4, /*0xe8-0xef*/
04257 0xd4bb, 0x95f5, 0xc7fa, 0xd2b7, 0xb8fc, 0x95f6, 0x95f7, 0xeac2, /*0xf0-0xf7*/
04258 0x95f8, 0xb2dc, 0x95f9, 0x95fa, 0xc2fc, 0x95fb, 0xd4f8, 0xcce6, /*0xf8-0xff*/
04259 /* 0x6700 */
04260 0xd7ee, 0x95fc, 0x95fd, 0x95fe, 0x9640, 0x9641, 0x9642, 0x9643, /*0x00-0x07*/
04261 0xd4c2, 0xd3d0, 0xebc3, 0xc5f3, 0x9644, 0xb7fe, 0x9645, 0x9646, /*0x08-0x0f*/
04262 0xebd4, 0x9647, 0x9648, 0x9649, 0xcbb7, 0xebde, 0x964a, 0xc0ca, /*0x10-0x17*/
04263 0x964b, 0x964c, 0x964d, 0xcdfb, 0x964e, 0xb3af, 0x964f, 0xc6da, /*0x18-0x1f*/
04264 0x9650, 0x9651, 0x9652, 0x9653, 0x9654, 0x9655, 0xebfc, 0x9656, /*0x20-0x27*/
04265 0xc4be, 0x9657, 0xcbe4, 0xc4a9, 0xb1be, 0xd4fd, 0x9658, 0xcacf, /*0x28-0x2f*/
04266 0x9659, 0xd6ec, 0x965a, 0x965b, 0xc6d3, 0xb6e4, 0x965c, 0x965d, /*0x30-0x37*/
04267 0x965e, 0x965f, 0xbbf5, 0x9660, 0x9661, 0xd0e0, 0x9662, 0x9663, /*0x38-0x3f*/
04268 0xc9b1, 0x9664, 0xd4d3, 0xc8a8, 0x9665, 0x9666, 0xb8cb, 0x9667, /*0x40-0x47*/

```



```
04269 0xe8be, 0xc9bc, 0x9668, 0x9669, 0xe8bb, 0x966a, 0xc0ee, 0xd0d3, /*0x48-0x4f*/
04270 0xb2c4, 0xb4e5, 0x966b, 0xe8bc, 0x966c, 0x966d, 0xd5c8, 0x966e, /*0x50-0x57*/
04271 0x966f, 0x9670, 0x9671, 0x9672, 0xb6c5, 0x9673, 0xe8bd, 0xcaf8, /*0x58-0x5f*/
04272 0xb8dc, 0xccf5, 0x9674, 0x9675, 0x9676, 0xc0b4, 0x9677, 0x9678, /*0x60-0x67*/
04273 0xd1ee, 0xe8bf, 0xe8c2, 0x9679, 0x967a, 0xbabc, 0x967b, 0xb1ad, /*0x68-0x6f*/
04274 0xbddc, 0x967c, 0xeabd, 0xe8c3, 0x967d, 0xe8c6, 0x967e, 0xe8cb, /*0x70-0x77*/
04275 0x9680, 0x9681, 0x9682, 0x9683, 0xe8cc, 0x9684, 0xc9c9, 0xb0e5, /*0x78-0x7f*/
04276 0x9685, 0xbcab, 0x9686, 0x9687, 0xb9b9, 0x9688, 0x9689, 0xe8c1, /*0x80-0x87*/
04277 0x968a, 0xcdf7, 0x968b, 0xe8ca, 0x968c, 0x968d, 0x968e, 0x968f, /*0x88-0x8f*/
04278 0xccef6, 0x9690, 0x9691, 0x9692, 0x9693, 0xd5ed, 0x9694, 0xc1d6, /*0x90-0x97*/
04279 0xe8c4, 0x9695, 0xc3b6, 0x9696, 0xb9fb, 0xd6a6, 0xe8c8, 0x9697, /*0x98-0x9f*/
04280 0x9698, 0x9699, 0xcae0, 0xd4e6, 0x969a, 0xe8c0, 0x969b, 0xe8c5, /*0xa0-0xa7*/
04281 0xe8c7, 0x969c, 0xc7b9, 0xb7e3, 0x969d, 0xe8c9, 0x969e, 0xbfd9, /*0xa8-0xaf*/
04282 0xe8d2, 0x969f, 0x96a0, 0xe8d7, 0x96a1, 0xe8d5, 0xbcdc, 0xbccf, /*0xb0-0xb7*/
04283 0xe8db, 0x96a2, 0x96a3, 0x96a4, 0x96a5, 0x96a6, 0x96a7, 0x96a8, /*0xb8-0xbf*/
04284 0x96a9, 0xe8de, 0x96aa, 0xe8da, 0xb1fa, 0x96ab, 0x96ac, 0x96ad, /*0xc0-0xc7*/
04285 0x96ae, 0x96af, 0x96b0, 0x96b1, 0x96b2, 0x96b3, 0x96b4, 0xb0d8, /*0xc8-0xcf*/
04286 0xc4b3, 0xb8cc, 0xc6e2, 0xc8be, 0xc8e1, 0x96b5, 0x96b6, 0x96b7, /*0xd0-0xd7*/
04287 0xe8cf, 0xe8d4, 0xe8d6, 0x96b8, 0xb9f1, 0xe8d8, 0xd7f5, 0x96b9, /*0xd8-0xdf*/
04288 0xc4fb, 0x96ba, 0xe8dc, 0x96bb, 0x96bc, 0xb2e9, 0x96bd, 0x96be, /*0xe0-0xef*/
04289 0x96bf, 0xe8d1, 0x96c0, 0x96c1, 0xbced, 0x96c2, 0x96c3, 0xbfc2, /*0xf0-0xf7*/
04290 0xe8cd, 0xd6f9, 0x96c4, 0xc1f8, 0xb2f1, 0x96c5, 0x96c6, 0x96c7, /*0xf8-0xff*/
04291 0x96c8, 0x96c9, 0x96ca, 0x96cb, 0x96cc, 0xe8df, 0x96cd, 0xcac1, /*0x00-0x07*/
04292 /* 0x6800 */
04293 0xe8d9, 0x96ce, 0x96cf, 0x96d0, 0x96d1, 0xd5a4, 0x96d2, 0xb1ea, /*0x08-0x0f*/
04294 0xd5bb, 0xe8ce, 0xe8d0, 0xb6b0, 0xe8d3, 0x96d3, 0xe8dd, 0xc0b8, /*0x10-0x17*/
04295 0x96d4, 0xcacf7, 0x96d5, 0xcba8, 0x96d6, 0x96d7, 0xc6dc, 0xc0f5, /*0x18-0x1f*/
04296 0x96d8, 0x96d9, 0x96da, 0x96db, 0x96dc, 0xe8e9, 0x96dd, 0x96de, /*0x20-0x27*/
04297 0x96df, 0xd0a3, 0x96e0, 0x96e1, 0x96e2, 0x96e3, 0x96e4, 0x96e5, /*0x28-0x2f*/
04298 0x96e6, 0xe8f2, 0xd6ea, 0x96e7, 0x96e8, 0x96e9, 0x96ea, 0x96eb, /*0x30-0x37*/
04299 0x96ec, 0x96ed, 0xe8e0, 0xe8e1, 0x96ee, 0x96ef, 0x96f0, 0xd1f9, /*0x38-0x3f*/
04300 0xbacb, 0xb8f9, 0x96f1, 0x96f2, 0xb8f1, 0xd4d4, 0xe8ef, 0x96f3, /*0x40-0x47*/
04301 0xe8ee, 0xe8ec, 0xb9f0, 0xcdd2, 0xe8ef, 0xc6a6, 0xbff2, 0x96f4, /*0x48-0x4f*/
04302 0xb0b8, 0xe8f1, 0xe8f0, 0x96f5, 0xd7c0, 0x96f6, 0xe8e4, 0x96f7, /*0x50-0x57*/
04303 0xcda9, 0xc9a3, 0x96f8, 0xbbb8, 0xbddb, 0xe8ea, 0x96f9, 0x96fa, /*0x58-0x5f*/
04304 0x96fb, 0x96fc, 0x96fd, 0x96fe, 0x9740, 0x9741, 0x9742, 0x9743, /*0x60-0x67*/
04305 0xe8e2, 0xe8e3, 0xe8e5, 0xb5b5, 0xe8e7, 0xc7c5, 0xe8eb, 0xe8ed, /*0x68-0x6f*/
04306 0xbdb0, 0xd7ae, 0x9744, 0xe8f8, 0x9745, 0x9746, 0x9747, 0x9748, /*0x70-0x77*/
04307 0x9749, 0x974a, 0x974b, 0x974c, 0xe8f5, 0x974d, 0xcdb0, 0xe8f6, /*0x78-0x7f*/
04308 0x974e, 0x974f, 0x9750, 0x9751, 0x9752, 0x9753, 0x9754, 0x9755, /*0x80-0x87*/
04309 0x9756, 0xc1ba, 0x9757, 0xe8e8, 0x9758, 0xc3b7, 0xb0f0, 0x9759, /*0x88-0x8f*/
04310 0x975a, 0x975b, 0x975c, 0x975d, 0x975e, 0x975f, 0x9760, 0xe8f4, /*0x90-0x97*/
04311 0x9761, 0x9762, 0x9763, 0xe8f7, 0x9764, 0x9765, 0x9766, 0xb9a3, /*0x98-0x9f*/
04312 0x9767, 0x9768, 0x9769, 0x976a, 0x976b, 0x976c, 0x976d, 0x976e, /*0xa0-0xaf*/
04313 0x976f, 0x9770, 0xc9d2, 0x9771, 0x9772, 0x9773, 0xc3ce, 0xcce0, /*0xb0-0xbf*/
04314 0xc0e6, 0x9774, 0x9775, 0x9776, 0x9777, 0xc9f3, 0x9778, 0xcddc, /*0xc0-0xc7*/
04315 0xd0b5, 0x9779, 0x977a, 0xcae1, 0x977b, 0xe8f3, 0x977c, 0x977d, /*0xc8-0xcf*/
04316 0x977e, 0x9780, 0x9781, 0x9782, 0x9783, 0x9784, 0x9785, 0x9786, /*0xd0-0xd7*/
04317 0xbcec, 0x9787, 0xe8f9, 0x9788, 0x9789, 0x978a, 0x978b, 0x978c, /*0xd8-0xdf*/
04318 0x978d, 0xc3de, 0x978e, 0xc6e5, 0x978f, 0xb9f7, 0x9790, 0x9791, /*0xe0-0xef*/
04319 0x9792, 0x9793, 0xb0f4, 0x9794, 0x9795, 0xd7d8, 0x9796, 0x9797, /*0xf0-0xf7*/
04320 0xbcac, 0x9798, 0xc5ef, 0x9799, 0x979a, 0x979b, 0x979c, 0x979d, /*0x00-0x07*/
04321 0xcccc4, 0x979e, 0x979f, 0xe9a6, 0x97a0, 0x97a1, 0x97a2, 0x97a3, /*0x08-0x0f*/
04322 0x97a4, 0x97a5, 0x97a6, 0x97a7, 0x97a8, 0x97a9, 0xc9ad, 0x97aa, /*0x10-0x17*/
04323 0xe9a2, 0xc0e2, 0x97ab, 0x97ac, 0x97ad, 0xbfc3, 0x97ae, 0x97af, /*0x18-0x1f*/
04324 0x97b0, 0xe8fe, 0xb9d7, 0x97b1, 0xe8fb, 0x97b2, 0x97b3, 0x97b4, /*0x20-0x27*/
04325 /* 0x6900 */
04326 0x97b5, 0xe9a4, 0x97b6, 0x97b7, 0x97b8, 0xd2ce, 0x97b9, 0x97ba, /*0x28-0x2f*/
04327 0x97bb, 0x97bd, 0x97be, 0xe9a3, 0x97bf, 0xd6b2, 0xd7b5, 0x97bf, /*0x30-0x37*/
04328 0xe9a7, 0x97c0, 0xbdb7, 0x97c1, 0x97c2, 0x97c3, 0x97c4, 0x97c5, /*0x38-0x3f*/
04329 0x97c6, 0x97c7, 0x97c8, 0x97c9, 0x97ca, 0x97cb, 0x97cc, 0xe8fc, /*0x40-0x47*/
04330 0xe8fd, 0x97cd, 0x97ce, 0x97cf, 0xe9a1, 0x97d0, 0x97d1, 0x97d2, /*0x48-0x4f*/
04331 0x97d3, 0x97d4, 0x97d5, 0x97d6, 0xcdd6, 0x97d7, 0x97d8, 0x97d9, /*0x50-0x57*/
04332 0xd2ac, 0x97da, 0x97db, 0x97dc, 0xe9b2, 0x97dd, 0x97de, 0x97df, /*0x58-0x5f*/
04333 0x97e0, 0xe9a9, 0x97e1, 0x97e2, 0x97e3, 0xb4aa, 0x97e4, 0xb4bb, /*0x60-0x67*/
04334 0x97e5, 0x97e6, 0xe9ab, 0x97e7, 0x97e8, 0x97e9, 0x97ea, 0x97eb, /*0x68-0x6f*/
04335 0x97ec, 0x97ed, 0x97ee, 0x97ef, 0x97f0, 0x97f1, 0x97f2, 0x97f3, /*0x70-0x77*/
04336 0x97f4, 0x97f5, 0x97f6, 0x97f7, 0xd0a8, 0x97f8, 0x97f9, 0xe9a5, /*0x78-0x7f*/
04337 0x97fa, 0x97fb, 0xb3fe, 0x97fc, 0x97fd, 0xe9ac, 0xc0e3, 0x97fe, /*0x80-0x87*/
04338 0xe9aa, 0x9840, 0x9841, 0xe9b9, 0x9842, 0x9843, 0xe9b8, 0x9844, /*0x88-0x8f*/
04339 0x9845, 0x9846, 0x9847, 0xe9ae, 0x9848, 0x9849, 0xe8fa, 0x984a, /*0x90-0x97*/
04340 0x984b, 0xe9a8, 0x984c, 0x984d, 0x984e, 0x984f, 0x9850, 0xbfac, /*0x98-0x9f*/
04341 0xe9b1, 0xe9ba, 0x9851, 0x9852, 0xc2a5, 0x9853, 0x9854, 0x9855, /*0xa0-0xaf*/
04342 0xe9af, 0x9856, 0xb8c5, 0x9857, 0xe9ad, 0x9858, 0xd3dc, 0xe9b4, /*0xb0-0xbf*/
04343 0xe9b5, 0xe9b7, 0x9859, 0x985a, 0x985b, 0xe9c7, 0x985c, 0x985d, /*0xc0-0xc7*/
04344 0x985e, 0x985f, 0x9860, 0x9861, 0xc0c6, 0xe9c5, 0x9862, 0x9863, /*0xc8-0xcf*/
04345 0xe9b0, 0x9864, 0x9865, 0xe9bb, 0xb0f1, 0x9866, 0x9867, 0x9868, /*0xd0-0xd7*/
04346 0x9869, 0x986a, 0x986b, 0x986c, 0x986d, 0x986e, 0x986f, 0xe9bc, /*0xd8-0xdf*/
04347 0xd5a5, 0x9870, 0x9871, 0xe9be, 0x9872, 0xe9bf, 0x9873, 0x9874, /*0xe0-0xef*/
04348 0x9875, 0xe9c1, 0x9876, 0x9877, 0xc1f1, 0x9878, 0x9879, 0xc8b6, /*0xf0-0xf7*/
04349 0x987a, 0x987b, 0x987c, 0xe9bd, 0x987d, 0x987e, 0x9880, 0x9881, /*0x00-0x07*/
04350 0x9882, 0xe9c2, 0x9883, 0x9884, 0x9885, 0x9886, 0x9887, 0x9888, /*0x08-0x0f*/
04351 0x9889, 0x988a, 0xe9c3, 0x988b, 0xe9b3, 0x988c, 0xe9b6, 0x988d, /*0x10-0x17*/
04352 0xbbbb1, 0x988e, 0x988f, 0x9890, 0xe9c0, 0x9891, 0x9892, 0x9893, /*0x18-0x1f*/
04353 0x9894, 0x9895, 0x9896, 0xbcf7, 0x9897, 0x9898, 0x9899, 0xe9c4, /*0x20-0x27*/
04354 0xe9c6, 0x989a, 0x989b, 0xe9c7, 0x989c, 0x989d, 0x989e, 0x989f, /*0x28-0x2f*/
04355 0x98a1, 0x98a2, 0x98a3, 0x98a4, 0x98a5, 0xe9ca, 0x98a6, 0x98a7, /*0x30-0x37*/
```

```

04356 0x98a8, 0x98a9, 0xe9ce, 0x98aa, 0x98ab, 0x98ac, 0x98ad, 0x98ae, /*0xf0-0xf7*/
04357 0x98af, 0x98b0, 0x98b1, 0x98b2, 0x98b3, 0xb2db, 0x98b4, 0xe9c8, /*0xf8-0xff*/
04358 /* 0x6a00 */
04359 0x98b5, 0x98b6, 0x98b7, 0x98b8, 0x98b9, 0x98ba, 0x98bb, 0x98bc, /*0x00-0x07*/
04360 0x98bd, 0x98be, 0xb7ae, 0x98bf, 0x98c0, 0x98c1, 0x98c2, 0x98c3, /*0x08-0x0f*/
04361 0x98c4, 0x98c5, 0x98c6, 0x98c7, 0x98c8, 0x98c9, 0x98ca, 0xe9cb, /*0x10-0x17*/
04362 0xe9cc, 0x98cb, 0x98cc, 0x98cd, 0x98ce, 0x98cf, 0x98d0, 0xd5c1, /*0x18-0x1f*/
04363 0x98d1, 0xc4a3, 0x98d2, 0x98d3, 0x98d4, 0x98d5, 0x98d6, 0x98d7, /*0x20-0x27*/
04364 0xe9d8, 0x98d8, 0xbae1, 0x98d9, 0x98da, 0x98db, 0x98dc, 0xe9c9, /*0x28-0x2f*/
04365 0x98dd, 0xd3a3, 0x98de, 0x98df, 0x98e0, 0xe9d4, 0x98e1, 0x98e2, /*0x30-0x37*/
04366 0x98e3, 0x98e4, 0x98e5, 0x98e6, 0x98e7, 0xe9d7, 0xe9d0, 0x98e8, /*0x38-0x3f*/
04367 0x98e9, 0x98ea, 0x98eb, 0x98ec, 0xe9cf, 0x98ed, 0x98ee, 0xc7c1, /*0x40-0x47*/
04368 0x98ef, 0x98f0, 0x98f1, 0x98f2, 0x98f3, 0x98f4, 0x98f5, 0x98f6, /*0x48-0x4f*/
04369 0xe9d2, 0x98f7, 0x98f8, 0x98f9, 0x98fa, 0x98fb, 0x98fc, 0x98fd, /*0x50-0x57*/
04370 0xe9d9, 0xb3c8, 0x98fe, 0xe9d3, 0x9940, 0x9941, 0x9942, 0x9943, /*0x58-0x5f*/
04371 0x9944, 0xcff0, 0x9945, 0x9946, 0x9947, 0xe9cd, 0x9948, 0x9949, /*0x60-0x67*/
04372 0x994a, 0x994b, 0x994c, 0x994d, 0x994e, 0x994f, 0x9950, 0x9951, /*0x68-0x6f*/
04373 0x9952, 0xb3f7, 0x9953, 0x9954, 0x9955, 0x9956, 0x9957, 0x9958, /*0x70-0x77*/
04374 0x9959, 0xe9d6, 0x995a, 0x995b, 0xe9da, 0x995c, 0x995d, 0x995e, /*0x78-0x7f*/
04375 0xccb4, 0x995f, 0x9960, 0x9961, 0xcfad, 0x9962, 0x9963, 0x9964, /*0x80-0x87*/
04376 0x9965, 0x9966, 0x9967, 0x9968, 0x9969, 0x996a, 0xe9d5, 0x996b, /*0x88-0x8f*/
04377 0xe9dc, 0xe9db, 0x996c, 0x996d, 0x996e, 0x996f, 0x9970, 0xe9de, /*0x90-0x97*/
04378 0x9971, 0x9972, 0x9973, 0x9974, 0x9975, 0x9976, 0x9977, 0x9978, /*0x98-0x9f*/
04379 0xe9d1, 0x9979, 0x997a, 0x997b, 0x997c, 0x997d, 0x997e, 0x9980, /*0xa0-0xaf*/
04380 0x9981, 0xe9dd, 0x9982, 0xe9df, 0xc3ca, 0x9983, 0x9984, 0x9985, /*0xab-0xb0*/
04381 0x9986, 0x9987, 0x9988, 0x9989, 0x998a, 0x998b, 0x998c, 0x998d, /*0xb1-0xb8*/
04382 0x998e, 0x998f, 0x9990, 0x9991, 0x9992, 0x9993, 0x9994, 0x9995, /*0xb9-0xc6*/
04383 0x9996, 0x9997, 0x9998, 0x9999, 0x999a, 0x999b, 0x999c, 0x999d, /*0xc7-0xcc*/
04384 0x999e, 0x999f, 0x99a0, 0x99a1, 0x99a2, 0x99a3, 0x99a4, 0x99a5, /*0xcd-0xca*/
04385 0x99a6, 0x99a7, 0x99a8, 0x99a9, 0x99aa, 0x99ab, 0x99ac, 0x99ad, /*0xcb-0xc8*/
04386 0x99ae, 0x99af, 0x99b0, 0x99b1, 0x99b2, 0x99b3, 0x99b4, 0x99b5, /*0xc9-0xce*/
04387 0x99b6, 0x99b7, 0x99b8, 0x99b9, 0x99ba, 0x99bb, 0x99bc, 0x99bd, /*0xcf-0xca*/
04388 0x99be, 0x99bf, 0x99c0, 0x99c1, 0x99c2, 0x99c3, 0x99c4, 0x99c5, /*0xcb-0xc8*/
04389 0x99c6, 0x99c7, 0x99c8, 0x99c9, 0x99ca, 0x99cb, 0x99cc, 0x99cd, /*0xd0-0xd7*/
04390 0x99ce, 0x99cf, 0x99d0, 0x99d1, 0x99d2, 0x99d3, 0x99d4, 0x99d5, /*0xd8-0xdd*/
04391 /* 0x6b00 */
04392 0x99d6, 0x99d7, 0x99d8, 0x99d9, 0x99da, 0x99db, 0x99dc, 0x99dd, /*0xe0-0xef*/
04393 0x99de, 0x99df, 0x99e0, 0x99e1, 0x99e2, 0x99e3, 0x99e4, 0x99e5, /*0xf0-0xf7*/
04394 0x99e6, 0x99ef, 0x99f0, 0x99f1, 0x99f2, 0x99f3, 0x99f4, 0x99f5, /*0xf8-0xff*/
04395 0x99fe, 0x99ff, 0x9a00, 0x9a01, 0x9a02, 0x9a03, 0x9a04, 0x9a05, /*0x00-0x07*/
04396 0xc7b7, 0xb4ce, 0xbbb6, 0xd0c0, 0xeca3, 0x99f6, 0x99f7, 0xc5b7, /*0x08-0x0f*/
04397 0x99f8, 0x99f9, 0x99fa, 0x99fb, 0x99fc, 0x99fd, 0x99fe, 0x99ff, /*0x10-0x17*/
04398 0x9a41, 0x9a42, 0xd3fb, 0x9a43, 0x9a44, 0x9a45, 0x9a46, 0xeca4, /*0x18-0x1f*/
04399 0x9a47, 0xeca5, 0xc6db, 0x9a48, 0x9a49, 0x9a4a, 0xbfee, 0x9a4b, /*0x20-0x27*/
04400 0x9a4c, 0x9a4d, 0x9a4e, 0xeca6, 0x9a4f, 0x9a50, 0xeca7, 0xd0aa, /*0x28-0x2f*/
04401 0x9a51, 0xc7b8, 0x9a52, 0x9a53, 0xb8e8, 0x9a54, 0x9a55, 0x9a56, /*0x30-0x37*/
04402 0x9a57, 0x9a58, 0x9a59, 0x9a5a, 0x9a5b, 0x9a5c, 0x9a5d, 0x9a5e, /*0x38-0x3f*/
04403 0x9a5f, 0xeca8, 0x9a60, 0x9a61, 0x9a62, 0x9a63, 0x9a64, 0x9a65, /*0x40-0x47*/
04404 0x9a66, 0x9a67, 0xd6b9, 0xd5fd, 0xb4cb, 0xb2bd, 0xcee4, 0xc6e7, /*0x48-0x4f*/
04405 0x9a68, 0x9a69, 0xcde1, 0x9a6a, 0x9a6b, 0x9a6c, 0x9a6d, 0x9a6e, /*0x50-0x57*/
04406 0x9a6f, 0x9a70, 0x9a71, 0x9a72, 0x9a73, 0x9a74, 0x9a75, 0x9a76, /*0x58-0x5f*/
04407 0x9a77, 0xb4f5, 0x9a78, 0xcbbc0, 0xbcdf, 0x9a79, 0x9a7a, 0x9a7b, /*0x60-0x67*/
04408 0x9a7c, 0xe9e2, 0xe9e3, 0xd1ea, 0xe9e5, 0x9a7d, 0xb4f9, 0xe9e4, /*0x68-0x6f*/
04409 0x9a7e, 0xd1b3, 0xcae2, 0xb2d0, 0x9a80, 0xe9e8, 0x9a81, 0x9a82, /*0x70-0x77*/
04410 0x9a83, 0x9a84, 0xe9e6, 0xe9e7, 0x9a85, 0x9a86, 0xd6b3, 0x9a87, /*0x78-0x7f*/
04411 0x9a88, 0x9a89, 0xe9e9, 0xe9ea, 0x9a8a, 0x9a8b, 0x9a8c, 0x9a8d, /*0x80-0x87*/
04412 0x9a8e, 0xe9eb, 0x9a8f, 0x9a90, 0x9a91, 0x9a92, 0x9a93, 0x9a94, /*0x88-0x8f*/
04413 0x9a95, 0x9a96, 0xe9ec, 0x9a97, 0x9a98, 0x9a99, 0x9a9a, 0x9a9b, /*0x90-0x97*/
04414 0x9a9c, 0x9a9d, 0x9a9e, 0xeca9, 0xc5b9, 0xb6ce, 0x9a9f, 0xd2f3, /*0x98-0x9f*/
04415 0x9aa0, 0x9aa1, 0x9aa2, 0x9aa3, 0x9aa4, 0x9aa5, 0x9aa6, 0xb5ee, /*0xa0-0xaf*/
04416 0x9aa7, 0xbdb9, 0xecb1, 0x9aa8, 0x9aa9, 0xd2e3, 0x9aaa, 0x9aab, /*0xab-0xba*/
04417 0x9aac, 0x9aad, 0x9aae, 0xcee3, 0x9aaf, 0xc4b8, 0x9ab0, 0xc3bf, /*0xbc-0xc9*/
04418 0x9ab1, 0x9ab2, 0xb6be, 0xd8b9, 0xb1c8, 0xb1cf, 0xb1d1, 0xc5fe, /*0xca-0xcb*/
04419 0x9ab3, 0xb1d0, 0x9ab4, 0xc3ab, 0x9ab5, 0x9ab6, 0x9ab7, 0x9ab8, /*0xcc-0xd1*/
04420 0x9ab9, 0xd5b1, 0x9aba, 0x9abb, 0x9abc, 0x9abd, 0x9abe, 0x9abf, /*0xd2-0xd9*/
04421 0x9ac0, 0x9ac1, 0xeba4, 0xbac1, 0x9ac2, 0x9ac3, 0x9ac4, 0xccba, /*0xe0-0xef*/
04422 0x9ac5, 0x9ac6, 0x9ac7, 0xeba5, 0x9ac8, 0xeba7, 0x9ac9, 0x9aca, /*0xf0-0xf7*/
04423 0x9acb, 0xeba8, 0x9acc, 0x9acd, 0x9ace, 0xeba6, 0x9acf, 0x9ad0, /*0xf8-0xff*/
04424 /* 0x6c00 */
04425 0x9ad1, 0x9ad2, 0x9ad3, 0x9ad4, 0x9ad5, 0xeba9, 0xebab, 0xebaa, /*0x00-0x07*/
04426 0x9ad6, 0x9ad7, 0x9ad8, 0x9ad9, 0x9ada, 0xebac, 0x9adb, 0xcacf, /*0x08-0x0f*/
04427 0xd8b5, 0xc3f1, 0x9adc, 0xc3a5, 0xc6f8, 0xebad, 0xc4ca, 0x9add, /*0x10-0x17*/
04428 0xebae, 0xebaf, 0xebb0, 0xb7d5, 0x9ade, 0x9adf, 0x9ae0, 0xb7fa, /*0x18-0x1f*/
04429 0x9ae1, 0xebb1, 0xc7e2, 0x9ae2, 0xebb3, 0x9ae3, 0xbaa4, 0xd1f5, /*0x20-0x27*/
04430 0xb0b1, 0xebb2, 0xebb4, 0x9ae4, 0x9ae5, 0x9ae6, 0xb5aa, 0xc2c8, /*0x28-0x2f*/
04431 0xc7e8, 0x9ae7, 0xebb5, 0x9ae8, 0xcbae, 0xe3df, 0x9ae9, 0x9aea, /*0x30-0x37*/
04432 0xd3c0, 0x9aeb, 0x9aeb, 0x9aed, 0x9aee, 0xd9db, 0x9aef, 0x9af0, /*0x38-0x3f*/
04433 0xcdal, 0xd6ad, 0xc7f3, 0x9af1, 0x9af2, 0x9af3, 0xd9e0, 0xbbe3, /*0x40-0x47*/
04434 0x9af4, 0xbaba, 0xe3e2, 0x9af5, 0x9af6, 0x9af7, 0x9af8, 0x9af9, /*0x48-0x4f*/
04435 0xcfab, 0x9afa, 0x9afb, 0x9afc, 0xe3e0, 0xc9c7, 0x9afd, 0xbab9, /*0x50-0x5f*/
04436 0x9afe, 0x9b40, 0x9b41, 0xd1b4, 0xe3e1, 0xc8ea, 0xb9af, 0xbdad, /*0x60-0x67*/
04437 0xb3d8, 0xcdbd, 0x9b42, 0x9b43, 0xcc00, 0x9b44, 0x9b45, 0x9b46, /*0x68-0x6f*/
04438 0xe3e8, 0xe3e9, 0xcdf4, 0x9b47, 0x9b48, 0x9b49, 0x9b4a, 0x9b4b, /*0x70-0x77*/
04439 0xccad, 0x9b4c, 0xbcb3, 0x9b4d, 0xe3ea, 0x9b4e, 0xe3eb, 0x9b4f, /*0x78-0x7f*/
04440 0x9b50, 0xd0da, 0x9b51, 0x9b52, 0x9b53, 0xc6fb, 0xb7da, 0x9b54, /*0x80-0x8f*/
04441 0x9b55, 0xc7df, 0xd2ca, 0xc6d6, 0x9b56, 0xe3ea, 0xe3ec, 0x9b57, /*0x90-0x9f*/
04442 0xc9f2, 0xb3c1, 0x9b58, 0x9b59, 0xe3e7, 0x9b5a, 0x9b5b, 0xc6e3, /*0xa0-0xaf*/

```

```
04443 0xe3e5, 0x9b5c, 0x9b5d, 0xedb3, 0xe3e6, 0x9b5e, 0x9b5f, 0x9b60, /*0x90-0x97*/
04444 0x9b61, 0xc9b3, 0x9b62, 0xc5e6, 0x9b63, 0x9b64, 0x9b65, 0xb9b5, /*0x98-0x9f*/
04445 0x9b66, 0xc3bb, 0x9b67, 0xe3e3, 0xc5bd, 0xc1a4, 0xc2d9, 0xb2d7, /*0xa0-0xa7*/
04446 0x9b68, 0xe3ed, 0xbba6, 0xc4ad, 0x9b69, 0xe3f0, 0xbeda, 0x9b6a, /*0xa8-0xaf*/
04447 0x9b6b, 0xe3fb, 0xe3f5, 0xbad3, 0x9b6c, 0x9b6d, 0x9b6e, 0x9b6f, /*0xb0-0xb7*/
04448 0xb7d0, 0xd3cd, 0x9b70, 0xd6ce, 0xd5d3, 0xb9c1, 0xd5b4, 0xd1d8, /*0xb8-0xbf*/
04449 0x9b71, 0x9b72, 0x9b73, 0x9b74, 0xd0b9, 0xc7f6, 0x9b75, 0x9b76, /*0xc0-0xc7*/
04450 0x9b77, 0xc8aa, 0xb2b4, 0x9b78, 0xc3da, 0x9b79, 0x9b7a, 0x9b7b, /*0xc8-0xcf*/
04451 0xe3ee, 0x9b7c, 0x9b7d, 0xe3fc, 0xe3ef, 0xb7a8, 0xe3f7, 0xe3f4, /*0xd0-0xd7*/
04452 0x9b7e, 0x9b80, 0x9b81, 0xb7ba, 0x9b82, 0x9b83, 0xc5a2, 0x9b84, /*0xd8-0xdf*/
04453 0xe3f6, 0xc5dd, 0xb2a8, 0xc6fc, 0x9b85, 0xc4e0, 0x9b86, 0x9b87, /*0xe0-0xe7*/
04454 0xd7a2, 0x9b88, 0xc0e1, 0xe3f9, 0x9b89, 0x9b8a, 0xe3fa, 0xe3fd, /*0xe8-0xef*/
04455 0xcc9a, 0xe3f3, 0x9b8b, 0xd3be, 0x9b8c, 0xb1c3, 0xedb4, 0xe3f1, /*0xf0-0xf7*/
04456 0xe3f2, 0x9b8d, 0xe3f8, 0xd0ba, 0xc6c3, 0xd4f3, 0xe3fe, 0x9b8e, /*0xf8-0xff*/
04457 /* 0x6d00 */
04458 0x9b8f, 0xbde0, 0x9b90, 0x9b91, 0xe4a7, 0x9b92, 0x9b93, 0xe4a6, /*0x00-0x07*/
04459 0x9b94, 0x9b95, 0x9b96, 0xd1f3, 0xe4a3, 0x9b97, 0xe4a9, 0x9b98, /*0x08-0x0f*/
04460 0x9b99, 0xd9b9, 0xc8f7, 0x9b9b, 0x9b9c, 0x9b9d, 0x9b9e, 0xcfb4, /*0x10-0x17*/
04461 0x9b9f, 0xe4a8, 0xe4ae, 0xc2e5, 0x9ba0, 0x9ba1, 0xb6b4, 0x9ba2, /*0x18-0x1f*/
04462 0x9ba3, 0x9ba4, 0x9ba5, 0x9ba6, 0x9ba7, 0xbdf2, 0x9ba8, 0xe4a2, /*0x20-0x27*/
04463 0x9ba9, 0x9baa, 0xbae9, 0xe4aa, 0x9bab, 0x9bac, 0xe4ac, 0x9bad, /*0x28-0x2f*/
04464 0x9bae, 0xb6fd, 0xd6de, 0xe4b2, 0x9baf, 0xe4ad, 0x9bb0, 0x9bb1, /*0x30-0x37*/
04465 0x9bb2, 0xe4a1, 0x9bb3, 0xbbee, 0xcddd, 0xc7a2, 0xc5c9, 0x9bb4, /*0x38-0x3f*/
04466 0x9bb5, 0xc1f7, 0x9bb6, 0xe4a4, 0x9bb7, 0xc7b3, 0xbdac, 0xbdbd, /*0x40-0x47*/
04467 0xe4a5, 0x9bb8, 0xd7c7, 0xb2e2, 0x9bb9, 0xe4ab, 0xbcc3, 0xe4af, /*0x48-0x4f*/
04468 0x9bba, 0xbbeb, 0xe4b0, 0xc5a8, 0xe4b1, 0x9bbb, 0x9bbc, 0x9bbd, /*0x50-0x57*/
04469 0x9bbe, 0xd5e3, 0xcfa3, 0x9bbf, 0xe4ba, 0x9bc0, 0xe4b7, 0x9bc1, /*0x58-0x5f*/
04470 0xe4bb, 0x9bc2, 0x9bc3, 0xe4bd, 0x9bc4, 0x9bc5, 0xc6d6, 0x9bc6, /*0x60-0x67*/
04471 0x9bc7, 0xbac6, 0xc0cb, 0x9bc8, 0x9bc9, 0x9bca, 0xb8a1, 0xe4b4, /*0x68-0x6f*/
04472 0x9bcd, 0x9bcc, 0x9bcd, 0x9bce, 0xd4a1, 0x9bcf, 0x9bd0, 0xbaa3, /*0x70-0x77*/
04473 0xbdfc, 0x9bd1, 0x9bd2, 0x9bd3, 0xe4bc, 0x9bd4, 0x9bd5, 0x9bd6, /*0x78-0x7f*/
04474 0x9bd7, 0x9bd8, 0xcdbf, 0x9bd9, 0x9bda, 0xc4f9, 0x9bdb, 0x9bdc, /*0x80-0x87*/
04475 0xcfffb, 0xc9e6, 0x9bdd, 0x9bde, 0xd3bf, 0x9bdf, 0xcfd1, 0x9be0, /*0x88-0x8f*/
04476 0x9be1, 0xe4b3, 0x9be2, 0xe4b8, 0xe4b9, 0xcce9, 0x9be3, 0x9be4, /*0x90-0x97*/
04477 0x9be5, 0x9be6, 0x9be7, 0xcce, 0x9be8, 0xc0d4, 0xe4b5, 0xc1b0, /*0x98-0x9f*/
04478 0xe4b6, 0xcdd0, 0x9be9, 0xbbc1, 0xb5d3, 0x9bea, 0xc8f3, 0xbda7, /*0xa0-0xa7*/
04479 0xd5c7, 0xc9ac, 0xb8a2, 0xe4ca, 0x9beb, 0x9bec, 0xe4cc, 0xd1c4, /*0xa8-0xaf*/
04480 0x9bed, 0x9bee, 0xd2ba, 0x9bef, 0x9bf0, 0xbaad, 0x9bf1, 0x9bf2, /*0xb0-0xb7*/
04481 0xbad4, 0x9bf3, 0x9bf4, 0x9bf5, 0x9bf6, 0x9bf7, 0x9bf8, 0xe4c3, /*0xb8-0xbf*/
04482 0xb5ed, 0x9bf9, 0x9bfa, 0x9bfb, 0xd7cd, 0xe4c0, 0xcffd, 0xe4bf, /*0xc0-0xc7*/
04483 0x9bfc, 0x9bfd, 0x9bfe, 0xc1cd, 0xcceca, 0x9c40, 0x9c41, 0x9c42, /*0xc8-0xcf*/
04484 0x9c43, 0xc9e7, 0xc9e8, 0x9c44, 0x9c45, 0x9c46, 0x9c47, 0xc4d7, 0x9c48, /*0xd0-0xd7*/
04485 0xc9cd4, 0xe4c8, 0x9c49, 0x9c4a, 0x9c4b, 0xe4c7, 0xe4c1, 0x9c4c, /*0xd8-0xdf*/
04486 0xe4c4, 0xb5ad, 0x9c4d, 0x9c4e, 0xd3d9, 0x9c4f, 0xe4c6, 0x9c50, /*0xe0-0xe7*/
04487 0x9c51, 0x9c52, 0x9c53, 0xd2f9, 0xb4e3, 0x9c54, 0xbbb4, 0x9c55, /*0xe8-0xef*/
04488 0x9c56, 0xc9ee, 0x9c57, 0xb4be, 0x9c58, 0x9c59, 0x9c5a, 0xbbec, /*0xf0-0xf7*/
04489 0x9c5b, 0xd1cd, 0x9c5c, 0xccecd, 0xedb5, 0x9c5d, 0x9c5e, 0x9c5f, /*0xf8-0xff*/
04490 /* 0x6e00 */
04491 0x9c60, 0x9c61, 0x9c62, 0x9c63, 0x9c64, 0xc7e5, 0x9c65, 0x9c66, /*0x00-0x07*/
04492 0x9c67, 0x9c68, 0xd4a8, 0x9c69, 0xe4cb, 0xd7d5, 0xe4c2, 0x9c6a, /*0x08-0x0f*/
04493 0xbda5, 0xe4c5, 0x9c6b, 0x9c6c, 0xd3e6, 0x9c6d, 0xe4c9, 0xc9f8, /*0x10-0x17*/
04494 0x9c6e, 0x9c6f, 0xe4be, 0x9c70, 0x9c71, 0xd3e5, 0x9c72, 0x9c73, /*0x18-0x1f*/
04495 0xc7fe, 0xb6c9, 0x9c74, 0xd4fc, 0xb2b3, 0xe4d7, 0x9c75, 0x9c76, /*0x20-0x27*/
04496 0x9c77, 0xcce2, 0x9c78, 0xe4cd, 0x9c79, 0xccebc, 0x9c7a, 0xb8db, /*0x28-0x2f*/
04497 0x9c7b, 0x9c7c, 0xe4d6, 0x9c7d, 0xbfca, 0x9c7e, 0x9c80, 0x9c81, /*0x30-0x37*/
04498 0xd3ce, 0x9c82, 0xc3ec, 0x9c83, 0x9c84, 0x9c85, 0x9c86, 0x9c87, /*0x38-0x3f*/
04499 0x9c88, 0x9c89, 0xc98a, 0xc5c8, 0x9c8b, 0x9c8c, 0x9c8d, 0x9c8e, /*0x40-0x47*/
04500 0x9c8e, 0x9c8f, 0x9c90, 0x9c91, 0x9c92, 0xc9cd4, 0xe4cf, 0x9c93, /*0x48-0x4f*/
04501 0x9c94, 0x9c95, 0x9c96, 0xe4d4, 0xe4d5, 0x9c97, 0xbafe, 0x9c98, /*0x50-0x57*/
04502 0xcfe6, 0x9c99, 0x9c9a, 0xd5bf, 0x9c9b, 0x9c9c, 0x9c9d, 0xe4d2, /*0x58-0x5f*/
04503 0x9c9e, 0x9c9f, 0x9ca0, 0x9ca1, 0x9ca2, 0x9ca3, 0x9ca4, 0x9ca5, /*0x60-0x67*/
04504 0x9ca6, 0x9ca7, 0x9ca8, 0xe4d0, 0x9ca9, 0x9caa, 0xe4ce, 0x9cab, /*0x68-0x6f*/
04505 0x9cac, 0x9cad, 0x9cae, 0x9caf, 0x9cb0, 0x9cb1, 0x9cb2, 0x9cb3, /*0x70-0x77*/
04506 0x9cb4, 0x9cb5, 0x9cb6, 0x9cb7, 0x9cb8, 0x9cb9, 0xcde5, 0xc9aa, /*0x78-0x7f*/
04507 0x9cba, 0x9cbb, 0x9cbc, 0xc0a3, 0x9cbd, 0xbda6, 0xe4d3, 0x9cbe, /*0x80-0x87*/
04508 0x9cbf, 0xb8c8, 0x9cc0, 0x9cc1, 0x9cc2, 0x9cc3, 0x9cc4, 0xe4e7, /*0x88-0x8f*/
04509 0xd4b4, 0x9cc5, 0x9cc6, 0x9cc7, 0x9cc8, 0x9cc9, 0x9cca, 0x9ccb, /*0x90-0x97*/
04510 0xe4db, 0x9ccc, 0x9ccd, 0x9cce, 0xc1ef, 0x9ccf, 0x9cd0, 0xe4e9, /*0x98-0x9f*/
04511 0x9cd1, 0x9cd2, 0xd2e7, 0x9cd3, 0x9cd4, 0xe4df, 0x9cd5, 0xe4e0, /*0xa0-0xa7*/
04512 0x9cd6, 0x9cd7, 0xcfaa, 0x9cd8, 0x9cd9, 0x9cda, 0x9cdb, 0xcdbd, /*0xa8-0xaf*/
04513 0x9cdc, 0xe4da, 0xe4d1, 0x9cdd, 0xe4e5, 0x9cde, 0xc8dc, 0xe4e3, /*0xb0-0xb7*/
04514 0x9cdf, 0x9ce0, 0xe4e7, 0xe4e2, 0x9ce1, 0x9ce2, 0x9ce3, 0x9ce4, /*0xb8-0xbf*/
04515 0x9ce4, 0xb3fc, 0xe4e8, 0x9ce5, 0x9ce6, 0x9ce7, 0x9ce8, 0xb5e1, /*0xc0-0xc7*/
04516 0x9ce9, 0x9cea, 0x9ceb, 0xd7cc, 0x9cec, 0x9ced, 0x9cee, 0xe4e6, /*0xc8-0xcf*/
04517 0x9cef, 0xbbac, 0x9cf0, 0xd7d2, 0xc9cf, 0xebf8, 0x9cf1, 0xe4e4, /*0xd0-0xd7*/
04518 0x9cf2, 0x9cf3, 0xb9f6, 0x9cf4, 0x9cf5, 0x9cf6, 0xd6cd, 0xe4d9, /*0xd8-0xdf*/
04519 0xe4dc, 0xc2fa, 0x9cf9, 0x9cfa, 0x9cfb, 0x9cfc, 0x9cf8, 0x9cfe, /*0xe0-0xe7*/
04520 0xb1f5, 0xc9cb2, 0x9cf9, 0x9cfa, 0x9cfb, 0x9cfc, 0x9cfd, 0x9cfe, /*0xe8-0xef*/
04521 0x9d40, 0x9d41, 0x9d42, 0x9d43, 0xb5ce, 0x9d44, 0x9d45, 0x9d46, /*0xf0-0xf7*/
04522 0x9d47, 0xe4ef, 0x9d48, 0x9d49, 0x9d4a, 0x9d4b, 0x9d4c, 0x9d4d, /*0xf8-0xff*/
04523 /* 0x6f00 */
04524 0x9d4e, 0x9d4f, 0xc6af, 0x9d50, 0x9d51, 0x9d52, 0xc6e1, 0x9d53, /*0x00-0x07*/
04525 0x9d54, 0xe4f5, 0x9d55, 0x9d56, 0x9d57, 0x9d58, 0x9d59, 0xc2a9, /*0x08-0x0f*/
04526 0x9d5a, 0x9d5b, 0x9d5c, 0xc0ec, 0xd1dd, 0xe4ee, 0x9d5d, 0x9d5e, /*0x10-0x17*/
04527 0x9d5f, 0x9d60, 0x9d61, 0x9d62, 0x9d63, 0x9d64, 0x9d65, 0x9d66, /*0x18-0x1f*/
04528 0xc4ae, 0x9d67, 0x9d68, 0x9d69, 0xe4ed, 0x9d6a, 0x9d6b, 0x9d6c, /*0x20-0x27*/
04529 0x9d6d, 0xe4f6, 0xe4f4, 0xc2fe, 0x9d6e, 0xe4dd, 0x9d6f, 0xe4f0, /*0x28-0x2f*/
```

```
04530 0x9d70, 0xcafe, 0x9d71, 0xd5c4, 0x9d72, 0x9d73, 0xe4f1, 0x9d74, /*0x30-0x37*/
04531 0x9d75, 0x9d76, 0x9d77, 0x9d78, 0x9d79, 0x9d7a, 0xd1fa, 0x9d7b, /*0x38-0x3f*/
04532 0x9d7c, 0x9d7d, 0x9d7e, 0x9d80, 0x9d81, 0x9d82, 0xe4eb, 0xe4ec, /*0x40-0x47*/
04533 0x9d83, 0x9d84, 0x9d85, 0xe4f2, 0x9d86, 0xceab, 0x9d87, 0x9d88, /*0x48-0x4f*/
04534 0x9d89, 0x9d8a, 0x9d8b, 0x9d8c, 0x9d8d, 0x9d8e, 0x9d8f, 0x9d90, /*0x50-0x57*/
04535 0xc5cb, 0x9d91, 0x9d92, 0x9d93, 0xc7b1, 0x9d94, 0xc2ba, 0x9d95, /*0x58-0x5f*/
04536 0x9d96, 0x9d97, 0xe4ea, 0x9d98, 0x9d99, 0x9d9a, 0xc1ca, 0x9d9b, /*0x60-0x67*/
04537 0x9d9c, 0x9d9d, 0x9d9e, 0x9d9f, 0x9da0, 0xccb6, 0xb3b1, 0x9da1, /*0x68-0x6f*/
04538 0x9da2, 0x9da3, 0xe4fb, 0x9da4, 0xe4f3, 0x9da5, 0x9da6, 0x9da7, /*0x70-0x77*/
04539 0xe4fa, 0x9da8, 0xe4fd, 0x9da9, 0xe4fc, 0x9daa, 0x9dab, 0x9dac, /*0x78-0x7f*/
04540 0x9dad, 0x9dae, 0x9daf, 0x9db0, 0xb3ce, 0x9db1, 0x9db2, 0x9db3, /*0x80-0x87*/
04541 0xb3ba, 0xe4f7, 0x9db4, 0x9db5, 0xe4f9, 0xe4f8, 0xc5ec, 0x9db6, /*0x88-0x8f*/
04542 0x9db7, 0x9db8, 0x9db9, 0x9dba, 0x9dbb, 0x9dbc, 0x9dbd, 0x9dbe, /*0x90-0x97*/
04543 0x9dbf, 0x9dc0, 0x9dc1, 0x9dc2, 0xc0bd, 0x9dc3, 0x9dc4, 0x9dc5, /*0x98-0x9f*/
04544 0x9dc6, 0xd4e8, 0x9dc7, 0x9dc8, 0x9dc9, 0xdca, 0x9dcb, 0xe5a2, /*0xa0-0xaf*/
04545 0x9dcc, 0x9dcd, 0x9dce, 0x9dcf, 0x9dd0, 0x9dd1, 0x9dd2, 0x9dd3, /*0xa8-0xaf*/
04546 0x9dd4, 0x9dd5, 0x9dd6, 0xb0c4, 0x9dd7, 0x9dd8, 0xe5a4, 0x9dd9, /*0xb0-0xb7*/
04547 0x9dda, 0xe5a3, 0x9ddb, 0x9ddc, 0x9dde, 0x9dde, 0x9de0, 0x9de1, /*0xb8-0xbf*/
04548 0xbca4, 0x9de1, 0xe5a5, 0x9de2, 0x9de3, 0x9de4, 0x9de5, 0x9de6, /*0xc0-0xc7*/
04549 0x9de7, 0xe5a1, 0x9de8, 0x9de9, 0x9dea, 0x9deb, 0x9dec, 0x9ded, /*0xc8-0xcf*/
04550 0x9dee, 0xc5eb, 0xb1f4, 0x9def, 0x9df0, 0x9df1, 0x9df2, 0x9df3, /*0xd0-0xd7*/
04551 0x9df4, 0x9df5, 0x9df6, 0x9df7, 0x9df8, 0x9df9, 0xe5a8, 0x9dfa, /*0xd8-0xdf*/
04552 0xe5a9, 0xe5a6, 0x9dfb, 0x9dfc, 0x9dfd, 0x9dfe, 0x9e40, 0x9e41, /*0xe0-0xe7*/
04553 0x9e42, 0x9e43, 0x9e44, 0x9e45, 0x9e46, 0x9e47, 0xe5a7, 0xe5aa, /*0xe8-0xef*/
04554 0x9e48, 0x9e49, 0x9e4a, 0x9e4b, 0x9e4c, 0x9e4d, 0x9e4e, 0x9e4f, /*0xf0-0xf7*/
04555 0x9e50, 0x9e51, 0x9e52, 0x9e53, 0x9e54, 0x9e55, 0x9e56, 0x9e57, /*0xf8-0xff*/
04556 /* 0x7000 */
04557 0x9e58, 0x9e59, 0x9e5a, 0x9e5b, 0x9e5c, 0x9e5d, 0x9e5e, 0x9e5f, /*0x00-0x07*/
04558 0x9e60, 0x9e61, 0x9e62, 0x9e63, 0x9e64, 0x9e65, 0x9e66, 0x9e67, /*0x08-0x0f*/
04559 0x9e68, 0xc6d9, 0x9e69, 0x9e6a, 0x9e6b, 0x9e6c, 0x9e6d, 0x9e6e, /*0x10-0x17*/
04560 0x9e6f, 0x9e70, 0xe5ab, 0xe5ad, 0x9e71, 0x9e72, 0x9e73, 0x9e74, /*0x18-0x1f*/
04561 0x9e75, 0x9e76, 0x9e77, 0xe5ac, 0x9e78, 0x9e79, 0x9e7a, 0x9e7b, /*0x20-0x27*/
04562 0x9e7c, 0x9e7d, 0x9e7e, 0x9e80, 0x9e81, 0x9e82, 0x9e83, 0x9e84, /*0x28-0x2f*/
04563 0x9e85, 0x9e86, 0x9e87, 0x9e88, 0x9e89, 0xe5af, 0x9e8a, 0x9e8b, /*0x30-0x37*/
04564 0x9e8c, 0xe5ae, 0x9e8d, 0x9e8e, 0x9e8f, 0x9e90, 0x9e91, 0x9e92, /*0x38-0x3f*/
04565 0x9e93, 0x9e94, 0x9e95, 0x9e96, 0x9e97, 0x9e98, 0x9e99, 0x9e9a, /*0x40-0x47*/
04566 0x9e9b, 0x9e9c, 0x9e9d, 0x9e9e, 0xb9e0, 0x9e9f, 0x9ea0, 0xe5b0, /*0x48-0x4f*/
04567 0x9ea1, 0x9ea2, 0x9ea3, 0x9ea4, 0x9ea5, 0x9ea6, 0x9ea7, 0x9ea8, /*0x50-0x57*/
04568 0x9ea9, 0x9eaa, 0x9eab, 0x9eac, 0x9ead, 0x9eae, 0xe5b1, 0x9eaf, /*0x58-0x5f*/
04569 0x9eb0, 0x9eb1, 0x9eb2, 0x9eb3, 0x9eb4, 0x9eb5, 0x9eb6, 0x9eb7, /*0x60-0x67*/
04570 0x9eb8, 0x9eb9, 0x9eba, 0xbbf0, 0xece1, 0xc3f0, 0x9ebb, 0xb5c6, /*0x68-0x6f*/
04571 0xbdb2, 0x9ebc, 0x9ebd, 0x9ebe, 0x9ebf, 0xc1e9, 0xd4ee, 0x9ec0, /*0x70-0x77*/
04572 0xbec4, 0x9ec1, 0x9ec2, 0x9ec3, 0xd7c6, 0x9ec4, 0xd4d6, 0xb2d3, /*0x78-0x7f*/
04573 0xecbe, 0x9ec5, 0x9ec6, 0x9ec7, 0x9ec8, 0xeac1, 0x9ec9, 0x9eca, /*0x80-0x87*/
04574 0x9ecb, 0xc2af, 0xb4b6, 0x9ecc, 0x9ecd, 0x9ece, 0xd1d7, 0x9ecf, /*0x88-0x8f*/
04575 0x9ed0, 0x9ed1, 0xb3b4, 0x9ed2, 0xc8b2, 0xbfb3, 0xecc0, 0x9ed3, /*0x90-0x97*/
04576 0x9ed4, 0xd6cb, 0x9ed5, 0x9ed6, 0xecbf, 0xecc1, 0x9ed7, 0x9ed8, /*0x98-0x9f*/
04577 0x9ed9, 0x9eda, 0x9edb, 0x9edc, 0x9edd, 0x9ede, 0x9edf, 0x9ee0, /*0xa0-0xaf*/
04578 0x9ee1, 0x9ee2, 0x9ee3, 0xecc5, 0xbec6, 0xcbbf, 0xc5da, 0xbecb, /*0xa8-0xaf*/
04579 0x9ee4, 0xecc6, 0x9ee5, 0xb1fe, 0x9ee6, 0x9ee7, 0x9ee8, 0xecc4, /*0xb0-0xb7*/
04580 0xd5a8, 0xb5e3, 0x9ee9, 0xecc2, 0xc1b6, 0xb3e3, 0x9eea, 0x9eeb, /*0xb8-0xbf*/
04581 0xecc3, 0xcbb8, 0xc0c3, 0xcce, 0x9eec, 0x9eed, 0x9eee, 0x9eef, /*0xc0-0xc7*/
04582 0xc1cd, 0x9ef0, 0xecc8, 0x9ef1, 0x9ef2, 0x9ef3, 0x9ef4, 0x9ef5, /*0xc8-0xcf*/
04583 0x9ef6, 0x9ef7, 0x9ef8, 0x9ef9, 0x9efa, 0x9efb, 0x9efc, 0x9efd, /*0xd0-0xd7*/
04584 0xbae6, 0xc0d3, 0x9efe, 0xd6f2, 0x9f40, 0x9f41, 0x9f42, 0xd1cc, /*0xd8-0xdf*/
04585 0x9f43, 0x9f44, 0x9f45, 0x9f46, 0xbfb6, 0x9f47, 0xb7b3, 0xc9d5, /*0xe0-0xe7*/
04586 0xecc7, 0xbbe2, 0x9f48, 0xcce, 0xbddf, 0xc8c8, 0x9f49, 0xcfa9, /*0xe8-0xef*/
04587 0x9f4a, 0x9f4b, 0x9f4c, 0x9f4d, 0x9f4e, 0x9f4f, 0x9f50, 0xcde9, /*0xf0-0xf7*/
04588 0x9f51, 0xc5eb, 0x9f52, 0x9f53, 0x9f54, 0xb7e9, 0x9f55, 0x9f56, /*0xf8-0xff*/
04589 /* 0x7100 */
04590 0x9f57, 0x9f58, 0x9f59, 0x9f5a, 0x9f5b, 0x9f5c, 0x9f5d, 0x9f5e, /*0x00-0x07*/
04591 0x9f5f, 0xd1c9, 0xbab8, 0x9f60, 0x9f61, 0x9f62, 0x9f63, 0x9f64, /*0x08-0x0f*/
04592 0xecc9, 0x9f65, 0x9f66, 0xecca, 0x9f67, 0xbbc0, 0xeccb, 0x9f68, /*0x10-0x17*/
04593 0xece2, 0xb1ba, 0xb7d9, 0x9f69, 0x9f6a, 0x9f6b, 0x9f6c, 0x9f6d, /*0x18-0x1f*/
04594 0x9f6e, 0x9f6f, 0x9f70, 0x9f71, 0x9f72, 0x9f73, 0xbdb9, 0x9f74, /*0x20-0x27*/
04595 0x9f75, 0x9f76, 0x9f77, 0x9f78, 0x9f79, 0x9f7a, 0x9f7b, 0xeccc, /*0x28-0x2f*/
04596 0xd1e6, 0xeccd, 0x9f7c, 0x9f7d, 0x9f7e, 0x9f80, 0xc8bb, 0x9f81, /*0x30-0x37*/
04597 0x9f82, 0x9f83, 0x9f84, 0x9f85, 0x9f86, 0x9f87, 0x9f88, 0x9f89, /*0x38-0x3f*/
04598 0x9f8a, 0x9f8b, 0x9f8c, 0x9f8d, 0x9f8e, 0xeecd, 0x9f8f, 0x9f90, /*0x40-0x47*/
04599 0x9f91, 0x9f92, 0xeecd, 0x9f93, 0xbbcd, 0x9f94, 0xbce5, 0x9f95, /*0x48-0x4f*/
04600 0x9f96, 0x9f97, 0x9f98, 0x9f99, 0x9f9a, 0x9f9b, 0x9f9c, 0x9f9d, /*0x50-0x57*/
04601 0x9f9e, 0x9f9f, 0x9fa0, 0x9fa1, 0xeccf, 0x9fa2, 0xc9b7, 0x9fa3, /*0x58-0x5f*/
04602 0x9fa4, 0x9fa5, 0x9fa6, 0x9fa7, 0xc3ba, 0x9fa8, 0xece3, 0xd5d5, /*0x60-0x67*/
04603 0xeecd, 0x9fa9, 0x9faa, 0x9fab, 0x9fac, 0x9fad, 0xd6f3, 0x9fae, /*0x68-0x6f*/
04604 0x9faf, 0x9fb0, 0xeecd, 0xeccc, 0x9fb1, 0x9fb2, 0x9fb3, 0x9fb4, /*0x70-0x77*/
04605 0xeecd, 0x9fb5, 0xeecd, 0x9fb6, 0x9fb7, 0xc9bf, 0x9fb8, 0x9fb9, /*0x78-0x7f*/
04606 0x9fba, 0x9fbb, 0x9fbc, 0x9fbd, 0xcfa8, 0x9fbe, 0x9fbf, 0x9fc0, /*0x80-0x87*/
04607 0x9fc1, 0x9fc2, 0xd0dc, 0x9fc3, 0x9fc4, 0x9fc5, 0x9fc6, 0xd1ac, /*0x88-0x8f*/
04608 0x9fc7, 0x9fc8, 0x9fc9, 0x9fca, 0xc8db, 0x9fcb, 0x9fcc, 0x9fcd, /*0x90-0x97*/
04609 0xeecd, 0xcce5, 0x9fce, 0x9fcf, 0x9fd0, 0x9fd1, 0x9fd2, 0xcaec, /*0x98-0x9f*/
04610 0xeecd, 0x9fd3, 0x9fd4, 0x9fd5, 0x9fd6, 0x9fd7, 0x9fd8, 0x9fd9, /*0xa0-0xaf*/
04611 0xeecd, 0x9fda, 0x9fdb, 0x9fdc, 0xb0be, 0x9fdd, 0x9fde, 0x9fdf, /*0xa8-0xaf*/
04612 0x9fe0, 0x9fe1, 0x9fe2, 0x9fe3, 0xeecd, 0x9fe4, 0x9fe5, 0x9fe6, /*0xb0-0xb7*/
04613 0x9fe7, 0xece4, 0x9fe8, 0x9fe9, 0x9fea, 0x9feb, 0x9fec, 0x9fed, /*0xb8-0xbf*/
04614 0x9fed, 0x9fee, 0x9fef, 0xc8bc, 0x9ff0, 0x9ff1, 0x9ff2, 0x9ff3, /*0xc0-0xc7*/
04615 0x9ffa, 0x9fff, 0x9fff, 0x9fff, 0x9fff, 0xc1c7, 0x9ffa, 0x9ffa, /*0xc8-0xcf*/
04616 0x9ffb, 0x9ffc, 0x9ffd, 0x9ffe, 0xeecd, 0xd1e0, 0xa040, 0xa041, /*0xd0-0xd7*/
```

```
04617 0xa042, 0xa043, 0xa044, 0xa045, 0xa046, 0xa047, 0xa048, 0xa049, /*0xd8-0xdf*/
04618 0xecdb, 0xa04a, 0xa04b, 0xa04c, 0xa04d, 0xd4ef, 0xa04e, 0xecdd, /*0xe0-0xe7*/
04619 0xa04f, 0xa050, 0xa051, 0xa052, 0xa053, 0xa054, 0xdbc6, 0xa055, /*0xe8-0xef*/
04620 0xa056, 0xa057, 0xa058, 0xa059, 0xa05a, 0xa05b, 0xa05c, 0xa05d, /*0xf0-0xf7*/
04621 0xa05e, 0xecde, 0xa05f, 0xa060, 0xa061, 0xa062, 0xa063, 0xa064, /*0xf8-0xff*/
04622 /* 0x7200 */
04623 0xa065, 0xa066, 0xa067, 0xa068, 0xa069, 0xa06a, 0xb1ac, 0xa06b, /*0x00-0x07*/
04624 0xa06c, 0xa06d, 0xa06e, 0xa06f, 0xa070, 0xa071, 0xa072, 0xa073, /*0x08-0x0f*/
04625 0xa074, 0xa075, 0xa076, 0xa077, 0xa078, 0xa079, 0xa07a, 0xa07b, /*0x10-0x17*/
04626 0xa07c, 0xa07d, 0xa07e, 0xa080, 0xa081, 0xecdf, 0xa082, 0xa083, /*0x18-0x1f*/
04627 0xa084, 0xa085, 0xa086, 0xa087, 0xa088, 0xa089, 0xa08a, 0xa08b, /*0x20-0x27*/
04628 0xece0, 0xa08c, 0xd7a6, 0xa08d, 0xc5c0, 0xa08e, 0xa08f, 0xa090, /*0x28-0x2f*/
04629 0xebbc, 0xb0ae, 0xa091, 0xa092, 0xa093, 0xbef4, 0xb8b8, 0xd2af, /*0x30-0x37*/
04630 0xb0d6, 0xb5f9, 0xa094, 0xd8b3, 0xa095, 0xcbac, 0xa096, 0xe3dd, /*0x38-0x3f*/
04631 0xa097, 0xa098, 0xa099, 0xa09a, 0xa09b, 0xa09c, 0xa09d, 0xc6ac, /*0x40-0x47*/
04632 0xb0e6, 0xa09e, 0xa09f, 0xa0a0, 0xc5c6, 0xebb9, 0xa0a1, 0xa0a2, /*0x48-0x4f*/
04633 0xa0a3, 0xa0a4, 0xebba, 0xa0a5, 0xa0a6, 0xa0a7, 0xebbb, 0xa0a8, /*0x50-0x57*/
04634 0xa0a9, 0xd1c0, 0xa0aa, 0xc5a3, 0xa0ab, 0xeaf2, 0xa0ac, 0xc4b2, /*0x58-0x5f*/
04635 0xa0ad, 0xc4b5, 0xc0ce, 0xa0ae, 0xa0af, 0xa0b0, 0xeaf3, 0xc4c1, /*0x60-0x67*/
04636 0xa0b1, 0xcceef, 0xa0b2, 0xa0b3, 0xa0b4, 0xa0b5, 0xeaf0, 0xeaf4, /*0x68-0x6f*/
04637 0xa0b6, 0xa0b7, 0xc9fc, 0xa0b8, 0xa0b9, 0xc7a3, 0xa0ba, 0xa0bb, /*0x70-0x77*/
04638 0xa0bc, 0xccd8, 0xcfe0, 0xa0bd, 0xa0be, 0xa0bf, 0xeaf5, 0xeaf6, /*0x78-0x7f*/
04639 0xcfac, 0xc0e7, 0xa0c0, 0xa0c1, 0xeaf7, 0xa0c2, 0xa0c3, 0xa0c4, /*0x80-0x87*/
04640 0xa0c5, 0xa0c6, 0xb6bf, 0xeaf8, 0xa0c7, 0xeaf9, 0xa0c8, 0xeafa, /*0x88-0x8f*/
04641 0xa0c9, 0xa0ca, 0xeafb, 0xa0cb, 0xa0cc, 0xa0cd, 0xa0ce, 0xa0cf, /*0x90-0x97*/
04642 0xa0d0, 0xa0d1, 0xa0d2, 0xa0d3, 0xa0d4, 0xa0d5, 0xa0d6, 0xeaf1, /*0x98-0x9f*/
04643 0xa0d7, 0xa0d8, 0xa0d9, 0xa0da, 0xa0db, 0xa0dc, 0xa0dd, 0xa0de, /*0xa0-0xaf*/
04644 0xa0df, 0xa0e0, 0xa0e1, 0xa0e2, 0xc8ae, 0xe1eb, 0xa0e3, 0xb7b8, /*0xa8-0xaf*/
04645 0xe1ec, 0xa0e4, 0xa0e5, 0xa0e6, 0xe1ed, 0xa0e7, 0xd7b4, 0xe1ee, /*0xb0-0xbf*/
04646 0xe1ef, 0xd3cc, 0xa0e8, 0xa0e9, 0xa0ea, 0xa0eb, 0xa0ec, 0xa0ed, /*0xb8-0xbf*/
04647 0xa0ee, 0xe1f1, 0xbff1, 0xe1f0, 0xb5d2, 0xa0ef, 0xa0f0, 0xa0f1, /*0xc0-0xc7*/
04648 0xb1b7, 0xa0f2, 0xa0f3, 0xa0f4, 0xa0f5, 0xe1f3, 0xe1f2, 0xa0f6, /*0xc8-0xcf*/
04649 0xbafc, 0xa0f7, 0xe1f4, 0xa0f8, 0xa0f9, 0xa0fa, 0xa0fb, 0xb9b7, /*0xd0-0xdf*/
04650 0xa0fc, 0xbcd1, 0xa0fd, 0xa0fe, 0xaa40, 0xaa41, 0xc4fc, 0xaa42, /*0xd8-0xdf*/
04651 0xbadd, 0xbdc6, 0xaa43, 0xaa44, 0xaa45, 0xaa46, 0xaa47, 0xaa48, /*0xe0-0xe7*/
04652 0xe1f5, 0xe1ff, 0xaa49, 0xaa4a, 0xb6c0, 0xcfc1, 0xc8aa, 0xe1f6, /*0xe8-0xef*/
04653 0xd5f8, 0xd3fc, 0xe1f8, 0xe1fc, 0xe1f9, 0xaa4b, 0xaa4c, 0xe1fa, /*0xf0-0xf7*/
04654 0xc0ea, 0xaa4d, 0xe1fe, 0xe2a1, 0xc0c7, 0xaa4e, 0xaa4f, 0xaa50, /*0xf8-0xff*/
04655 /* 0x7300 */
04656 0xaa51, 0xe1fb, 0xaa52, 0xe1fd, 0xaa53, 0xaa54, 0xaa55, 0xaa56, /*0x00-0x07*/
04657 0xaa57, 0xaa58, 0xe2a5, 0xaa59, 0xaa5a, 0xaa5b, 0xc1d4, 0xaa5c, /*0x08-0x0f*/
04658 0xaa5d, 0xaa5e, 0xe2a5, 0xe2a3, 0xaa60, 0xe2a8, 0xb2fe, 0xe2a2, /*0x10-0x17*/
04659 0xaa61, 0xaa62, 0xaa63, 0xc3cd, 0xb2c2, 0xe2a7, 0xe2a6, 0xaa64, /*0x18-0x1f*/
04660 0xaa65, 0xe2a4, 0xe2a9, 0xaa66, 0xaa67, 0xe2ab, 0xaa68, 0xaa69, /*0x20-0x27*/
04661 0xaa6a, 0xd0c9, 0xd6ed, 0xc3a8, 0xe2ac, 0xaa6b, 0xcfd7, 0xaa6c, /*0x28-0x2f*/
04662 0xaa6d, 0xe2ae, 0xaa6e, 0xaa6f, 0xbaef, 0xaa70, 0xaa71, 0xe9e0, /*0x30-0x37*/
04663 0xe2ad, 0xe2aa, 0xaa72, 0xaa73, 0xaa74, 0xaa75, 0xbbab, 0xd4b3, /*0x38-0x3f*/
04664 0xaa76, 0xaa77, 0xaa78, 0xaa79, 0xaa7a, 0xaa7b, 0xaa7c, 0xaa7d, /*0x40-0x47*/
04665 0xaa7e, 0xaa80, 0xaa81, 0xaa82, 0xaa83, 0xe2b0, 0xaa84, 0xaa85, /*0x48-0x4f*/
04666 0xe2af, 0xaa86, 0xe9e1, 0xaa87, 0xaa88, 0xaa89, 0xaa8a, 0xe2b1, /*0x50-0x57*/
04667 0xaa8b, 0xaa8c, 0xaa8d, 0xaa8e, 0xaa8f, 0xaa90, 0xaa91, 0xaa92, /*0x58-0x5f*/
04668 0xe2b2, 0xaa93, 0xaa94, 0xaa95, 0xaa96, 0xaa97, 0xaa98, 0xaa99, /*0x60-0x67*/
04669 0xaa9a, 0xaa9b, 0xaa9c, 0xaa9d, 0xe2b3, 0xc8ca, 0xaa9e, 0xe2b4, /*0x68-0x6f*/
04670 0xaa9f, 0xaa9a, 0xab40, 0xab41, 0xab43, 0xab44, 0xab45, 0xab46, /*0x70-0x77*/
04671 0xab46, 0xab47, 0xab48, 0xab49, 0xab4a, 0xab4b, 0xe2b5, 0xab4c, /*0x78-0x7f*/
04672 0xab4d, 0xab4e, 0xab4f, 0xab50, 0xd0fe, 0xab51, 0xab52, 0xc2ca, /*0x80-0x87*/
04673 0xab53, 0xd3f1, 0xab54, 0xcdf5, 0xab55, 0xab56, 0xe7e0, 0xab57, /*0x88-0x8f*/
04674 0xab58, 0xe7e1, 0xab59, 0xab5a, 0xab5b, 0xab5c, 0xbec1, 0xab5d, /*0x90-0x97*/
04675 0xab5e, 0xab5f, 0xab60, 0xc2ea, 0xab61, 0xab62, 0xab63, 0xe7e4, /*0x98-0x9f*/
04676 0xab64, 0xab65, 0xe7e3, 0xab66, 0xab67, 0xab68, 0xab69, 0xab6a, /*0xa0-0xaf*/
04677 0xab6b, 0xcde6, 0xab6c, 0xc3b5, 0xab6d, 0xab6e, 0xe7e2, 0xbbb7, /*0xa8-0xaf*/
04678 0xcfd6, 0xab6f, 0xc1e1, 0xe7e9, 0xab70, 0xab71, 0xab72, 0xe7e8, /*0xb0-0xbf*/
04679 0xab73, 0xab74, 0xe7f4, 0xb2a3, 0xab75, 0xab76, 0xab77, 0xab78, /*0xb8-0xbf*/
04680 0xe7ea, 0xab79, 0xe7e6, 0xab7a, 0xab7b, 0xab7c, 0xab7d, 0xab7e, /*0xc0-0xc7*/
04681 0xe7ec, 0xe7eb, 0xc9ba, 0xab80, 0xab81, 0xd5e4, 0xab82, 0xe7e5, /*0xc8-0xcf*/
04682 0xb7a9, 0xe7e7, 0xab83, 0xab84, 0xab85, 0xab86, 0xab87, 0xab88, /*0xd0-0xd7*/
04683 0xab89, 0xe7ee, 0xab8a, 0xab8b, 0xab8c, 0xab8d, 0xe7f3, 0xab8e, /*0xd8-0xdf*/
04684 0xd6e9, 0xab8f, 0xab90, 0xab91, 0xab92, 0xe7ed, 0xab93, 0xe7f2, /*0xe0-0xe7*/
04685 0xab94, 0xe7f1, 0xab95, 0xab96, 0xab97, 0xb0e0, 0xab98, 0xab99, /*0xe8-0xef*/
04686 0xab9a, 0xab9b, 0xe7f5, 0xab9c, 0xab9d, 0xab9e, 0xab9f, 0aba0, /*0xf0-0xf7*/
04687 0xac40, 0xac41, 0xac42, 0xac43, 0xac44, 0xac45, 0xac46, 0xac47, /*0xf8-0xff*/
04688 /* 0x7400 */
04689 0xac48, 0xac49, 0xac4a, 0xc7f2, 0xac4b, 0xc0c5, 0xc0ed, 0xac4c, /*0x00-0x07*/
04690 0xac4d, 0xc1f0, 0xe7f0, 0xac4e, 0xac4f, 0xac50, 0xac51, 0xe7f6, /*0x08-0x0f*/
04691 0xcbbf, 0xac52, 0xac53, 0xac54, 0xac55, 0xac56, 0xac57, 0xac58, /*0x10-0x17*/
04692 0xac59, 0xac5a, 0xe8a2, 0xe8a1, 0xac5b, 0xac5c, 0xac5d, 0xac5e, /*0x18-0x1f*/
04693 0xac5f, 0xac60, 0xd7c1, 0xac61, 0xac62, 0xe7fa, 0xe7f9, 0xac63, /*0x20-0x27*/
04694 0xe7fb, 0xac64, 0xe7f7, 0xac65, 0xe7fe, 0xac66, 0xe7fd, 0xac67, /*0x28-0x2f*/
04695 0xe7fc, 0xac68, 0xac69, 0xc1d5, 0xc7d9, 0xc5fd, 0xc5c3, 0xac6a, /*0x30-0x37*/
04696 0xac6b, 0xac6c, 0xac6d, 0xc6e, 0xc7ed, 0xac6f, 0xac70, 0xac71, /*0x38-0x3f*/
04697 0xac72, 0xe8a3, 0xac73, 0xac74, 0xac75, 0xac76, 0xac77, 0xac78, /*0x40-0x47*/
04698 0xac79, 0xac7a, 0xac7b, 0xac7c, 0xac7d, 0xac7e, 0xac80, 0xac81, /*0x48-0x4f*/
04699 0xac82, 0xac83, 0xac84, 0xac85, 0xac86, 0xe8a6, 0xac87, 0xe8a5, /*0x50-0x57*/
04700 0xac88, 0xe8a7, 0xbaf7, 0xe7f8, 0xe8a4, 0xac89, 0xc8f0, 0xc9aa, /*0x58-0x5f*/
04701 0xac8a, 0xac8b, 0xac8c, 0xac8d, 0xac8e, 0xac8f, 0xac90, 0xac91, /*0x60-0x67*/
04702 0xac92, 0xac93, 0xac94, 0xac95, 0xac96, 0xe8a9, 0xac97, 0xac98, /*0x68-0x6f*/
04703 0xb9e5, 0xac99, 0xac9a, 0xac9b, 0xac9c, 0xac9d, 0xd1fe, 0xe8a8, /*0x70-0x77*/
```



```

04704 0xac9e, 0xac9f, 0xaca0, 0xad40, 0xad41, 0xad42, 0xe8aa, 0xad43, /*0x78-0x7f*/
04705 0xe8ad, 0xe8ae, 0xad44, 0xc1a7, 0xad45, 0xad46, 0xad47, 0xe8af, /*0x80-0x87*/
04706 0xad48, 0xad49, 0xad4a, 0xe8b0, 0xad4b, 0xad4c, 0xe8ac, 0xad4d, /*0x88-0x8f*/
04707 0xe8b4, 0xad4e, 0xad4f, 0xad50, 0xad51, 0xad52, 0xad53, 0xad54, /*0x90-0x97*/
04708 0xad55, 0xad56, 0xad57, 0xad58, 0xe8ab, 0xad59, 0xe8b1, 0xad5a, /*0x98-0x9f*/
04709 0xad5b, 0xad5c, 0xad5d, 0xad5e, 0xad5f, 0xad60, 0xad61, 0xe8b5, /*0xa0-0xa7*/
04710 0xe8b2, 0xe8b3, 0xad62, 0xad63, 0xad64, 0xad65, 0xad66, 0xad67, /*0xa8-0xaf*/
04711 0xad68, 0xad69, 0xad6a, 0xad6b, 0xad6c, 0xad6d, 0xad6e, 0xad6f, /*0xb0-0xb7*/
04712 0xad70, 0xad71, 0xe8b7, 0xad72, 0xad73, 0xad74, 0xad75, 0xad76, /*0xb8-0xbf*/
04713 0xad77, 0xad78, 0xad79, 0xad7a, 0xad7b, 0xad7c, 0xad7d, 0xad7e, /*0xc0-0xc7*/
04714 0xad80, 0xad81, 0xad82, 0xad83, 0xad84, 0xad85, 0xad86, 0xad87, /*0xc8-0xcf*/
04715 0xad88, 0xad89, 0xe8b6, 0xad8a, 0xad8b, 0xad8c, 0xad8d, 0xad8e, /*0xd0-0xd7*/
04716 0xad8f, 0xad90, 0xad91, 0xad92, 0xb9cf, 0xad93, 0xf0ac, 0xad94, /*0xd8-0xdf*/
04717 0xf0ad, 0xad95, 0xc6b0, 0xb0ea, 0xc8bf, 0xad96, 0xcddc, 0xad97, /*0xe0-0xe7*/
04718 0xad98, 0xad99, 0xad9a, 0xad9b, 0xad9c, 0xcecd, 0xeab1, /*0xe8-0xef*/
04719 0xad9e, 0xad9f, 0xada0, 0xae40, 0xeab2, 0xae41, 0xc6bf, 0xb4c9, /*0xf0-0xf7*/
04720 0xae42, 0xae43, 0xae44, 0xae45, 0xae46, 0xae47, 0xae48, 0xeab3, /*0xf8-0xff*/
04721 /* 0x7500 */
04722 0xae49, 0xae4a, 0xae4b, 0xae4c, 0xd5e7, 0xae4d, 0xae4e, 0xae4f, /*0x00-0x07*/
04723 0xae50, 0xae51, 0xae52, 0xae53, 0xae54, 0xddf9, 0xae55, 0xeab4, /*0x08-0x0f*/
04724 0xae56, 0xae5b, 0xae57, 0xeab6, 0xae58, 0xae59, 0xae5a, 0xae5b, /*0x10-0x17*/
04725 0xb8ca, 0xdfb0, 0xc9f5, 0xae5c, 0xccf0, 0xae5d, 0xae5e, 0xc9fa, /*0x18-0x1f*/
04726 0xae5f, 0xae60, 0xae61, 0xae62, 0xae63, 0xc9fb, 0xae64, 0xae65, /*0x20-0x27*/
04727 0xd3c3, 0xcba6, 0xae66, 0xb8a6, 0xf0ae, 0xb1c2, 0xae67, 0xe5b8, /*0x28-0x2f*/
04728 0xccef, 0xd3c9, 0xbcd7, 0xc9ea, 0xae68, 0xb5e7, 0xae69, 0xc4d0, /*0x30-0x37*/
04729 0xb5e9, 0xae6a, 0xeeae, 0xbbad, 0xae6b, 0xae6c, 0xe7de, 0xae6d, /*0x38-0x3f*/
04730 0xeeaf, 0xae6e, 0xae6f, 0xae70, 0xae71, 0xb3a9, 0xae72, 0xae73, /*0x40-0x47*/
04731 0xeeb2, 0xae74, 0xae75, 0xeeb1, 0xbde7, 0xae76, 0xeeb0, 0xceb7, /*0x48-0x4f*/
04732 0xae77, 0xae78, 0xae79, 0xae7a, 0xc5cf, 0xae7b, 0xae7c, 0xae7d, /*0x50-0x57*/
04733 0xae7e, 0xc1f4, 0xdbce, 0xeeb3, 0xd0f3, 0xae80, 0xae81, 0xae82, /*0x58-0x5f*/
04734 0xae83, 0xae84, 0xae85, 0xae86, 0xae87, 0xc2d4, 0xc6e8, 0xae88, /*0x60-0x67*/
04735 0xae89, 0xae8a, 0xb7ac, 0xae8b, 0xae8c, 0xae8d, 0xae8e, 0xae8f, /*0x68-0x6f*/
04736 0xae90, 0xae91, 0xeeb4, 0xae92, 0xb3eb, 0xae93, 0xae94, 0xae95, /*0x70-0x77*/
04737 0xbbfb, 0xeeb5, 0xae96, 0xae97, 0xae98, 0xae99, 0xae9a, 0xe7dc, /*0x78-0x7f*/
04738 0xae9b, 0xae9c, 0xae9d, 0xeeb6, 0xae9e, 0xae9f, 0xbdae, 0xaea0, /*0x80-0x87*/
04739 0xaf40, 0xaf41, 0xaf42, 0xaf43, 0xaf44, 0xaf45, 0xc8ae, 0xc8ae, /*0x88-0x8f*/
04740 0xaf46, 0xd2c9, 0xf0da, 0xaf47, 0xf0db, 0xaf48, 0xf0dc, 0xc1c6, /*0x90-0x97*/
04741 0xaf49, 0xb8ed, 0xbece, 0xaf4a, 0xaf4b, 0xf0de, 0xaf4c, 0xc5b1, /*0x98-0x9f*/
04742 0xf0dd, 0xaf4f, 0xaf4d, 0xf0e0, 0xb0cc, 0xbdae, 0xaf4e, 0xaf4f, /*0xa0-0xa7*/
04743 0xaf50, 0xaf51, 0xaf52, 0xd2df, 0xf0df, 0xaf53, 0xb4af, 0xb7e8, /*0xa8-0xaf*/
04744 0xf0e6, 0xf0e5, 0xc6a3, 0xf0e1, 0xf0e2, 0xb4c3, 0xaf54, 0xaf55, /*0xb0-0xb7*/
04745 0xf0e3, 0xd5ee, 0xaf56, 0xaf57, 0xccdb, 0xbed2, 0xbcb2, 0xaf58, /*0xb8-0xbf*/
04746 0xaf59, 0xaf5a, 0xf0e8, 0xf0e7, 0xf0e4, 0xb2a1, 0xaf5b, 0xd6a2, /*0xc0-0xc7*/
04747 0xd3b8, 0xbdb7, 0xc8ac, 0xaf5c, 0xaf5d, 0xf0ea, 0xaf5e, 0xaf5f, /*0xc8-0xcf*/
04748 0xaf60, 0xaf61, 0xd1f7, 0xaf62, 0xd6cc, 0xbadb, 0xf0e9, 0xaf63, /*0xd0-0xd7*/
04749 0xb6bb, 0xaf64, 0xaf65, 0xcdb4, 0xaf66, 0xaf67, 0xc6a6, 0xaf68, /*0xd8-0xdf*/
04750 0xaf69, 0xaf6a, 0xc1a1, 0xf0eb, 0xf0ee, 0xaf6b, 0xf0ed, 0xf0f0, /*0xe0-0xe7*/
04751 0xf0ec, 0xaf6c, 0xbbbe, 0xf0ef, 0xaf6d, 0xaf6e, 0xaf6f, 0xaf70, /*0xe8-0xef*/
04752 0xccb5, 0xf0f2, 0xaf71, 0xaf72, 0xb3d5, 0xaf73, 0xaf74, 0xaf75, /*0xf0-0xf7*/
04753 0xaf76, 0xb1d4, 0xaf77, 0xaf78, 0xf0f3, 0xaf79, 0xaf7a, 0xf0f4, /*0xf8-0xff*/
04754 /* 0x7600 */
04755 0xf0f6, 0xb4e1, 0xaf7b, 0xf0f1, 0xaf7c, 0xf0f7, 0xaf7d, 0xaf7e, /*0x00-0x07*/
04756 0xaf80, 0xaf81, 0xf0fa, 0xaf82, 0xf0f8, 0xaf83, 0xaf84, 0xaf85, /*0x08-0x0f*/
04757 0xf0f5, 0xaf86, 0xaf87, 0xaf88, 0xf0fd, 0xaf89, 0xf0fa, 0xf0f9, /*0x10-0x17*/
04758 0xf0fc, 0xf0fe, 0xaf8b, 0xf1a1, 0xaf8c, 0xaf8d, 0xaf8e, 0xcce1, /*0x18-0x1f*/
04759 0xf1a4, 0xaf8f, 0xf1a3, 0xaf90, 0xc1f6, 0xf0fb, 0xcadd, 0xaf91, /*0x20-0x27*/
04760 0xaf92, 0xb4f1, 0xb1f1, 0xccb1, 0xf1a6, 0xaf94, 0xaf95, 0xaf95, /*0x28-0x2f*/
04761 0xf1a7, 0xaf96, 0xaf97, 0xf1ac, 0xd5ce, 0xf1a9, 0xaf98, 0xaf99, /*0x30-0x37*/
04762 0xc8b3, 0xaf9a, 0xaf9b, 0xaf9c, 0xf1a2, 0xaf9d, 0xf1ab, 0xf1a8, /*0x38-0x3f*/
04763 0xf1a5, 0xaf9e, 0xaf9f, 0xf1aa, 0xf0a0, 0xb041, 0xb042, 0xb042, /*0x40-0x47*/
04764 0xb043, 0xb044, 0xb045, 0xb046, 0xb0a9, 0xf1ad, 0xb047, 0xb048, /*0x48-0x4f*/
04765 0xb049, 0xb04a, 0xb04b, 0xb04c, 0xf1af, 0xb04d, 0xf1b1, 0xb04e, /*0x50-0x57*/
04766 0xb04f, 0xb050, 0xb051, 0xb052, 0xf1b0, 0xb053, 0xf1ae, 0xb054, /*0x58-0x5f*/
04767 0xb055, 0xb056, 0xb057, 0xd1a2, 0xb058, 0xb059, 0xb05a, 0xb05b, /*0x60-0x67*/
04768 0xb05c, 0xb05d, 0xb05e, 0xf1b2, 0xb05f, 0xb060, 0xf1b3, 0xf1b3, /*0x68-0x6f*/
04769 0xb062, 0xb063, 0xb064, 0xb065, 0xb066, 0xb067, 0xb068, 0xb069, /*0x70-0x77*/
04770 0xb06e, 0xb06a, 0xb06b, 0xb06c, 0xb06d, 0xb06e, 0xb06d, 0xb06d, /*0x78-0x7f*/
04771 0xb06e, 0xb06f, 0xd4ed, 0xb070, 0xb5c4, 0xb071, 0xbdd4, 0xbbc4, /*0x80-0x87*/
04772 0xf0a7, 0xb072, 0xb073, 0xb8de, 0xb074, 0xb075, 0xf0a8, 0xb076, /*0x88-0x8f*/
04773 0xb077, 0xb0a8, 0xb078, 0xf0a9, 0xb079, 0xb07a, 0xcdee, 0xb07b, /*0x90-0x97*/
04774 0xb07c, 0xf0aa, 0xb07d, 0xb07e, 0xb080, 0xb081, 0xb082, 0xb083, /*0x98-0x9f*/
04775 0xb084, 0xb085, 0xb086, 0xb087, 0xf0ab, 0xb088, 0xb089, 0xb08a, /*0xa0-0xa7*/
04776 0xb08b, 0xb08c, 0xb08d, 0xb08e, 0xb08f, 0xb090, 0xc6a4, 0xb091, /*0xa8-0xaf*/
04777 0xb092, 0xd6e5, 0xf1e4, 0xb093, 0xf1e5, 0xb094, 0xb095, 0xb096, /*0xb0-0xb7*/
04778 0xb097, 0xb098, 0xb099, 0xb09a, 0xb09b, 0xb09c, 0xb09d, 0xc3f3, /*0xb8-0xbf*/
04779 0xb09e, 0xb09f, 0xd3db, 0xb0a0, 0xb140, 0xd6d1, 0xc5e8, 0xb141, /*0xc0-0xc7*/
04780 0xd3af, 0xb142, 0xd2e6, 0xb143, 0xeecl, 0xb0bb, 0xd5b5, 0xd5b5, /*0xc8-0xcf*/
04781 0xd1ce, 0xb0ce, 0xb0ad, 0xb145, 0xb0ff, 0xb146, 0xb8c7, 0xb5c1, /*0xd0-0xd7*/
04782 0xc5cc, 0xb147, 0xb148, 0xc5aa, 0xb149, 0xb14a, 0xb14b, 0xc3cb, /*0xd8-0xdf*/
04783 0xb14c, 0xb14d, 0xb14e, 0xb14f, 0xb150, 0xeecl, 0xb151, 0xb152, /*0xe0-0xe7*/
04784 0xb153, 0xb154, 0xb155, 0xb156, 0xb157, 0xb158, 0xc4b7, 0xb6a2, /*0xe8-0xef*/
04785 0xb159, 0xedec, 0xc3a4, 0xb15a, 0xd6b1, 0xb15b, 0xb15c, 0xb15d, /*0xf0-0xf7*/
04786 0xcfe0, 0xedef, 0xb15e, 0xb15f, 0xc5ce, 0xb160, 0xb6dc, 0xb161, /*0xf8-0xff*/
04787 /* 0x7700 */
04788 0xb162, 0xcaa1, 0xb163, 0xb164, 0xeded, 0xb165, 0xb166, 0xedf0, /*0x00-0x07*/
04789 0xedf1, 0xc3bc, 0xb167, 0xbfb4, 0xb168, 0xedee, 0xb169, 0xb16a, /*0x08-0x0f*/
04790 0xb16b, 0xb16c, 0xb16d, 0xb16e, 0xb16f, 0xb170, 0xb171, 0xb172, /*0x10-0x17*/

```

```
04791 0xb173, 0xedf4, 0xedf2, 0xb174, 0xb175, 0xb176, 0xb177, 0xd5e6, /*0x18-0x1f*/
04792 0xc3df, 0xb178, 0xedf3, 0xb179, 0xb17a, 0xb17b, 0xedf6, 0xb17c, /*0x20-0x27*/
04793 0xd5a3, 0xd1a3, 0xb17d, 0xb17e, 0xb180, 0xedf5, 0xb181, 0xc3d0, /*0x28-0x2f*/
04794 0xb182, 0xb183, 0xb184, 0xb185, 0xb186, 0xedf7, 0xbff4, 0xbeec, /*0x30-0x37*/
04795 0xedf8, 0xb187, 0xccf7, 0xb188, 0xd1db, 0xb189, 0xb18a, 0xb18b, /*0x38-0x3f*/
04796 0xd7c5, 0xd5f6, 0xb18c, 0xedfc, 0xb18d, 0xb18e, 0xb18f, 0xedfb, /*0x40-0x47*/
04797 0xb190, 0xb191, 0xb192, 0xb193, 0xb194, 0xb195, 0xb196, 0xb197, /*0x48-0x4f*/
04798 0xedf9, 0xedfa, 0xb198, 0xb199, 0xb19a, 0xb19b, 0xb19c, 0xb19d, /*0x50-0x57*/
04799 0xb19e, 0xb19f, 0xedfd, 0xbea6, 0xb1a0, 0xb240, 0xb241, 0xb242, /*0x58-0x5f*/
04800 0xb243, 0xcbafe, 0xeea1, 0xb6bd, 0xb244, 0xeea2, 0xc4c0, 0xb245, /*0x60-0x67*/
04801 0xedfe, 0xb246, 0xb247, 0xbddc, 0xb2c7, 0xb248, 0xb249, 0xb24a, /*0x68-0x6f*/
04802 0xb24b, 0xb24c, 0xb24d, 0xb24e, 0xb24f, 0xb250, 0xb251, 0xb252, /*0x70-0x77*/
04803 0xb253, 0xb6c3, 0xb254, 0xb255, 0xb256, 0xeea5, 0xd8ba, 0xeea3, /*0x78-0x7f*/
04804 0xeea6, 0xb257, 0xb258, 0xb259, 0xc3e9, 0xb3f2, 0xb25a, 0xb25b, /*0x80-0x87*/
04805 0xb25c, 0xb25d, 0xb25e, 0xb25f, 0xeea7, 0xeea4, 0xcfb9, 0xb260, /*0x88-0x8f*/
04806 0xb261, 0xeea8, 0xc2f7, 0xb262, 0xb263, 0xb264, 0xb265, 0xb266, /*0x90-0x97*/
04807 0xb267, 0xb268, 0xb269, 0xb26a, 0xb26b, 0xb26c, 0xb26d, 0xeea9, /*0x98-0x9f*/
04808 0xeeaa, 0xb26e, 0xdeb, 0xb26f, 0xb270, 0xc6b3, 0xb271, 0xc7c6, /*0xa0-0xaf*/
04809 0xb272, 0xd6f5, 0xb5c9, 0xb273, 0xcbb2, 0xb274, 0xb275, 0xb276, /*0xa8-0xaf*/
04810 0xeeab, 0xb277, 0xb278, 0xcdab, 0xb279, 0xeeac, 0xb27a, 0xb27b, /*0xb0-0xbf*/
04811 0xb27c, 0xb27d, 0xb27e, 0xd5b0, 0xb280, 0xeead, 0xb281, 0xf6c4, /*0xb8-0xbf*/
04812 0xb282, 0xb283, 0xb284, 0xb285, 0xb286, 0xb287, 0xb288, 0xb289, /*0xc0-0xcf*/
04813 0xb28a, 0xb28b, 0xb28c, 0xb28d, 0xb28e, 0xbdc7, 0xb28f, 0xb290, /*0xc8-0xcf*/
04814 0xb291, 0xb292, 0xb293, 0xb294, 0xb295, 0xb296, 0xb297, 0xb4a3, /*0xd0-0xdf*/
04815 0xb298, 0xb299, 0xb29a, 0xc3ac, 0xf1e6, 0xb29b, 0xb29c, 0xb29d, /*0xd8-0xdf*/
04816 0xb29e, 0xb29f, 0xcab8, 0xd2d3, 0xb2a0, 0xd6aa, 0xb340, 0xeff2, /*0xe0-0xef*/
04817 0xb341, 0xb2ed8, 0xb342, 0xbdc3, 0xeff3, 0xb6cc, 0xb0ab, 0xb343, /*0xe8-0xef*/
04818 0xb344, 0xb345, 0xb346, 0xcaaf, 0xb347, 0xb348, 0xedb6, 0xb349, /*0xf0-0xf7*/
04819 0xedb7, 0xb34a, 0xb34b, 0xb34c, 0xb34d, 0xcfe9, 0xb7af, 0xbff3, /*0xf8-0xff*/
04820 /* 0x7800 */
04821 0xedb8, 0xc2eb, 0xc9b0, 0xb34e, 0xb34f, 0xb350, 0xb351, 0xb352, /*0x00-0x07*/
04822 0xb353, 0xedb9, 0xb354, 0xb355, 0xc6f6, 0xbfb3, 0xb356, 0xb357, /*0x08-0x0f*/
04823 0xb358, 0xedbc, 0xc5f8, 0xb359, 0xd1d0, 0xb35a, 0xd7a9, 0xedba, /*0x10-0x17*/
04824 0xedbb, 0xb35b, 0xd1e2, 0xb35c, 0xedbf, 0xedc0, 0xb35d, 0xedc4, /*0x18-0x1f*/
04825 0xb35e, 0xb35f, 0xb360, 0xedc8, 0xb361, 0xedc6, 0xedce, 0xd5e8, /*0x20-0x27*/
04826 0xb362, 0xedc9, 0xb363, 0xb364, 0xedc7, 0xedbe, 0xb365, 0xb366, /*0x28-0x2f*/
04827 0xc5e9, 0xb367, 0xb368, 0xb369, 0xc6c6, 0xb36a, 0xb36b, 0xc9e9, /*0x30-0x37*/
04828 0xd4d2, 0xedc1, 0xedc2, 0xedc3, 0xedc5, 0xb36c, 0xc0f9, 0xb36d, /*0x38-0x3f*/
04829 0xb4a1, 0xb36e, 0xb36f, 0xb370, 0xb371, 0xb9e8, 0xb372, 0xedd0, /*0x40-0x47*/
04830 0xb373, 0xb374, 0xb375, 0xb376, 0xedd1, 0xb377, 0xedca, 0xb378, /*0x48-0x4f*/
04831 0xedcf, 0xb379, 0xcfe8, 0xb37a, 0xb37b, 0xcbb6, 0xedcc, 0xedcd, /*0x50-0x57*/
04832 0xb37c, 0xb37d, 0xb37e, 0xb380, 0xb381, 0xcff5, 0xb382, 0xb383, /*0x58-0x5f*/
04833 0xb384, 0xb385, 0xb386, 0xb387, 0xb388, 0xb389, 0xb38a, 0xb38b, /*0x60-0x67*/
04834 0xb38c, 0xb38d, 0xedd2, 0xc1f2, 0xd3b2, 0xedcb, 0xc8b7, 0xb38e, /*0x68-0x6f*/
04835 0xb38f, 0xb390, 0xb391, 0xb392, 0xb393, 0xb394, 0xb395, 0xbcef, /*0x70-0x77*/
04836 0xb396, 0xb397, 0xb398, 0xb399, 0xc5f0, 0xb39a, 0xb39b, 0xb39c, /*0x78-0x7f*/
04837 0xb39d, 0xb39e, 0xb39f, 0xb3a0, 0xb440, 0xb441, 0xb442, 0xedd6, /*0x80-0x87*/
04838 0xb443, 0xb444, 0xb445, 0xc2b5, 0xb0ad, 0xcbe9, 0xb446, /*0x88-0x8f*/
04839 0xb447, 0xb1ae, 0xb448, 0xedd4, 0xb449, 0xb44a, 0xb44b, 0xcdeb, /*0x90-0x97*/
04840 0xb5e2, 0xb44c, 0xedd5, 0xedd3, 0xedd7, 0xb44d, 0xb44e, 0xb5fa, /*0x98-0x9f*/
04841 0xb44f, 0xedd8, 0xb450, 0xedd9, 0xb451, 0xeddc, 0xb452, 0xb1cc, /*0xa0-0xaf*/
04842 0xb453, 0xb454, 0xb455, 0xb456, 0xb457, 0xb458, 0xb459, 0xb45a, /*0xa8-0xaf*/
04843 0xc5f6, 0xbcee, 0xedda, 0xcabc, 0xb2ea, 0xb45b, 0xb45c, 0xb45d, /*0xb0-0xbf*/
04844 0xb45e, 0xeddb, 0xb45f, 0xb460, 0xb461, 0xb462, 0xc4eb, 0xb463, /*0xb8-0xbf*/
04845 0xb464, 0xb4c5, 0xb465, 0xb466, 0xb467, 0xb0f5, 0xb468, 0xb469, /*0xc0-0xcf*/
04846 0xb46a, 0xeddf, 0xc0da, 0xb4e8, 0xb46b, 0xb46c, 0xb46d, 0xb46e, /*0xc8-0xcf*/
04847 0xc5cd, 0xb46f, 0xb470, 0xb471, 0xeddd, 0xbfc4, 0xb472, 0xb473, /*0xd0-0xdf*/
04848 0xb474, 0xedde, 0xb475, 0xb476, 0xb477, 0xb478, 0xb479, 0xb47a, /*0xd8-0xdf*/
04849 0xb47b, 0xb47c, 0xb47d, 0xb47e, 0xb480, 0xb481, 0xb482, 0xb483, /*0xe0-0xef*/
04850 0xc4a5, 0xb484, 0xb485, 0xb486, 0xede0, 0xb487, 0xb488, 0xb489, /*0xe8-0xef*/
04851 0xb48a, 0xb48b, 0xede1, 0xb48c, 0xede3, 0xb48d, 0xb48e, 0xc1d7, /*0xf0-0xf7*/
04852 0xb48f, 0xb490, 0xbbc7, 0xb491, 0xb492, 0xb493, 0xb494, 0xb495, /*0xf8-0xff*/
04853 /* 0x7900 */
04854 0xb496, 0xbdb8, 0xb497, 0xb498, 0xb499, 0xede2, 0xb49a, 0xb49b, /*0x00-0x07*/
04855 0xb49c, 0xb49d, 0xb49e, 0xb49f, 0xb4a0, 0xb540, 0xb541, 0xb542, /*0x08-0x0f*/
04856 0xb543, 0xb544, 0xb545, 0xede4, 0xb546, 0xb547, 0xb548, 0xb549, /*0x10-0x17*/
04857 0xb54a, 0xb54b, 0xb54c, 0xb54d, 0xb54e, 0xb54f, 0xede6, 0xb550, /*0x18-0x1f*/
04858 0xb551, 0xb552, 0xb553, 0xb554, 0xede5, 0xb555, 0xb556, 0xb557, /*0x20-0x27*/
04859 0xb558, 0xb559, 0xb55a, 0xb55b, 0xb55c, 0xb55d, 0xb55e, 0xb55f, /*0x28-0x2f*/
04860 0xb560, 0xb561, 0xb562, 0xb563, 0xede7, 0xb564, 0xb565, 0xb566, /*0x30-0x37*/
04861 0xb567, 0xb568, 0xcabe, 0xeece, 0xc0f1, 0xb569, 0xc9e7, 0xb56a, /*0x38-0x3f*/
04862 0xeeceb, 0xc6ee, 0xb56b, 0xb56c, 0xb56d, 0xb56e, 0xeecec, 0xb56f, /*0x40-0x47*/
04863 0xc6fed, 0xeced, 0xb570, 0xb571, 0xb572, 0xb573, 0xb574, 0xb575, /*0x48-0x4f*/
04864 0xb576, 0xb577, 0xb578, 0xecf0, 0xb579, 0xb57a, 0xd7e6, 0xecf3, /*0x50-0x57*/
04865 0xb57b, 0xb57c, 0xecf1, 0xeece, 0xecef, 0xd7a3, 0xc9f1, 0xcbee, /*0x58-0x5f*/
04866 0xecf4, 0xb57d, 0xecf2, 0xb57e, 0xb580, 0xcfe9, 0xb581, 0xecf6, /*0x60-0x67*/
04867 0xc6b1, 0xb582, 0xb583, 0xb584, 0xb585, 0xb586, 0xb587, 0xecf5, /*0x68-0x6f*/
04868 0xb588, 0xb589, 0xb58a, 0xb58b, 0xb58c, 0xb58d, 0xb58e, 0xb58f, /*0x70-0x77*/
04869 0xb590, 0xb58e, 0xecf7, 0xb58f, 0xb590, 0xb591, 0xb592, 0xb593, /*0x78-0x7f*/
04870 0xd9f7, 0xbdfb, 0xb594, 0xb595, 0xc2bb, 0xecf8, 0xb596, 0xb597, /*0x80-0x8f*/
04871 0xb598, 0xb599, 0xecf9, 0xb59a, 0xb59b, 0xb59c, 0xb59d, 0xb8a3, /*0x88-0x8f*/
04872 0xb59e, 0xb59f, 0xb5a0, 0xb640, 0xb641, 0xb642, 0xb643, 0xb644, /*0x90-0x97*/
04873 0xb645, 0xb646, 0xecfa, 0xb647, 0xb648, 0xb649, 0xb64a, 0xb64b, /*0x98-0x9f*/
04874 0xb64c, 0xb64d, 0xb64e, 0xb64f, 0xb650, 0xb651, 0xb652, 0xecfb, /*0xa0-0xaf*/
04875 0xb653, 0xb654, 0xb655, 0xb656, 0xb657, 0xb658, 0xb659, 0xb65a, /*0xa8-0xaf*/
04876 0xb65b, 0xb65c, 0xb65d, 0xecfc, 0xb65e, 0xb65f, 0xb660, 0xb661, /*0xb0-0xbf*/
04877 0xb662, 0xd3ed, 0xd8ae, 0xc0eb, 0xb663, 0xc7dd, 0xbacc, 0xb664, /*0xb8-0xbf*/
```

```
04878 0xd0e3, 0xcbbd, 0xb665, 0xcdba, 0xb666, 0xb667, 0xb8d1, 0xb668, /*xc0-0xc7*/
04879 0xb669, 0xb1fc, 0xb66a, 0xc7ef, 0xb66b, 0xd6d6, 0xb66c, 0xb66d, /*xc8-0xcf*/
04880 0xb66e, 0xbfc6, 0xc3eb, 0xb66f, 0xb670, 0xef5, 0xb671, 0xb672, /*xd0-0xd7*/
04881 0xc3d8, 0xb673, 0xb674, 0xb675, 0xb676, 0xb677, 0xb678, 0xd7e2, /*xd8-0xdf*/
04882 0xb679, 0xb67a, 0xb67b, 0xefff7, 0xb3d3, 0xb67c, 0xc7d8, 0xd1ed, /*xe0-0xe7*/
04883 0xb67d, 0xd6c8, 0xb67e, 0xefff8, 0xb680, 0xefff6, 0xb681, 0xbbfdf, /*xe8-0xef*/
04884 0xb3c6, 0xb682, 0xb683, 0xb684, 0xb685, 0xb686, 0xb687, 0xb688, /*xf0-0xf7*/
04885 0xbdd5, 0xb689, 0xb68a, 0xd2c6, 0xb68b, 0xbbe0, 0xb68c, 0xb68d, /*xf8-0xff*/
04886 /* 0x7a00 */
04887 0xcfa1, 0xb68e, 0xeffc, 0xeffb, 0xb68f, 0xb690, 0xefff9, 0xb691, /*x00-0x07*/
04888 0xb692, 0xb693, 0xb694, 0xb3cc, 0xb695, 0xc9d4, 0xcbb0, 0xb696, /*x08-0x0f*/
04889 0xb697, 0xb698, 0xb699, 0xb69a, 0xeffe, 0xb69b, 0xb69c, 0xb0de, /*x10-0x17*/
04890 0xb69d, 0xb69e, 0xd6c9, 0xb69f, 0xb6a0, 0xb740, 0xeffd, 0xb741, /*x18-0x1f*/
04891 0xb3ed, 0xb742, 0xb743, 0xf6d5, 0xb744, 0xb745, 0xb746, 0xb747, /*x20-0x27*/
04892 0xb748, 0xb749, 0xb74a, 0xb74b, 0xb74c, 0xb74d, 0xb74e, 0xb74f, /*x28-0x2f*/
04893 0xb750, 0xb751, 0xb752, 0xccec8, 0xb753, 0xb754, 0xb755, 0xf0a2, /*x30-0x37*/
04894 0xb756, 0xf0a1, 0xb757, 0xb5be, 0xbcdca, 0xbbfbc, 0xb758, 0xb8e5, /*x38-0x3f*/
04895 0xb759, 0xb75a, 0xb75b, 0xb75c, 0xb75d, 0xb75e, 0xc4c2, 0xb75f, /*x40-0x47*/
04896 0xb760, 0xb761, 0xb762, 0xb763, 0xb764, 0xb765, 0xb766, 0xb767, /*x48-0x4f*/
04897 0xb768, 0xf0a3, 0xb769, 0xb76a, 0xb76b, 0xb76c, 0xb76d, 0xcbeeb, /*x50-0x57*/
04898 0xb76e, 0xb76f, 0xb770, 0xb771, 0xb772, 0xb773, 0xb774, 0xb775, /*x58-0x5f*/
04899 0xb776, 0xb777, 0xb778, 0xb779, 0xb77a, 0xb77b, 0xb77c, 0xb77d, /*x60-0x67*/
04900 0xb77e, 0xb780, 0xb781, 0xb782, 0xb783, 0xb784, 0xb785, 0xb786, /*x68-0x6f*/
04901 0xf0a6, 0xb787, 0xb788, 0xb789, 0xd1a8, 0xb78a, 0xbefb, 0xc7ee, /*x70-0x77*/
04902 0xf1b6, 0xf1b7, 0xbfd5, 0xb78b, 0xb78c, 0xb78d, 0xb78e, 0xb4a9, /*x78-0x7f*/
04903 0xf1b8, 0xcdbb, 0xb78f, 0xc7d4, 0xd5ad, 0xb790, 0xf1b9, 0xb791, /*x80-0x87*/
04904 0xf1ba, 0xb792, 0xb793, 0xb794, 0xb795, 0xc7cf, 0xb796, 0xb797, /*x88-0x8f*/
04905 0xb798, 0xd2a4, 0xd6cf, 0xb799, 0xb79a, 0xf1bb, 0xbdd1, 0xb4b0, /*x90-0x97*/
04906 0xbefb, 0xb79b, 0xb79c, 0xb79d, 0xb4dc, 0xcdd1, 0xb79e, 0xbfdf, /*x98-0x9f*/
04907 0xf1bd, 0xb79f, 0xb7a0, 0xb840, 0xb841, 0xbffa, 0xf1bc, 0xb842, /*xa0-0xaf*/
04908 0xf1bf, 0xb843, 0xb844, 0xb845, 0xf1be, 0xf1c0, 0xb846, 0xb847, /*xa8-0xaf*/
04909 0xb848, 0xb849, 0xb84a, 0xf1c1, 0xb84b, 0xb84c, 0xb84d, 0xb84e, /*xb0-0xb7*/
04910 0xb84f, 0xb850, 0xb851, 0xb852, 0xb853, 0xb854, 0xb855, 0xc1fe, /*xb8-0xbf*/
04911 0xb856, 0xb857, 0xb858, 0xb859, 0xb85a, 0xb85b, 0xb85c, 0xb85d, /*xc0-0xc7*/
04912 0xb85e, 0xb85f, 0xb860, 0xc1a2, 0xb861, 0xb862, 0xb863, 0xb864, /*xc8-0xcf*/
04913 0xb865, 0xb866, 0xb867, 0xb868, 0xb869, 0xb86a, 0xcafa, 0xb86b, /*xd0-0xd7*/
04914 0xb86c, 0xd5be, 0xb86d, 0xb86e, 0xb86f, 0xb870, 0xb871, 0xb872, /*xd8-0xdf*/
04915 0xd5c2, 0xb871, 0xb872, 0xb873, 0xb874, 0xb875, 0xb876, 0xb877, /*xe0-0xe7*/
04916 0xb878, 0xb879, 0xb87a, 0xb87b, 0xb87c, 0xb87d, 0xb87e, 0xb87f, /*xe8-0xef*/
04917 0xb878, 0xb879, 0xb87a, 0xb87b, 0xb87c, 0xb87d, 0xb87e, 0xb87f, /*xf0-0xf7*/
04918 0xb884, 0xd6f1, 0xf3c3, 0xb885, 0xb886, 0xf3c4, 0xb887, 0xb88d, /*xf8-0xff*/
04919 /* 0x7b00 */
04920 0xb888, 0xb889, 0xb88a, 0xf3c6, 0xf3c7, 0xb88b, 0xb0ca, 0xb88c, /*x00-0x07*/
04921 0xf3c5, 0xb88d, 0xf3c9, 0xcbf1, 0xb88e, 0xb88f, 0xb890, 0xf3cb, /*x08-0x0f*/
04922 0xb891, 0xd0a6, 0xb892, 0xb893, 0xb1ca, 0xf3c8, 0xb894, 0xb895, /*x10-0x17*/
04923 0xb896, 0xf3cf, 0xb897, 0xb5d1, 0xb898, 0xb899, 0xf3d7, 0xb89a, /*x18-0x1f*/
04924 0xf3d2, 0xb89b, 0xb89c, 0xb89d, 0xf3d4, 0xf3d3, 0xb7fb, 0xb89e, /*x20-0x27*/
04925 0xb1bf, 0xb89f, 0xf3ce, 0xf3ca, 0xb5da, 0xb8a0, 0xf3d0, 0xb940, /*x28-0x2f*/
04926 0xb941, 0xf3d1, 0xb942, 0xf3d5, 0xb943, 0xb944, 0xb945, 0xb946, /*x30-0x37*/
04927 0xf3cd, 0xb947, 0xbce3, 0xb948, 0xc1fd, 0xb949, 0xf3d6, 0xb94a, /*x38-0x3f*/
04928 0xb94b, 0xb94c, 0xb94d, 0xb94e, 0xb94f, 0xf3da, 0xb950, 0xf3cc, /*x40-0x47*/
04929 0xb951, 0xb5c8, 0xb952, 0xbdee, 0xf3dc, 0xb953, 0xb954, 0xb7a4, /*x48-0x4f*/
04930 0xbbf0, 0xd6fe, 0xcdb2, 0xb955, 0xb4f0, 0xb956, 0xb2df, 0xb957, /*x50-0x57*/
04931 0xf3d8, 0xb958, 0xf3d9, 0xc9b8, 0xb959, 0xf3dd, 0xb95a, 0xb95b, /*x58-0x5f*/
04932 0xf3de, 0xb95c, 0xf3e1, 0xb95d, 0xb95e, 0xb95f, 0xb960, 0xb961, /*x60-0x67*/
04933 0xb962, 0xb963, 0xb964, 0xb965, 0xb966, 0xb967, 0xf3df, 0xb968, /*x68-0x6f*/
04934 0xb969, 0xf3e3, 0xf3e2, 0xb96a, 0xb96b, 0xf3db, 0xb96c, 0xbfea, /*x70-0x77*/
04935 0xb96d, 0xb3ef, 0xb96e, 0xf3e0, 0xb96f, 0xb970, 0xc7a9, 0xb971, /*x78-0x7f*/
04936 0xbcf2, 0xb972, 0xb973, 0xb974, 0xb975, 0xf3eb, 0xb976, 0xb977, /*x80-0x87*/
04937 0xb978, 0xb979, 0xb97a, 0xb97b, 0xb97c, 0xb9bf, 0xb97d, 0xb97e, /*x88-0x8f*/
04938 0xf3e4, 0xb980, 0xb981, 0xb982, 0xb2ad, 0xbbfef, 0xb983, 0xcbe3, /*x90-0x97*/
04939 0xb984, 0xb985, 0xb986, 0xb987, 0xf3ed, 0xf3e9, 0xb988, 0xb989, /*x98-0x9f*/
04940 0xb98a, 0xb9dc, 0xf3ee, 0xb98b, 0xb98c, 0xb98d, 0xf3e5, 0xf3e6, /*xa0-0xaf*/
04941 0xf3ea, 0xc2e1, 0xf3ec, 0xf3ef, 0xf3e8, 0xbcfdf, 0xb98e, 0xb98f, /*xa8-0xaf*/
04942 0xb990, 0xcfe4, 0xb991, 0xb992, 0xf3f0, 0xb993, 0xb994, 0xb995, /*xb0-0xb7*/
04943 0xf3e7, 0xb996, 0xb997, 0xb998, 0xb999, 0xb99a, 0xb99b, 0xb99c, /*xb8-0xbf*/
04944 0xb99d, 0xf3f2, 0xb99e, 0xb99f, 0xb9a0, 0xba40, 0xd7ad, 0xc6aa, /*xc0-0xc7*/
04945 0xba41, 0xba42, 0xba43, 0xba44, 0xf3f3, 0xba45, 0xba46, 0xba47, /*xc8-0xcf*/
04946 0xba48, 0xf3f1, 0xba49, 0xc2a8, 0xba4a, 0xba4b, 0xba4c, 0xba4d, /*xd0-0xd7*/
04947 0xba4e, 0xb8dd, 0xf3f5, 0xba4f, 0xba50, 0xf3f4, 0xba51, 0xba52, /*xd8-0xdf*/
04948 0xba53, 0xb4db, 0xba54, 0xba55, 0xba56, 0xf3f7, 0xba57, 0xba58, /*xe0-0xe7*/
04949 0xba58, 0xba59, 0xf3f8, 0xba5a, 0xba5b, 0xba5c, 0xc0ba, 0xba5d, /*xe8-0xef*/
04950 0xba5e, 0xc0e9, 0xba5f, 0xba60, 0xba61, 0xba62, 0xba63, 0xc5f1, /*xf0-0xf7*/
04951 0xba64, 0xba65, 0xba66, 0xba67, 0xf3fb, 0xba68, 0xf3fa, 0xba69, /*xf8-0xff*/
04952 /* 0x7c00 */
04953 0xba6a, 0xba6b, 0xba6c, 0xba6d, 0xba6e, 0xba6f, 0xba70, 0xb4d8, /*x00-0x07*/
04954 0xba71, 0xba72, 0xba73, 0xf3f9, 0xba74, 0xba75, 0xf3fc, 0xf3fe, /*x08-0x0f*/
04955 0xba76, 0xba77, 0xba78, 0xba79, 0xba7a, 0xba7b, 0xf3fd, 0xba7c, /*x10-0x17*/
04956 0xba7d, 0xba7e, 0xba80, 0xba81, 0xba82, 0xba83, 0xba84, 0xf4a1, /*x18-0x1f*/
04957 0xba85, 0xba86, 0xba87, 0xba88, 0xba89, 0xba8a, 0xf4a3, 0xbbc9, /*x20-0x27*/
04958 0xba8b, 0xba8c, 0xf4a2, 0xba8d, 0xba8e, 0xba8f, 0xba90, 0xba91, /*x28-0x2f*/
04959 0xba92, 0xba93, 0xba94, 0xba95, 0xba96, 0xba97, 0xba98, 0xba99, /*x30-0x37*/
04960 0xf4a4, 0xba9a, 0xba9b, 0xba9c, 0xba9d, 0xba9e, 0xba9f, 0xb2be, /*x38-0x3f*/
04961 0xf4a6, 0xf4a5, 0xbaa0, 0xbb40, 0xbb41, 0xbb42, 0xbb43, 0xbb44, /*x40-0x47*/
04962 0xbb45, 0xbb46, 0xbb47, 0xbb48, 0xbb49, 0xbcae, 0xbb4a, 0xbb4b, /*x48-0x4f*/
04963 0xbb4c, 0xbb4d, 0xbb4e, 0xbb4f, 0xbb50, 0xbb51, 0xbb52, 0xbb53, /*x50-0x57*/
04964 0xbb54, 0xbb55, 0xbb56, 0xbb57, 0xbb58, 0xbb59, 0xbb5a, 0xbb5b, /*x58-0x5f*/
```



```
04965 0xbb5c, 0xbb5d, 0xbb5e, 0xbb5f, 0xbb60, 0xbb61, 0xbb62, 0xbb63, /*x60-0x67*/
04966 0xbb64, 0xbb65, 0xbb66, 0xbb67, 0xbb68, 0xbb69, 0xbb6a, 0xbb6b, /*x68-0x6f*/
04967 0xbb6c, 0xbb6d, 0xbb6e, 0xbb6f, 0xbb70, 0xbb71, 0xbb72, 0xbb73, /*x70-0x77*/
04968 0xbb74, 0xbb75, 0xbb76, 0xbb77, 0xbb78, 0xbb79, 0xbb7a, 0xbb7b, /*x78-0x7f*/
04969 0xbb7c, 0xbb7d, 0xbb7e, 0xbb7f, 0xbb80, 0xbb81, 0xbb82, 0xbb83, /*x80-0x87*/
04970 0xbb84, 0xbb85, 0xbb86, 0xbb87, 0xbb88, 0xbb89, 0xbb8a, 0xbb8b, /*x88-0x8f*/
04971 0xbb8c, 0xbb8d, 0xbb8e, 0xbb8f, 0xbb90, 0xbb91, 0xbb92, 0xbb93, /*x90-0x97*/
04972 0xbb94, 0xbb95, 0xbb96, 0xbb97, 0xbb98, 0xbb99, 0xbb9a, 0xbb9b, /*x98-0x9f*/
04973 0xbb9c, 0xbb9d, 0xbb9e, 0xbb9f, 0xba00, 0xba01, 0xba02, 0xba03, /*xa0-0xa7*/
04974 0xba04, 0xba05, 0xba06, 0xba07, 0xba08, 0xba09, 0xba0a, 0xba0b, /*xa8-0xaf*/
04975 0xba0c, 0xba0d, 0xba0e, 0xba0f, 0xba10, 0xba11, 0xba12, 0xba13, /*xb0-0xb7*/
04976 0xba14, 0xba15, 0xba16, 0xba17, 0xba18, 0xba19, 0xba1a, 0xba1b, /*xb8-0xbf*/
04977 0xba1c, 0xba1d, 0xba1e, 0xba1f, 0xba20, 0xba21, 0xba22, 0xba23, /*xc0-0xcf*/
04978 0xba24, 0xba25, 0xba26, 0xba27, 0xba28, 0xba29, 0xba2a, 0xba2b, /*xc8-0xcf*/
04979 0xba2c, 0xba2d, 0xba2e, 0xba2f, 0xba30, 0xba31, 0xba32, 0xba33, /*xd0-0xdf*/
04980 0xba34, 0xba35, 0xba36, 0xba37, 0xba38, 0xba39, 0xba3a, 0xba3b, /*xd8-0xdf*/
04981 0xba3c, 0xba3d, 0xba3e, 0xba3f, 0xba40, 0xba41, 0xba42, 0xba43, /*xe0-0xef*/
04982 0xba44, 0xba45, 0xba46, 0xba47, 0xba48, 0xba49, 0xba4a, 0xba4b, /*xe8-0xef*/
04983 0xba4c, 0xba4d, 0xba4e, 0xba4f, 0xba50, 0xba51, 0xba52, 0xba53, /*xf0-0xf7*/
04984 0xba54, 0xba55, 0xba56, 0xba57, 0xba58, 0xba59, 0xba5a, 0xba5b, /*xf8-0xff*/
04985 /* 0x7d00 */
04986 0xbc6f, 0xbc70, 0xbc71, 0xbc72, 0xbc73, 0xbc74, 0xbc75, 0xbc76, /*x00-0x07*/
04987 0xbc77, 0xbc78, 0xbc79, 0xbc7a, 0xbc7b, 0xbc7c, 0xbc7d, 0xbc7e, /*x08-0x0f*/
04988 0xbc7f, 0xbc80, 0xbc81, 0xbc82, 0xbc83, 0xbc84, 0xbc85, 0xbc86, /*x10-0x17*/
04989 0xbc87, 0xbc88, 0xbc89, 0xbc8a, 0xbc8b, 0xbc8c, 0xbc8d, 0xbc8e, /*x18-0x1f*/
04990 0xbc8f, 0xbc90, 0xbc91, 0xbc92, 0xbc93, 0xbc94, 0xbc95, 0xbc96, /*x20-0x27*/
04991 0xbc97, 0xbc98, 0xbc99, 0xbca0, 0xbca1, 0xbca2, 0xbca3, /*x28-0x2f*/
04992 0xbca4, 0xbca5, 0xbca6, 0xbca7, 0xbca8, 0xbca9, 0xbcaa, 0xbcab, /*x30-0x37*/
04993 0xbcad, 0xbcae, 0xbcaf, 0xbc00, 0xbc01, 0xbc02, 0xbc03, /*x38-0x3f*/
04994 0xbc04, 0xbc05, 0xbc06, 0xbc07, 0xbc08, 0xbc09, 0xbc0a, 0xbc0b, /*x40-0x47*/
04995 0xbc0c, 0xbc0d, 0xbc0e, 0xbc0f, 0xbc10, 0xbc11, 0xbc12, 0xbc13, /*x48-0x4f*/
04996 0xbc14, 0xbc15, 0xbc16, 0xbc17, 0xbc18, 0xbc19, 0xbc1a, 0xbc1b, /*x50-0x57*/
04997 0xbc1c, 0xbc1d, 0xbc1e, 0xbc1f, 0xbc20, 0xbc21, 0xbc22, 0xbc23, /*x58-0x5f*/
04998 0xbc24, 0xbc25, 0xbc26, 0xbc27, 0xbc28, 0xbc29, 0xbc2a, 0xbc2b, /*x60-0x67*/
04999 0xbc2c, 0xbc2d, 0xbc2e, 0xbc2f, 0xbc30, 0xbc31, 0xbc32, 0xbc33, /*x68-0x6f*/
05000 0xbc34, 0xbc35, 0xbc36, 0xbc37, 0xbc38, 0xbc39, 0xbc3a, 0xbc3b, /*x70-0x77*/
05001 0xbc3c, 0xbc3d, 0xbc3e, 0xbc3f, 0xbc40, 0xbc41, 0xbc42, 0xbc43, /*x78-0x7f*/
05002 0xbc44, 0xbc45, 0xbc46, 0xbc47, 0xbc48, 0xbc49, 0xbc4a, 0xbc4b, /*x80-0x87*/
05003 0xbc4c, 0xbc4d, 0xbc4e, 0xbc4f, 0xbc50, 0xbc51, 0xbc52, 0xbc53, /*x88-0x8f*/
05004 0xbc54, 0xbc55, 0xbc56, 0xbc57, 0xbc58, 0xbc59, 0xbc5a, 0xbc5b, /*x90-0x97*/
05005 0xbc5c, 0xbc5d, 0xbc5e, 0xbc5f, 0xbc60, 0xbc61, 0xbc62, 0xbc63, /*x98-0x9f*/
05006 0xbc64, 0xbc65, 0xbc66, 0xbc67, 0xbc68, 0xbc69, 0xbc6a, 0xbc6b, /*xa0-0xa7*/
05007 0xbc6c, 0xbc6d, 0xbc6e, 0xbc6f, 0xbc70, 0xbc71, 0xbc72, 0xbc73, /*xa8-0xaf*/
05008 0xbc74, 0xbc75, 0xbc76, 0xbc77, 0xbc78, 0xbc79, 0xbc7a, 0xbc7b, /*xb0-0xb7*/
05009 0xbc7c, 0xbc7d, 0xbc7e, 0xbc7f, 0xbc80, 0xbc81, 0xbc82, 0xbc83, /*xb8-0xbf*/
05010 0xbc84, 0xbc85, 0xbc86, 0xbc87, 0xbc88, 0xbc89, 0xbc8a, 0xbc8b, /*xc0-0xcf*/
05011 0xbc8c, 0xbc8d, 0xbc8e, 0xbc8f, 0xbc90, 0xbc91, 0xbc92, 0xbc93, /*xc8-0xcf*/
05012 0xbc94, 0xbc95, 0xbc96, 0xbc97, 0xbc98, 0xbc99, 0xbca0, 0xbca1, /*xd0-0xdf*/
05013 0xbca2, 0xbca3, 0xbca4, 0xbca5, 0xbca6, 0xbca7, 0xbca8, 0xbca9, /*xd8-0xdf*/
05014 0xbcaa, 0xbcab, 0xbcac, 0xbcad, 0xbcae, 0xbcaf, 0xbc00, 0xbc01, /*xe0-0xef*/
05015 0xbc02, 0xbc03, 0xbc04, 0xbc05, 0xbc06, 0xbc07, 0xbc08, 0xbc09, /*xe8-0xef*/
05016 0xbc0a, 0xbc0b, 0xbc0c, 0xbc0d, 0xbc0e, 0xbc0f, 0xbc10, 0xbc11, /*xf0-0xf7*/
05017 0xbc12, 0xbc13, 0xbc14, 0xbc15, 0xbc16, 0xbc17, 0xbc18, 0xbc19, /*xf8-0xff*/
05018 /* 0x7e00 */
05019 0xbf45, 0xbf46, 0xbf47, 0xbf48, 0xbf49, 0xbf4a, 0xbf4b, 0xbf4c, /*x00-0x07*/
05020 0xbf4d, 0xbf4e, 0xbf4f, 0xbf50, 0xbf51, 0xbf52, 0xbf53, 0xbf54, /*x08-0x0f*/
05021 0xbf55, 0xbf56, 0xbf57, 0xbf58, 0xbf59, 0xbf5a, 0xbf5b, 0xbf5c, /*x10-0x17*/
05022 0xbf5d, 0xbf5e, 0xbf5f, 0xbf60, 0xbf61, 0xbf62, 0xbf63, 0xbf64, /*x18-0x1f*/
05023 0xbf65, 0xbf66, 0xbf67, 0xbf68, 0xbf69, 0xbf6a, 0xbf6b, 0xbf6c, /*x20-0x27*/
05024 0xbf6d, 0xbf6e, 0xbf6f, 0xbf70, 0xbf71, 0xbf72, 0xbf73, 0xbf74, /*x28-0x2f*/
05025 0xbf75, 0xbf76, 0xbf77, 0xbf78, 0xbf79, 0xbf7a, 0xbf7b, 0xbf7c, /*x30-0x37*/
05026 0xbf7d, 0xbf7e, 0xbf7f, 0xbf80, 0xbf81, 0xbf82, 0xbf83, 0xbf84, /*x38-0x3f*/
05027 0xbf85, 0xbf86, 0xbf87, 0xbf88, 0xbf89, 0xbf8a, 0xbf8b, 0xbf8c, /*x40-0x47*/
05028 0xbf8d, 0xbf8e, 0xbf8f, 0xbf90, 0xbf91, 0xbf92, 0xbf93, 0xbf94, /*x48-0x4f*/
05029 0xbf95, 0xbf96, 0xbf97, 0xbf98, 0xbf99, 0xbfa0, 0xbfa1, 0xbfa2, /*x50-0x57*/
05030 0xbfa3, 0xbfa4, 0xbfa5, 0xbfa6, 0xbfa7, 0xbfa8, 0xbfa9, 0xbfab, /*x58-0x5f*/
05031 0xbfac, 0xbfad, 0xbfae, 0xbfaf, 0xbfb0, 0xbfb1, 0xbfb2, 0xbfb3, /*x60-0x67*/
05032 0xbfb4, 0xbfb5, 0xbfb6, 0xbfb7, 0xbfb8, 0xbfb9, 0xbfba, 0xbfbb, /*x68-0x6f*/
05033 0xbfbc, 0xbfbd, 0xbfbe, 0xbfbf, 0xbfc0, 0xbfc1, 0xbfc2, 0xbfc3, /*x70-0x77*/
05034 0xbfc4, 0xbfc5, 0xbfc6, 0xbfc7, 0xbfc8, 0xbfc9, 0xbfca, 0xbfcb, /*x78-0x7f*/
05035 0xbfcc, 0xbfcd, 0xbfce, 0xbfcf, 0xbfd0, 0xbfd1, 0xbfd2, 0xbfd3, /*x80-0x87*/
05036 0xbfd4, 0xbfd5, 0xbfd6, 0xbfd7, 0xbfd8, 0xbfd9, 0xbfda, 0xbfdb, /*x88-0x8f*/
05037 0xbfdc, 0xbfde, 0xbfdf, 0xbfe0, 0xbfe1, 0xbfe2, 0xbfe3, 0xbfe4, /*x90-0x97*/
05038 0xbfe5, 0xbfe6, 0xbfe7, 0xbfe8, 0xbfe9, 0xbfea, 0xbfef, 0xbff0, /*x98-0x9f*/
05039 0xbff1, 0xbff2, 0xbff3, 0xbff4, 0xbff5, 0xbff6, 0xbff7, 0xbff8, /*xa0-0xa7*/
05040 0xbff9, 0xbffa, 0xbffb, 0xbffc, 0xbffd, 0xbffe, 0xbfff, 0xc000, /*xa8-0xaf*/
05041 0xc001, 0xc002, 0xc003, 0xc004, 0xc005, 0xc006, 0xc007, 0xc008, /*xb0-0xb7*/
05042 0xc009, 0xc00a, 0xc00b, 0xc00c, 0xc00d, 0xc00e, 0xc00f, 0xc010, /*xb8-0xbf*/
05043 0xc011, 0xc012, 0xc013, 0xc014, 0xc015, 0xc016, 0xc017, 0xc018, /*xc0-0xcf*/
05044 0xc019, 0xc01a, 0xc01b, 0xc01c, 0xc01d, 0xc01e, 0xc01f, 0xc020, /*xc8-0xcf*/
05045 0xc021, 0xc022, 0xc023, 0xc024, 0xc025, 0xc026, 0xc027, 0xc028, /*xd0-0xdf*/
05046 0xc029, 0xc02a, 0xc02b, 0xc02c, 0xc02d, 0xc02e, 0xc02f, 0xc030, /*xd8-0xdf*/
05047 0xc031, 0xc032, 0xc033, 0xc034, 0xc035, 0xc036, 0xc037, 0xc038, /*xe0-0xef*/
05048 0xc039, 0xc03a, 0xc03b, 0xc03c, 0xc03d, 0xc03e, 0xc03f, 0xc040, /*xe8-0xef*/
05049 0xc041, 0xc042, 0xc043, 0xc044, 0xc045, 0xc046, 0xc047, 0xc048, /*xf0-0xf7*/
05050 0xc049, 0xc04a, 0xc04b, 0xc04c, 0xc04d, 0xc04e, 0xc04f, 0xc050, /*xf8-0xff*/
05051 /* 0x7f00 */
```

```

05052 0xd7ba, 0xe7bb, 0xe7bc, 0xe7bd, 0xbcea, 0xc3e5, 0xc0c2, 0xe7be, /*x00-0x07*/
05053 0xe7bf, 0xbca9, 0xc088, 0xe7c0, 0xe7c1, 0xe7b6, 0xb6d0, 0xe7c2, /*x08-0x0f*/
05054 0xc089, 0xe7c3, 0xe7c4, 0xbbaa, 0xb5de, 0xc2c6, 0xb1e0, 0xe7c5, /*x10-0x17*/
05055 0xd4b5, 0xe7c6, 0xb8bf, 0xe7c8, 0xe7c7, 0xb7ec, 0xc08a, 0xe7c9, /*x18-0x1f*/
05056 0xb2f8, 0xe7ca, 0xe7cb, 0xe7cc, 0xe7cd, 0xe7ce, 0xe7cf, 0xe7d0, /*x20-0x27*/
05057 0xd3a7, 0xcbbf, 0xe7d1, 0xe7d2, 0xe7d3, 0xe7d4, 0xc9c9, 0xe7d5, /*x28-0x2f*/
05058 0xe7d6, 0xe7d7, 0xe7d8, 0xe7d9, 0xbdc9, 0xe7da, 0xf3be, 0xc08b, /*x30-0x37*/
05059 0xb8d7, 0xc08c, 0xc8b1, 0xc08d, 0xc08e, 0xc08f, 0xc090, 0xc091, /*x38-0x3f*/
05060 0xc092, 0xc093, 0xf3bf, 0xc094, 0xf3c0, 0xf3c1, 0xc095, 0xc096, /*x40-0x47*/
05061 0xc097, 0xc098, 0xc099, 0xc09a, 0xc09b, 0xc09c, 0xc09d, 0xc09e, /*x48-0x4f*/
05062 0xb9de, 0xcdf8, 0xc09f, 0xc0a0, 0xd8e8, 0xbab1, 0xc140, 0xc2de, /*x50-0x57*/
05063 0xeeb7, 0xc141, 0xb7a3, 0xc142, 0xc143, 0xc144, 0xc145, 0xeeb9, /*x58-0x5f*/
05064 0xc146, 0xeeb8, 0xb0d5, 0xc147, 0xc148, 0xc149, 0xc14a, 0xc14b, /*x60-0x67*/
05065 0xeebb, 0xd5d6, 0xd7ef, 0xc14c, 0xc14d, 0xc14e, 0xd6c3, 0xc14f, /*x68-0x6f*/
05066 0xc150, 0xeebd, 0xcaf0, 0xc151, 0xeebc, 0xc152, 0xc153, 0xc154, /*x70-0x77*/
05067 0xc155, 0xeebe, 0xc156, 0xc157, 0xc158, 0xc159, 0xeec0, 0xc15a, /*x78-0x7f*/
05068 0xc15b, 0xeebf, 0xc15c, 0xc15d, 0xc15e, 0xc15f, 0xc160, 0xc161, /*x80-0x87*/
05069 0xc162, 0xc163, 0xd1f2, 0xc164, 0xc7bc, 0xc165, 0xc3c0, 0xc166, /*x88-0x8f*/
05070 0xc167, 0xc168, 0xc169, 0xc16a, 0xb8e1, 0xc16b, 0xc16c, 0xc16d, /*x90-0x97*/
05071 0xc16e, 0xc16f, 0xc1e7, 0xc170, 0xc171, 0xf4c6, 0xd0df, 0xf4c7, /*x98-0x9f*/
05072 0xc172, 0xcfdb, 0xc173, 0xc174, 0xc8ba, 0xc175, 0xc176, 0xf4c8, /*xa0-0xaf*/
05073 0xc177, 0xc178, 0xc179, 0xc17a, 0xc17b, 0xc17c, 0xc17d, 0xf4c9, /*xa8-0xaf*/
05074 0xf4ca, 0xc17e, 0xf4cb, 0xc180, 0xc181, 0xc182, 0xc183, 0xc184, /*xb0-0xb7*/
05075 0xd9fa, 0xb8fe, 0xc185, 0xc186, 0xc187, 0xd3f0, 0xc187, 0xf4e0, /*xb8-0xbf*/
05076 0xc188, 0xccec, 0xc189, 0xc18a, 0xc18b, 0xb3e1, 0xc18c, 0xc18d, /*xc0-0xc7*/
05077 0xc18e, 0xc18f, 0xf1b4, 0xc190, 0xd2ee, 0xc191, 0xf4e1, 0xc192, /*xc8-0xcf*/
05078 0xc193, 0xc194, 0xc195, 0xc196, 0xcfe8, 0xf4e2, 0xc197, 0xc198, /*xd0-0xd7*/
05079 0xc7cc, 0xc199, 0xc19a, 0xc19b, 0xc19c, 0xc19d, 0xc19e, 0xb5d4, /*xd8-0xdf*/
05080 0xb4e4, 0xf4e4, 0xc19f, 0xc1a0, 0xc240, 0xf4e3, 0xf4e5, 0xc241, /*xe0-0xe7*/
05081 0xc242, 0xf4e6, 0xc243, 0xc244, 0xc246, 0xf4e7, 0xc247, 0xc247, /*xe8-0xef*/
05082 0xbab2, 0xb0bf, 0xc248, 0xf4e8, 0xc249, 0xc24a, 0xc24b, 0xc24c, /*xf0-0xf7*/
05083 0xc24d, 0xc24e, 0xc24f, 0xb7ad, 0xd2ed, 0xc250, 0xc251, 0xc252, /*xf8-0xff*/
05084 /* 0x8000 */
05085 0xd2ab, 0xc0cf, 0xc253, 0xbfbf, 0xeba3, 0xd5df, 0xeac8, 0xc254, /*x00-0x07*/
05086 0xc255, 0xc256, 0xc257, 0xf1f3, 0xb6f8, 0xcba3, 0xc258, 0xc259, /*x08-0x0f*/
05087 0xc4cd, 0xc25a, 0xf1e7, 0xc25b, 0xb8fb, 0xf1e9, 0xbac4, 0xbac4, /*x10-0x17*/
05088 0xd4c5, 0xb0d2, 0xc25c, 0xc25d, 0xf1ea, 0xc25e, 0xc25f, 0xc260, /*x18-0x1f*/
05089 0xf1eb, 0xc261, 0xf1ec, 0xc262, 0xc263, 0xf1ed, 0xf1ee, 0xf1ef, /*x20-0x27*/
05090 0xf1f1, 0xf1f0, 0xc264, 0xc265, 0xc266, 0xc267, 0xc268, 0xc268, /*x28-0x2f*/
05091 0xc269, 0xf1f2, 0xc26a, 0xb6fa, 0xc26b, 0xf1f4, 0xd2ae, 0xdec7, /*x30-0x37*/
05092 0xc26c, 0xc26c, 0xc26d, 0xb3dc, 0xc26e, 0xb5a2, 0xc26f, 0xb9a2, /*x38-0x3f*/
05093 0xc270, 0xf1fa, 0xc271, 0xc272, 0xc273, 0xf1f6, 0xc274, 0xc274, /*x40-0x47*/
05094 0xc275, 0xc276, 0xc1c4, 0xc1fb, 0xd6b0, 0xf1f7, 0xc277, 0xc278, /*x48-0x4f*/
05095 0xc279, 0xc27a, 0xf1fb, 0xc27b, 0xc1aa, 0xc27c, 0xc27d, 0xc27e, /*x50-0x57*/
05096 0xc6b8, 0xc280, 0xc281, 0xc282, 0xc283, 0xc284, 0xc285, 0xc285, /*x58-0x5f*/
05097 0xc286, 0xc287, 0xc288, 0xc289, 0xc28a, 0xc28b, 0xc28c, 0xc28d, /*x60-0x67*/
05098 0xc28e, 0xf1fa, 0xb4cf, 0xc28f, 0xc290, 0xc291, 0xc292, 0xc293, /*x68-0x6f*/
05099 0xc294, 0xf1fa, 0xc295, 0xc296, 0xc297, 0xc298, 0xc299, 0xc29a, /*x70-0x77*/
05100 0xc29b, 0xc29c, 0xc29d, 0xc29e, 0xc29f, 0xc2a0, 0xc340, 0xedb2, /*x78-0x7f*/
05101 0xedb1, 0xc341, 0xc342, 0xcbe0, 0xd2de, 0xc343, 0xc343, 0xd5d8, /*x80-0x87*/
05102 0xc344, 0xc8e2, 0xc345, 0xc0df, 0xbca1, 0xc346, 0xc347, 0xc348, /*x88-0x8f*/
05103 0xc349, 0xc34a, 0xc34b, 0xebc1, 0xc34c, 0xc34d, 0xd0a4, 0xc34e, /*x90-0x97*/
05104 0xd6e2, 0xc34f, 0xb6c7, 0xb8d8, 0xebc0, 0xb8ce, 0xc350, 0xebbf, /*x98-0x9f*/
05105 0xb3a6, 0xb9c9, 0xd6ab, 0xc351, 0xb7f4, 0xb7ca, 0xc352, 0xc353, /*xa0-0xaf*/
05106 0xc354, 0xbce7, 0xb7be, 0xebc6, 0xc355, 0xebc7, 0xb0b9, 0xbfcf, /*xa8-0xaf*/
05107 0xc356, 0xebc5, 0xd3fd, 0xc357, 0xebc8, 0xc358, 0xc359, 0xebc9, /*xb0-0xb7*/
05108 0xc35a, 0xc35b, 0xb7ce, 0xc35c, 0xebc2, 0xebc4, 0xc9f6, 0xd6d7, /*xb8-0xbf*/
05109 0xd5cd, 0xd0b2, 0xebcf, 0xebc8, 0xebd0, 0xc35d, 0xb5a8, 0xc35e, /*xc0-0xc7*/
05110 0xc35f, 0xc360, 0xc361, 0xc362, 0xb1b3, 0xebd2, 0xc363, 0xc363, /*xc8-0xcf*/
05111 0xc364, 0xc365, 0xc366, 0xc367, 0xc368, 0xc369, 0xc5d6, 0xebd3, /*xd0-0xd7*/
05112 0xc36a, 0xebd1, 0xc5df, 0xebce, 0xc36a, 0xebd5, 0xb0fb, 0xc36b, /*xd8-0xdf*/
05113 0xc36c, 0xbafa, 0xc36d, 0xc36e, 0xd8b7, 0xf1e3, 0xc36f, 0xebca, /*xe0-0xe7*/
05114 0xebcb, 0xebcc, 0xebcd, 0xebd6, 0xe6c0, 0xebd9, 0xc370, 0xbfe8, /*xe8-0xef*/
05115 0xd2c8, 0xebd7, 0xebdc, 0xb8ec, 0xebd8, 0xc371, 0xbdba, 0xc372, /*xf0-0xf7*/
05116 0xd0d8, 0xc373, 0xb0b7, 0xc374, 0xebdd, 0xc4dc, 0xc375, 0xc376, /*xf8-0xff*/
05117 /* 0x8100 */
05118 0xc377, 0xc378, 0xd6ac, 0xc379, 0xc37a, 0xc37b, 0xb4e0, 0xc37c, /*x00-0x07*/
05119 0xc37d, 0xc2f6, 0xbcb9, 0xc37e, 0xc380, 0xebda, 0xebdb, 0xd4e0, /*x08-0x0f*/
05120 0xc6ea, 0xc4d4, 0xebdf, 0xc5a7, 0xd9f5, 0xc381, 0xb2b1, 0xc382, /*x10-0x17*/
05121 0xebef, 0xc383, 0xbdc5, 0xc384, 0xc385, 0xc386, 0xebef, 0xc387, /*x18-0x1f*/
05122 0xc388, 0xc389, 0xc38a, 0xc38b, 0xc38c, 0xc38d, 0xc38e, 0xc38f, /*x20-0x27*/
05123 0xc390, 0xc391, 0xc392, 0xc393, 0xebef, 0xc394, 0xc395, 0xb8ac, /*x28-0x2f*/
05124 0xc396, 0xcdd1, 0xebef, 0xc397, 0xc398, 0xc399, 0xebef, 0xc39a, /*x30-0x37*/
05125 0xc1b3, 0xc39b, 0xc39c, 0xc39d, 0xc39e, 0xc39f, 0xc6a2, 0xc3a0, /*x38-0x3f*/
05126 0xc440, 0xc441, 0xc442, 0xc443, 0xc444, 0xc445, 0xc446, 0xc446, /*x40-0x47*/
05127 0xebef, 0xc447, 0xc0b0, 0xd2b8, 0xebef, 0xc448, 0xc449, 0xc44a, /*x48-0x4f*/
05128 0xb8af, 0xb8ad, 0xc44b, 0xebef, 0xc44c, 0xc44d, 0xc44e, 0xc44d, /*x50-0x57*/
05129 0xc44e, 0xebef, 0xebef, 0xc44f, 0xc450, 0xc451, 0xc452, 0xc453, /*x58-0x5f*/
05130 0xebef, 0xc454, 0xc455, 0xc456, 0xc457, 0xd0c8, 0xc458, 0xebf2, /*x60-0x67*/
05131 0xc459, 0xebef, 0xc45a, 0xc45b, 0xc45c, 0xebf1, 0xc8f9, 0xc45d, /*x68-0x6f*/
05132 0xd1fc, 0xebef, 0xc45e, 0xc45f, 0xebef, 0xc460, 0xc461, 0xc462, /*x70-0x77*/
05133 0xc463, 0xb8b9, 0xc4fd, 0xc4e5, 0xebef, 0xebf0, 0xc464, 0xc465, /*x78-0x7f*/
05134 0xb0f2, 0xc464, 0xebf6, 0xc465, 0xc466, 0xc467, 0xc468, 0xc469, /*x80-0x87*/
05135 0xebf5, 0xc46a, 0xb2b2, 0xc46b, 0xc46c, 0xc46d, 0xc46e, 0xb8e0, /*x88-0x8f*/
05136 0xc46f, 0xebf7, 0xc470, 0xc471, 0xc472, 0xc473, 0xc474, 0xc475, /*x90-0x97*/
05137 0xb1ec, 0xc476, 0xc477, 0xc478, 0xc479, 0xc47a, 0xc47b, 0xc47c, /*x98-0x9f*/
05138 0xc47a, 0xc47b, 0xc47c, 0xebf9, 0xc47d, 0xc47e, 0xeca2, 0xc480, /*xa0-0xaf*/

```

```
05139 0xc5f2, 0xc481, 0xebfa, 0xc482, 0xc483, 0xc484, 0xc485, 0xc486, /*0xa8-0xaf*/
05140 0xc487, 0xc488, 0xc489, 0xc9c5, 0xc48a, 0xc48b, 0xc48c, 0xc48d, /*0xb0-0xb7*/
05141 0xc48e, 0xc48f, 0xe2df, 0xebfe, 0xc490, 0xc491, 0xc492, 0xc493, /*0xb8-0xbf*/
05142 0xcdce, 0xeca1, 0xb1db, 0xd3b7, 0xc494, 0xc495, 0xd2dc, 0xc496, /*0xc0-0xc7*/
05143 0xc497, 0xc498, 0xebfd, 0xc499, 0xebfb, 0xc49a, 0xc49b, 0xc49c, /*0xc8-0xcf*/
05144 0xc49d, 0xc49e, 0xc49f, 0xc4a0, 0xc540, 0xc541, 0xc542, 0xc543, /*0xd0-0xd7*/
05145 0xc544, 0xc545, 0xc546, 0xc547, 0xc548, 0xc549, 0xc54a, 0xc54b, /*0xd8-0xdf*/
05146 0xc54c, 0xc54d, 0xc54e, 0xb3bc, 0xc54f, 0xc550, 0xc551, 0xeab0, /*0xe0-0xe7*/
05147 0xc552, 0xc553, 0xd7d4, 0xc554, 0xf4ab, 0xb3f4, 0xc555, 0xc556, /*0xe8-0xef*/
05148 0xc557, 0xc558, 0xc559, 0xd6c1, 0xd6c2, 0xc55a, 0xc55b, 0xc55c, /*0xf0-0xf7*/
05149 0xc55d, 0xc55e, 0xc55f, 0xd5e9, 0xbeca, 0xc560, 0xf4a7, 0xc561, /*0xf8-0xff*/
05150 /* 0x8200 */
05151 0xd2a8, 0xf4a8, 0xf4a9, 0xc562, 0xf4aa, 0xbecb, 0xd3df, 0xc563, /*0x00-0x07*/
05152 0xc564, 0xc565, 0xc566, 0xc567, 0xc9e0, 0xc9e1, 0xc568, 0xc569, /*0x08-0x0f*/
05153 0xf3c2, 0xc56a, 0xcae6, 0xc56b, 0xccf2, 0xc56c, 0xc56d, 0xc56e, /*0x10-0x17*/
05154 0xc56f, 0xc570, 0xc571, 0xe2b6, 0xcbb4, 0xc572, 0xcee8, 0xd6db, /*0x18-0x1f*/
05155 0xc573, 0xf4ad, 0xf4ae, 0xf4af, 0xc574, 0xc575, 0xc576, 0xc577, /*0x20-0x27*/
05156 0xf4b2, 0xc578, 0xbabd, 0xf4b3, 0xb0e3, 0xf4b0, 0xc579, 0xf4b1, /*0x28-0x2f*/
05157 0xbda2, 0xb2d5, 0xc57a, 0xf4b6, 0xf4b7, 0xb6e6, 0xb2b0, 0xcfcf, /*0x30-0x37*/
05158 0xf4b4, 0xb4ac, 0xc57b, 0xf4b5, 0xc57c, 0xc57d, 0xf4b8, 0xc57e, /*0x38-0x3f*/
05159 0xc580, 0xc581, 0xc582, 0xc583, 0xf4b9, 0xc584, 0xc585, 0xcda7, /*0x40-0x47*/
05160 0xc586, 0xf4ba, 0xc587, 0xf4bb, 0xc588, 0xc589, 0xc58a, 0xf4bc, /*0x48-0x4f*/
05161 0xc58b, 0xc58c, 0xc58d, 0xc58e, 0xc58f, 0xc590, 0xc591, 0xc592, /*0x50-0x57*/
05162 0xc593, 0xc594, 0xf4bd, 0xc595, 0xc596, 0xc597, 0xf4be, /*0x58-0x5f*/
05163 0xc598, 0xc599, 0xc59a, 0xc59b, 0xc59c, 0xc59d, 0xc59e, 0xc59f, /*0x60-0x6f*/
05164 0xf4bf, 0xc5a0, 0xc640, 0xc641, 0xc642, 0xc643, 0xf4de, 0xc1bc, /*0x68-0x6f*/
05165 0xbce8, 0xc644, 0xc9ab, 0xd1de, 0xe5f5, 0xc645, 0xc646, 0xc647, /*0x70-0x77*/
05166 0xc648, 0xdcbb, 0xd2d5, 0xc649, 0xc64a, 0xdcbb, 0xb0ac, 0xdcbb, /*0x78-0x7f*/
05167 0xc64b, 0xc64c, 0xbdda, 0xc64d, 0xdcbb, 0xc64e, 0xc64f, 0xc650, /*0x80-0x87*/
05168 0xd8c2, 0xc651, 0xdcbb, 0xd3f3, 0xc652, 0xc653, 0xdcba, 0xdcbb, /*0x88-0x8f*/
05169 0xc653, 0xdcbb, 0xc3a2, 0xc654, 0xc655, 0xc656, 0xc657, 0xdcbb, /*0x90-0x97*/
05170 0xdcc5, 0xdcbb, 0xc658, 0xc659, 0xc65a, 0xd6a5, 0xc65a, 0xdccf, /*0x98-0x9f*/
05171 0xc65b, 0xdcc5, 0xc65c, 0xc65d, 0xdcc2, 0xbde6, 0xc2ab, 0xc65e, /*0xa0-0xaf*/
05172 0xdcbb, 0xdccb, 0xdcce, 0xdcbe, 0xb7d2, 0xb0c5, 0xdcc7, 0xd0be, /*0xa8-0xaf*/
05173 0xdcc1, 0xbba8, 0xc65f, 0xb7bc, 0xdccc, 0xc660, 0xc661, 0xdcc6, /*0xb0-0xb7*/
05174 0xdccb, 0xc7db, 0xc662, 0xc663, 0xc664, 0xd1bf, 0xdccc, 0xc665, /*0xb8-0xbf*/
05175 0xc666, 0xdcca, 0xc667, 0xc668, 0xdcd0, 0xc669, 0xc66a, 0xcead, /*0xc0-0xc7*/
05176 0xdcc2, 0xc66b, 0xdcc3, 0xdcc4, 0xdcc5, 0xb2d4, 0xdcd1, 0xcbb5, /*0xc8-0xcf*/
05177 0xc66c, 0xd4b7, 0xdccb, 0xdccf, 0xc66d, 0xdcc6, 0xc66d, 0xc3e7, /*0xd0-0xd7*/
05178 0xdcdc, 0xc66e, 0xc66f, 0xbfc1, 0xdcd9, 0xc670, 0xb0fa, 0xb9b6, /*0xd8-0xdf*/
05179 0xdce5, 0xdcd3, 0xc671, 0xdccc, 0xdcd6, 0xc8f4, 0xbfe0, 0xc672, /*0xe0-0xe7*/
05180 0xc673, 0xc674, 0xc675, 0xc9bb, 0xc676, 0xc677, 0xc678, 0xb1bd, /*0xe8-0xef*/
05181 0xc679, 0xd3a2, 0xc67a, 0xc67b, 0xdcd4, 0xc67c, 0xc67d, 0xdcd5, /*0xf0-0xf7*/
05182 0xc67e, 0xc6bb, 0xc680, 0xdcdc, 0xc681, 0xc682, 0xc683, 0xc684, /*0xf8-0xff*/
05183 /* 0x8300 */
05184 0xc685, 0xd7c2, 0xc3af, 0xb7b6, 0xc7d1, 0xc3a9, 0xdce2, 0xdcd8, /*0x00-0x07*/
05185 0xdceb, 0xdcd4, 0xc686, 0xc687, 0xdcd5, 0xc688, 0xbea5, 0xdcd7, /*0x08-0x0f*/
05186 0xc689, 0xdce0, 0xc68a, 0xc68b, 0xdce3, 0xdce4, 0xc68c, 0xdcd8, /*0x10-0x17*/
05187 0xc68d, 0xc68e, 0xdce1, 0xdda2, 0xdce7, 0xc68f, 0xc690, 0xc691, /*0x18-0x1f*/
05188 0xc692, 0xc693, 0xc694, 0xc695, 0xc696, 0xc697, 0xc698, 0xbceb, /*0x20-0x27*/
05189 0xb4c4, 0xc699, 0xc69a, 0xc3a3, 0xb2e7, 0xdccf, 0xc69b, 0xdcd9, /*0x28-0x2f*/
05190 0xc69c, 0xdcef, 0xc69d, 0xdccf, 0xdcee, 0xd2f0, 0xb2e8, 0xc69e, /*0x30-0x37*/
05191 0xc8d7, 0xc8e3, 0xdccf, 0xc69f, 0xdced, 0xc6a0, 0xc740, 0xc741, /*0x38-0x3f*/
05192 0xdccf, 0xc742, 0xc743, 0xdccf, 0xc744, 0xc745, 0xbea3, 0xdcd4, /*0x40-0x47*/
05193 0xc746, 0xb2dd, 0xc747, 0xc748, 0xc749, 0xc74a, 0xc74b, 0xdccf, /*0x48-0x4f*/
05194 0xbcf6, 0xdce8, 0xbbc4, 0xc74c, 0xc0f3, 0xc74d, 0xc74e, 0xc74f, /*0x50-0x57*/
05195 0xc750, 0xc751, 0xbcd4, 0xdce9, 0xdcea, 0xc752, 0xdccf, 0xdccf, /*0x58-0x5f*/
05196 0xdccf, 0xb5b4, 0xc753, 0xc8d9, 0xbbe7, 0xdccf, 0xdccf, 0xd3ab, /*0x60-0x6f*/
05197 0xdda1, 0xdda3, 0xdda5, 0xd2f1, 0xdda4, 0xdda6, 0xdda7, 0xd2a9, /*0x68-0x6f*/
05198 0xc754, 0xc755, 0xc756, 0xc757, 0xc758, 0xc759, 0xc75a, 0xbac9, /*0x70-0x77*/
05199 0xdda9, 0xc75b, 0xc75c, 0xddb6, 0xddb1, 0xddb4, 0xc75d, 0xc75e, /*0x78-0x7f*/
05200 0xc75f, 0xc760, 0xc761, 0xc762, 0xc763, 0xddb0, 0xc6ce, 0xc764, /*0x80-0x87*/
05201 0xc765, 0xc0f2, 0xc766, 0xc767, 0xc768, 0xc769, 0xc9a1, 0xc76a, /*0x88-0x8f*/
05202 0xc76b, 0xc76c, 0xdcec, 0xddae, 0xc76d, 0xc76e, 0xc76f, 0xc770, /*0x90-0x97*/
05203 0xddb7, 0xc771, 0xc772, 0xdccf, 0xddaf, 0xc773, 0xddb8, 0xc774, /*0x98-0x9f*/
05204 0xddac, 0xc775, 0xc776, 0xc777, 0xc778, 0xc779, 0xc77a, 0xc77b, /*0xa0-0xaf*/
05205 0xddb9, 0xddb3, 0xddad, 0xc4aa, 0xc77c, 0xc77d, 0xc77e, 0xc780, /*0xa8-0xaf*/
05206 0xdda8, 0xc0b3, 0xc1ab, 0xddaa, 0xddab, 0xc781, 0xddb2, 0xbbf1, /*0xb0-0xbf*/
05207 0xddb5, 0xd3a8, 0xddba, 0xc782, 0xddbb, 0xc3a7, 0xc783, 0xc784, /*0xb8-0xbf*/
05208 0xdddd, 0xddbc, 0xc785, 0xc786, 0xc787, 0xdddd, 0xc788, 0xb9bd, /*0xc0-0xc7*/
05209 0xc789, 0xc78a, 0xbcd5, 0xc78b, 0xbefa, 0xc78c, 0xc78d, 0xbaca, /*0xc8-0xcf*/
05210 0xc78e, 0xc78f, 0xc790, 0xc791, 0xddca, 0xc792, 0xddc5, 0xc793, /*0xd0-0xd7*/
05211 0xddbf, 0xc794, 0xc795, 0xc796, 0xb2cb, 0xddc3, 0xc797, 0xddcb, /*0xd8-0xdf*/
05212 0xb2a4, 0xdddd, 0xc798, 0xc799, 0xc79a, 0xddbe, 0xc79b, 0xc79c, /*0xe0-0xe7*/
05213 0xc79d, 0xc6d0, 0xdddd, 0xc79e, 0xc79f, 0xc7a0, 0xc840, 0xc841, /*0xe8-0xef*/
05214 0xdddd, 0xc1e2, 0xb7c6, 0xc842, 0xc843, 0xc844, 0xc845, 0xc846, /*0xf0-0xf7*/
05215 0xddce, 0xddcf, 0xc847, 0xc848, 0xc849, 0xddc4, 0xc84a, 0xc84b, /*0xf8-0xff*/
05216 /* 0x8400 */
05217 0xc84c, 0xdddb, 0xc84d, 0xdddd, 0xddc1, 0xc84e, 0xddc9, 0xc84f, /*0x00-0x07*/
05218 0xc850, 0xc851, 0xc852, 0xddc2, 0xc3c8, 0xc6bc, 0xceae, 0xddcc, /*0x08-0x0f*/
05219 0xc853, 0xddc8, 0xc854, 0xc855, 0xc856, 0xc857, 0xc858, 0xc859, /*0x10-0x17*/
05220 0xddc1, 0xc85a, 0xc85b, 0xc85c, 0xddc6, 0xc2dc, 0xc85d, 0xc85e, /*0x18-0x1f*/
05221 0xc85f, 0xc860, 0xc861, 0xc862, 0xd3a9, 0xd3aa, 0xdddd, 0xcff4, /*0x20-0x27*/
05222 0xc8f8, 0xc863, 0xc864, 0xc865, 0xc866, 0xc867, 0xc868, 0xc869, /*0x28-0x2f*/
05223 0xc86a, 0xddde, 0xc86b, 0xc86c, 0xc86d, 0xc86e, 0xc86f, 0xc870, /*0x30-0x37*/
05224 0xdddc, 0xc871, 0xc872, 0xc873, 0xddde, 0xc2e4, 0xc874, 0xc875, /*0x38-0x3f*/
05225 0xc876, 0xc877, 0xc878, 0xc879, 0xc87a, 0xc87b, 0xddde, 0xc87c, /*0x40-0x47*/
```

```

05226 0xc87d, 0xc87e, 0xc880, 0xc881, 0xc882, 0xc883, 0xc884, 0xc885, /*0x48-0x4f*/
05227 0xc886, 0xdddd7, 0xc887, 0xc888, 0xc889, 0xc88a, 0xc88b, 0xd6f8, /*0x50-0x57*/
05228 0xc88c, 0xdddd9, 0xdddd8, 0xb8f0, 0xdddd6, 0xc88d, 0xc88e, 0xc88f, /*0x58-0x5f*/
05229 0xc890, 0xc6cf, 0xc891, 0xb6ad, 0xc892, 0xc893, 0xc894, 0xc895, /*0x60-0x67*/
05230 0xc896, 0xddde2, 0xc897, 0xbaf9, 0xd4e1, 0xddde7, 0xc898, 0xc899, /*0x68-0x6f*/
05231 0xc89a, 0xb4d0, 0xc89b, 0xddda, 0xc89c, 0xbffb, 0xddde3, 0xc89d, /*0x70-0x77*/
05232 0xdddf, 0xc89e, 0xdddd, 0xc89f, 0xc8a0, 0xc940, 0xc941, 0xc942, /*0x78-0x7f*/
05233 0xc943, 0xc944, 0xb5d9, 0xc945, 0xc946, 0xc947, 0xc948, 0xdddb, /*0x80-0x87*/
05234 0xdddc, 0xddde, 0xc949, 0xbdaf, 0xddde4, 0xc94a, 0xddde5, 0xc94b, /*0x88-0x8f*/
05235 0xc94c, 0xc94d, 0xc94e, 0xc94f, 0xc950, 0xc951, 0xc952, 0xddf5, /*0x90-0x97*/
05236 0xc953, 0xc954, 0xc955, 0xc956, 0xc957, 0xc958, 0xc959, 0xc95a, /*0x98-0x9f*/
05237 0xc95b, 0xdddf2, 0xc95c, 0xc95d, 0xc95e, 0xc95f, 0xc960, 0xc961, /*0xa0-0xa7*/
05238 0xc962, 0xc963, 0xc964, 0xc965, 0xc966, 0xc967, 0xc968, 0xc969, /*0xa8-0xaf*/
05239 0xc96a, 0xc96b, 0xc96c, 0xc96d, 0xc96e, 0xc96f, 0xc970, 0xc971, /*0xb0-0xbf*/
05240 0xc972, 0xc973, 0xc974, 0xc975, 0xc976, 0xc977, 0xc978, 0xc979, /*0xc0-0xcf*/
05241 0xc97a, 0xc97b, 0xc97c, 0xc97d, 0xc97e, 0xc97f, 0xc980, 0xc981, /*0xd0-0xdf*/
05242 0xc982, 0xc983, 0xc984, 0xc985, 0xc986, 0xc987, 0xc988, 0xc989, /*0xe0-0xef*/
05243 0xc98a, 0xc98b, 0xc98c, 0xc98d, 0xc98e, 0xc98f, 0xc990, 0xc991, /*0xf0-0xf7*/
05244 0xc992, 0xc993, 0xc994, 0xc995, 0xc996, 0xc997, 0xc998, 0xc999, /*0xf8-0xff*/
05245 /* 0x8500 */
05250 0xc99e, 0xc99f, 0xc9a0, 0xca40, 0xca41, 0xca42, 0xca43, 0xca44, /*0x00-0x07*/
05251 0xca45, 0xca46, 0xca47, 0xca48, 0xca49, 0xca4a, 0xca4b, 0xca4c, /*0x08-0x0f*/
05252 0xca4d, 0xca4e, 0xca4f, 0xca50, 0xca51, 0xca52, 0xca53, 0xca54, /*0x10-0x17*/
05253 0xca55, 0xca56, 0xca57, 0xca58, 0xca59, 0xca5a, 0xca5b, 0xca5c, /*0x18-0x1f*/
05254 0xca5d, 0xca5e, 0xca5f, 0xca60, 0xca61, 0xca62, 0xca63, 0xca64, /*0x20-0x27*/
05255 0xca65, 0xca66, 0xca67, 0xca68, 0xca69, 0xca6a, 0xca6b, 0xca6c, /*0x28-0x2f*/
05256 0xca6d, 0xca6e, 0xca6f, 0xca70, 0xca71, 0xca72, 0xca73, 0xca74, /*0x30-0x37*/
05257 0xca75, 0xca76, 0xca77, 0xca78, 0xca79, 0xca7a, 0xca7b, 0xca7c, /*0x38-0x3f*/
05258 0xca7d, 0xca7e, 0xca7f, 0xca80, 0xca81, 0xca82, 0xca83, 0xca84, /*0x40-0x47*/
05259 0xca85, 0xca86, 0xca87, 0xca88, 0xca89, 0xca8a, 0xca8b, 0xca8c, /*0x48-0x4f*/
05260 0xca8d, 0xca8e, 0xca8f, 0xca90, 0xca91, 0xca92, 0xca93, 0xca94, /*0x50-0x57*/
05261 0xca95, 0xca96, 0xca97, 0xca98, 0xca99, 0xca9a, 0xca9b, 0xca9c, /*0x58-0x5f*/
05262 0xca9d, 0xca9e, 0xca9f, 0xcaa0, 0xcaa1, 0xcaa2, 0xcaa3, 0xcaa4, /*0x60-0x67*/
05263 0xcaa5, 0xcaa6, 0xcaa7, 0xcaa8, 0xcaa9, 0xcaaa, 0xcaab, 0xcaac, /*0x68-0x6f*/
05264 0xcaad, 0xcaae, 0xcaaf, 0xcab0, 0xcab1, 0xcab2, 0xcab3, 0xcab4, /*0x70-0x77*/
05265 0xcab5, 0xcab6, 0xcab7, 0xcab8, 0xcab9, 0xcaba, 0xcabb, 0xcabc, /*0x78-0x7f*/
05266 0xcabd, 0xcabe, 0xcabf, 0xcac0, 0xcac1, 0xcac2, 0xcac3, 0xcac4, /*0x80-0x87*/
05267 0xcac5, 0xcac6, 0xcac7, 0xcac8, 0xcac9, 0xcaca, 0xcacb, 0xcacc, /*0x88-0x8f*/
05268 0xcacd, 0xcace, 0xcacf, 0xcad0, 0xcad1, 0xcad2, 0xcad3, 0xcad4, /*0x90-0x97*/
05269 0xcad5, 0xcad6, 0xcad7, 0xcad8, 0xcad9, 0xcada, 0xcadb, 0xcadc, /*0x98-0x9f*/
05270 0xcade, 0xcadf, 0xcae0, 0xcae1, 0xcae2, 0xcae3, 0xcae4, 0xcae5, /*0xa0-0xaf*/
05271 0xcae6, 0xcae7, 0xcae8, 0xcae9, 0xcaea, 0xcaeb, 0xcaec, 0xcaed, /*0xb0-0xbf*/
05272 0xcaee, 0xcaef, 0xcab0, 0xcab1, 0xcab2, 0xcab3, 0xcab4, 0xcab5, /*0xc0-0xcf*/
05273 0xcab6, 0xcab7, 0xcab8, 0xcab9, 0xcaba, 0xcabb, 0xcabc, 0xcabd, /*0xd0-0xdf*/
05274 0xcabe, 0xcabf, 0xcac0, 0xcac1, 0xcac2, 0xcac3, 0xcac4, 0xcac5, /*0xe0-0xef*/
05275 0xcac6, 0xcac7, 0xcac8, 0xcac9, 0xcaca, 0xcacb, 0xcacd, 0xcace, /*0xf0-0xf7*/
05276 0xcacf, 0xcad0, 0xcad1, 0xcad2, 0xcad3, 0xcad4, 0xcad5, 0xcad6, /*0xf8-0xff*/
05277 0xcad7, 0xcad8, 0xcad9, 0xcada, 0xcadb, 0xcadc, 0xcade, 0xcadf, /*0x00-0x07*/
05278 0xcae0, 0xcae1, 0xcae2, 0xcae3, 0xcae4, 0xcae5, 0xcae6, 0xcae7, /*0x08-0x0f*/
05279 0xcae8, 0xcae9, 0xcaea, 0xcaeb, 0xcaec, 0xcaed, 0xcaee, 0xcaef, /*0x10-0x17*/
05280 0xcab0, 0xcab1, 0xcab2, 0xcab3, 0xcab4, 0xcab5, 0xcab6, 0xcab7, /*0x18-0x1f*/
05281 0xcab8, 0xcab9, 0xcaba, 0xcabb, 0xcabc, 0xcabd, 0xcabe, 0xcabf, /*0x20-0x27*/
05282 0xcac0, 0xcac1, 0xcac2, 0xcac3, 0xcac4, 0xcac5, 0xcac6, 0xcac7, /*0x28-0x2f*/
05283 0xcac8, 0xcac9, 0xcaca, 0xcacb, 0xcacd, 0xcace, 0xcacf, 0xcad0, /*0x30-0x37*/
05284 0xcad1, 0xcad2, 0xcad3, 0xcad4, 0xcad5, 0xcad6, 0xcad7, 0xcad8, /*0x38-0x3f*/
05285 0xcad9, 0xcada, 0xcadb, 0xcadc, 0xcade, 0xcadf, 0xcae0, 0xcae1, /*0x40-0x47*/
05286 0xcae2, 0xcae3, 0xcae4, 0xcae5, 0xcae6, 0xcae7, 0xcae8, 0xcae9, /*0x48-0x4f*/
05287 0xcaea, 0xcaeb, 0xcaec, 0xcaed, 0xcaee, 0xcaef, 0xcab0, 0xcab1, /*0x50-0x57*/
05288 0xcab2, 0xcab3, 0xcab4, 0xcab5, 0xcab6, 0xcab7, 0xcab8, 0xcab9, /*0x58-0x5f*/
05289 0xcaba, 0xcabb, 0xcabc, 0xcabd, 0xcabe, 0xcabf, 0xcac0, 0xcac1, /*0x60-0x67*/
05290 0xcac2, 0xcac3, 0xcac4, 0xcac5, 0xcac6, 0xcac7, 0xcac8, 0xcac9, /*0x68-0x6f*/
05291 0xcaca, 0xcacb, 0xcacd, 0xcace, 0xcacf, 0xcad0, 0xcad1, 0xcad2, /*0x70-0x77*/
05292 0xcad3, 0xcad4, 0xcad5, 0xcad6, 0xcad7, 0xcad8, 0xcad9, 0xcada, /*0x78-0x7f*/
05293 0xcadb, 0xcadc, 0xcade, 0xcadf, 0xcae0, 0xcae1, 0xcae2, 0xcae3, /*0x80-0x87*/
05294 0xcae4, 0xcae5, 0xcae6, 0xcae7, 0xcae8, 0xcae9, 0xcaea, 0xcaeb, /*0x88-0x8f*/
05295 0xcaec, 0xcaed, 0xcaee, 0xcaef, 0xcab0, 0xcab1, 0xcab2, 0xcab3, /*0x90-0x97*/
05296 0xcab4, 0xcab5, 0xcab6, 0xcab7, 0xcab8, 0xcab9, 0xcaba, 0xcabb, /*0x98-0x9f*/
05297 0xcabc, 0xcabd, 0xcabe, 0xcabf, 0xcac0, 0xcac1, 0xcac2, 0xcac3, /*0xa0-0xaf*/
05298 0xcac4, 0xcac5, 0xcac6, 0xcac7, 0xcac8, 0xcac9, 0xcaca, 0xcacb, /*0xb0-0xbf*/
05299 0xcacd, 0xcace, 0xcacf, 0xcad0, 0xcad1, 0xcad2, 0xcad3, 0xcad4, /*0xc0-0xcf*/
05300 0xcad5, 0xcad6, 0xcad7, 0xcad8, 0xcad9, 0xcada, 0xcadb, 0xcadc, /*0xd0-0xdf*/
05301 0xcade, 0xcadf, 0xcae0, 0xcae1, 0xcae2, 0xcae3, 0xcae4, 0xcae5, /*0xe0-0xef*/
05302 0xcae6, 0xcae7, 0xcae8, 0xcae9, 0xcaea, 0xcaeb, 0xcaec, 0xcaed, /*0xf0-0xf7*/
05303 0xcaee, 0xcaef, 0xcab0, 0xcab1, 0xcab2, 0xcab3, 0xcab4, 0xcab5, /*0xf8-0xff*/
05304 0xcab6, 0xcab7, 0xcab8, 0xcab9, 0xcaba, 0xcabb, 0xcabc, 0xcabd,
05305 0xcabe, 0xcabf, 0xcac0, 0xcac1, 0xcac2, 0xcac3, 0xcac4, 0xcac5,
05306 0xcac6, 0xcac7, 0xcac8, 0xcac9, 0xcaca, 0xcacb, 0xcacd, 0xcace,
05307 0xcacf, 0xcad0, 0xcad1, 0xcad2, 0xcad3, 0xcad4, 0xcad5, 0xcad6,
05308 0xcad7, 0xcad8, 0xcad9, 0xcada, 0xcadb, 0xcadc, 0xcade, 0xcadf,
05309 0xcae0, 0xcae1, 0xcae2, 0xcae3, 0xcae4, 0xcae5, 0xcae6, 0xcae7,
05310 0xcae8, 0xcae9, 0xcaea, 0xcaeb, 0xcaec, 0xcaed, 0xcaee, 0xcaef,
05311 0xcab0, 0xcab1, 0xcab2, 0xcab3, 0xcab4, 0xcab5, 0xcab6, 0xcab7,
05312 0xcab8, 0xcab9, 0xcaba, 0xcabb, 0xcabc, 0xcabd, 0xcabe, 0xcabf,

```

```
05313 0xd5dd, 0xf2cc, 0xf2cd, 0xf2cf, 0xf2d3, 0xcd8d, 0xcd8e, 0xcd8f, /*0xf0-0xf7*/
05314 0xf2d9, 0xd3bc, 0xcd90, 0xcd91, 0xcd92, 0xcd93, 0xb6ea, 0xcd94, /*0xf8-0xff*/
05315 /* 0x8700 */
05316 0xcaf1, 0xcd95, 0xb7e4, 0xf2d7, 0xcd96, 0xcd97, 0xcd98, 0xf2d8, /*0x00-0x07*/
05317 0xf2da, 0xf2dd, 0xf2db, 0xcd99, 0xcd9a, 0xf2dc, 0xcd9b, 0xcd9c, /*0x08-0x0f*/
05318 0xcd9d, 0xcd9e, 0xd1d1, 0xf2d1, 0xcd9f, 0xcdc9, 0xcda0, 0xcecf, /*0x10-0x17*/
05319 0xd6a9, 0xce40, 0xf2e3, 0xce41, 0xc3db, 0xce42, 0xf2e0, 0xce43, /*0x18-0x1f*/
05320 0xce44, 0xc0af, 0xf2ec, 0xf2de, 0xce45, 0xf2e1, 0xce46, 0xce47, /*0x20-0x27*/
05321 0xce48, 0xf2e8, 0xce49, 0xce4a, 0xce4b, 0xce4c, 0xf2e2, 0xce4d, /*0x28-0x2f*/
05322 0xce4e, 0xf2e7, 0xce4f, 0xce50, 0xf2e6, 0xce51, 0xce52, 0xf2e9, /*0x30-0x37*/
05323 0xce53, 0xce54, 0xce55, 0xf2df, 0xce56, 0xce57, 0xf2e4, 0xf2ea, /*0x38-0x3f*/
05324 0xce58, 0xce59, 0xce5a, 0xce5b, 0xce5c, 0xce5d, 0xce5e, 0xd3ac, /*0x40-0x47*/
05325 0xf2e5, 0xb2f5, 0xce5f, 0xce60, 0xf2f2, 0xce61, 0xd0ab, 0xce62, /*0x48-0x4f*/
05326 0xce63, 0xce64, 0xce65, 0xf2f5, 0xce66, 0xce67, 0xce68, 0xbbc8, /*0x50-0x57*/
05327 0xce69, 0xf2f9, 0xce6a, 0xce6b, 0xce6c, 0xce6d, 0xce6e, 0xce6f, /*0x58-0x5f*/
05328 0xf2f0, 0xce70, 0xce71, 0xf2f6, 0xf2f8, 0xf2fa, 0xce72, 0xce73, /*0x60-0x67*/
05329 0xce74, 0xce75, 0xce76, 0xce77, 0xce78, 0xce79, 0xf2f3, 0xce7a, /*0x68-0x6f*/
05330 0xf2f1, 0xce7b, 0xce7c, 0xce7d, 0xbafb, 0xce7e, 0xb5fb, 0xce80, /*0x70-0x77*/
05331 0xce81, 0xce82, 0xce83, 0xf2ef, 0xf2f7, 0xf2ed, 0xf2ee, 0xce84, /*0x78-0x7f*/
05332 0xce85, 0xce86, 0xf2eb, 0xf3a6, 0xce87, 0xf3a3, 0xce88, 0xce89, /*0x80-0x87*/
05333 0xf3a2, 0xce8a, 0xce8b, 0xf2f4, 0xce8c, 0xc8da, 0xce8d, 0xce8e, /*0x88-0x8f*/
05334 0xce8f, 0xce90, 0xce91, 0xf2fb, 0xce92, 0xce93, 0xce94, 0xf3a5, /*0x90-0x97*/
05335 0xce95, 0xce96, 0xce97, 0xce98, 0xce99, 0xce9a, 0xce9b, 0xc3f8, /*0x98-0x9f*/
05336 0xce9c, 0xce9d, 0xce9e, 0xce9f, 0xcea0, 0xcf40, 0xcf41, 0xcf42, /*0xa0-0xaf*/
05337 0xf2fd, 0xcf43, 0xcf44, 0xf3a7, 0xf3a9, 0xf3a4, 0xcf45, 0xf2fc, /*0xa8-0xaf*/
05338 0xcf46, 0xcf47, 0xcf48, 0xf3ab, 0xcf49, 0xf3aa, 0xcf4a, 0xcf4b, /*0xb0-0xb7*/
05339 0xcf4c, 0xcf4d, 0xc2dd, 0xcf4e, 0xcf4f, 0xf3ae, 0xcf50, 0xcf51, /*0xb8-0xbf*/
05340 0xf3b0, 0xcf52, 0xcf53, 0xcf54, 0xcf55, 0xcf56, 0xf3a1, 0xcf57, /*0xc0-0xc7*/
05341 0xcf58, 0xcf59, 0xf3b1, 0xf3ac, 0xcf5a, 0xcf5b, 0xcf5c, 0xcf5d, /*0xc8-0xcf*/
05342 0xcf5e, 0xf3af, 0xf2fe, 0xf3ad, 0xcf5f, 0xcf60, 0xcf61, 0xcf62, /*0xd0-0xd7*/
05343 0xcf63, 0xcf64, 0xcf65, 0xf3b2, 0xcf66, 0xcf67, 0xcf68, 0xcf69, /*0xd8-0xdf*/
05344 0xf3b4, 0xcf6a, 0xcf6b, 0xcf6c, 0xcf6d, 0xf3a8, 0xcf6e, 0xcf6f, /*0xe0-0xef*/
05345 0xcf70, 0xcf71, 0xf3b3, 0xcf72, 0xcf73, 0xcf74, 0xf3b5, 0xcf75, /*0xe8-0xef*/
05346 0xcf76, 0xcf77, 0xcf78, 0xcf79, 0xcf7a, 0xcf7b, 0xcf7c, 0xcf7d, /*0xf0-0xf7*/
05347 0xcf7e, 0xd0b7, 0xcf80, 0xcf81, 0xcf82, 0xcf83, 0xf3b8, 0xcf84, /*0xf8-0xff*/
05348 /* 0x8800 */
05349 0xcf85, 0xcf86, 0xcf87, 0xd9f9, 0xcf88, 0xcf89, 0xcf8a, 0xcf8b, /*0x00-0x07*/
05350 0xcf8c, 0xcf8d, 0xf3b9, 0xcf8e, 0xcf8f, 0xcf90, 0xcf91, 0xcf92, /*0x08-0x0f*/
05351 0xcf93, 0xcf94, 0xcf95, 0xf3b7, 0xcf96, 0xc8e4, 0xf3b6, 0xcf97, /*0x10-0x17*/
05352 0xcf98, 0xcf99, 0xcf9a, 0xf3ba, 0xcf9b, 0xcf9c, 0xcf9d, 0xcf9e, /*0x18-0x1f*/
05353 0xcf9f, 0xf3bb, 0xb4c0, 0xcfa0, 0xd040, 0xd041, 0xd042, 0xd043, /*0x20-0x27*/
05354 0xd044, 0xd045, 0xd046, 0xd047, 0xd048, 0xd049, 0xd04a, 0xd04b, /*0x28-0x2f*/
05355 0xd04c, 0xd04d, 0xeec3, 0xd04e, 0xd04f, 0xd050, 0xd051, 0xd052, /*0x30-0x37*/
05356 0xd053, 0xf3bc, 0xd054, 0xd055, 0xf3bd, 0xd056, 0xd057, 0xd058, /*0x38-0x3f*/
05357 0xd1aa, 0xd059, 0xd05a, 0xd05b, 0xf4ac, 0xd0c6, 0xd05c, 0xd05d, /*0x40-0x47*/
05358 0xd05e, 0xd05f, 0xd060, 0xd061, 0xd0d0, 0xd1dc, 0xd062, 0xd063, /*0x48-0x4f*/
05359 0xd064, 0xd065, 0xd066, 0xd067, 0xcfce, 0xd068, 0xd069, 0xbdd6, /*0x50-0x57*/
05360 0xd06a, 0xd1c3, 0xd06b, 0xd06c, 0xd06d, 0xd06e, 0xd06f, 0xd070, /*0x58-0x5f*/
05361 0xd071, 0xbae2, 0xe1e9, 0xd2c2, 0xf1c2, 0xb2b9, 0xd072, 0xd073, /*0x60-0x67*/
05362 0xb1ed, 0xf1c3, 0xd074, 0xc9c0, 0xb3c4, 0xd075, 0xd9f2, 0xd076, /*0x68-0x6f*/
05363 0xcba5, 0xd077, 0xf1c4, 0xd078, 0xd079, 0xd07a, 0xd07b, 0xd6d4, /*0x70-0x77*/
05364 0xd07c, 0xd07d, 0xd07e, 0xd080, 0xd081, 0xf1c5, 0xf4c0, 0xf1c6, /*0x78-0x7f*/
05365 0xd082, 0xd4ac, 0xf1c7, 0xd083, 0xb0c0, 0xf4c1, 0xd084, 0xd085, /*0x80-0x87*/
05366 0xf4c2, 0xd086, 0xd087, 0xb4fc, 0xd088, 0xc5db, 0xd089, 0xd08a, /*0x88-0x8f*/
05367 0xd08b, 0xd08c, 0xcbb, 0xd08d, 0xd08e, 0xd08f, 0xd0e4, 0xd090, /*0x90-0x97*/
05368 0xd091, 0xd092, 0xd093, 0xd094, 0xcde0, 0xd095, 0xd096, 0xd097, /*0x98-0x9f*/
05369 0xd098, 0xd099, 0xf1c8, 0xd09a, 0xd09b, 0xd09c, 0xd09d, 0xd09e, /*0xa0-0xaf*/
05370 0xd09e, 0xd09f, 0xd0a0, 0xb1bb, 0xd140, 0xcfae, 0xd141, 0xd142, /*0xa8-0xaf*/
05371 0xd143, 0xb8a4, 0xd144, 0xd145, 0xd146, 0xd147, 0xd148, 0xf1ca, /*0xb0-0xb7*/
05372 0xd149, 0xd14a, 0xd14b, 0xd14c, 0xf1cb, 0xd14d, 0xd14e, 0xd14f, /*0xb8-0xbf*/
05373 0xd150, 0xb2c3, 0xc1d1, 0xd151, 0xd152, 0xd7b0, 0xf1c9, 0xd153, /*0xc0-0xc7*/
05374 0xd154, 0xf1cc, 0xd155, 0xd156, 0xd157, 0xd158, 0xf1ce, 0xd159, /*0xc8-0xcf*/
05375 0xd15a, 0xd15b, 0xd9f6, 0xd15c, 0xd2e1, 0xd4a3, 0xd15d, 0xd15e, /*0xd0-0xd7*/
05376 0xf4c3, 0xc8b9, 0xd15f, 0xd160, 0xd161, 0xd162, 0xd163, 0xf4c4, /*0xd8-0xdf*/
05377 0xd164, 0xd165, 0xf1cd, 0xf1cf, 0xbfe3, 0xf1d0, 0xd166, 0xd167, /*0xe0-0xef*/
05378 0xf1d4, 0xd168, 0xd169, 0xd16a, 0xd16b, 0xd16c, 0xd16d, 0xd16e, /*0xe8-0xef*/
05379 0xf1d6, 0xf1d1, 0xd16f, 0xc9d1, 0xc5e1, 0xd170, 0xd171, 0xd172, /*0xf0-0xf7*/
05380 0xc2e3, 0xb9fc, 0xd173, 0xd174, 0xf1d3, 0xd175, 0xf1d5, 0xd176, /*0xf8-0xff*/
05381 /* 0x8900 */
05382 0xd177, 0xd178, 0xb9d3, 0xd179, 0xd17a, 0xd17b, 0xd17c, 0xd17d, /*0x00-0x07*/
05383 0xd17e, 0xd180, 0xf1db, 0xd181, 0xd182, 0xd183, 0xd184, 0xd185, /*0x08-0x0f*/
05384 0xbad6, 0xd186, 0xb0fd, 0xf1d9, 0xd187, 0xd188, 0xd189, 0xd18a, /*0x10-0x17*/
05385 0xd18b, 0xf1d8, 0xf1d2, 0xf1da, 0xd18c, 0xd18d, 0xd18e, 0xd18f, /*0x18-0x1f*/
05386 0xd190, 0xf1d7, 0xd191, 0xd192, 0xd193, 0xc8ec, 0xd194, 0xd195, /*0x20-0x27*/
05387 0xd196, 0xd197, 0xcdca, 0xf1dd, 0xd198, 0xd199, 0xd19a, 0xd19b, /*0x28-0x2f*/
05388 0xe5bd, 0xd19c, 0xd19d, 0xd19e, 0xf1dc, 0xd19f, 0xf1de, 0xd1a0, /*0x30-0x37*/
05389 0xd240, 0xd241, 0xd242, 0xd243, 0xd244, 0xd245, 0xd246, 0xd247, /*0x38-0x3f*/
05390 0xd248, 0xf1df, 0xd249, 0xd24a, 0xcfe5, 0xd24b, 0xd24c, 0xd24d, /*0x40-0x47*/
05391 0xd24e, 0xd24f, 0xd250, 0xd251, 0xd252, 0xd253, 0xd254, 0xd255, /*0x48-0x4f*/
05392 0xd256, 0xd257, 0xd258, 0xd259, 0xd25a, 0xd25b, 0xd25c, 0xd25d, /*0x50-0x57*/
05393 0xd25e, 0xd25f, 0xd260, 0xd261, 0xd262, 0xd263, 0xf4c5, 0xbdf3, /*0x58-0x5f*/
05394 0xd264, 0xd265, 0xd266, 0xd267, 0xd268, 0xd269, 0xf1e0, 0xd26a, /*0x60-0x67*/
05395 0xd26b, 0xd26c, 0xd26d, 0xd26e, 0xd26f, 0xd270, 0xd271, 0xd272, /*0x68-0x6f*/
05396 0xd273, 0xd274, 0xd275, 0xd276, 0xd277, 0xd278, 0xd279, 0xd27a, /*0x70-0x77*/
05397 0xd27b, 0xd27c, 0xd27d, 0xf1e1, 0xd27e, 0xd280, 0xd281, 0xcef7, /*0x78-0x7f*/
05398 0xd282, 0xd2aa, 0xd283, 0xf1fb, 0xd284, 0xd285, 0xb8b2, 0xd286, /*0x80-0x87*/
05399 0xd287, 0xd288, 0xd289, 0xd28a, 0xd28b, 0xd28c, 0xd28d, 0xd28e, /*0x88-0x8f*/
```



```

05400 0xd28f, 0xd290, 0xd291, 0xd292, 0xd293, 0xd294, 0xd295, 0xd296, /*0x90-0x97*/
05401 0xd297, 0xd298, 0xd299, 0xd29a, 0xd29b, 0xd29c, 0xd29d, 0xd29e, /*0x98-0x9f*/
05402 0xd29f, 0xd2a0, 0xd2a1, 0xd2a2, 0xd2a3, 0xd2a4, 0xd2a5, 0xd2a6, /*0xa0-0xa7*/
05403 0xd2a7, 0xd2a8, 0xd2a9, 0xd2aa, 0xd2ab, 0xd2ac, 0xd2ad, 0xd2ae, /*0xa8-0xaf*/
05404 0xd2af, 0xd2b0, 0xd2b1, 0xd2b2, 0xd2b3, 0xd2b4, 0xd2b5, 0xd2b6, /*0xb0-0xb7*/
05405 0xd2b7, 0xd2b8, 0xd2b9, 0xd2ba, 0xd2bb, 0xd2bc, 0xd2bd, 0xd2be, /*0xb8-0xbf*/
05406 0xd2bf, 0xd2c0, 0xd2c1, 0xd2c2, 0xd2c3, 0xd2c4, 0xd2c5, 0xd2c6, /*0xc0-0xc7*/
05407 0xd2c7, 0xd2c8, 0xd2c9, 0xd2ca, 0xd2cb, 0xd2cc, 0xd2cd, 0xd2ce, /*0xc8-0xcf*/
05408 0xd2cf, 0xd2d0, 0xd2d1, 0xd2d2, 0xd2d3, 0xd2d4, 0xd2d5, 0xd2d6, /*0xd0-0xd7*/
05409 0xd2d7, 0xd2d8, 0xd2d9, 0xd2da, 0xd2db, 0xd2dc, 0xd2dd, 0xd2de, /*0xd8-0xdf*/
05410 0xd2df, 0xd2e0, 0xd2e1, 0xd2e2, 0xd2e3, 0xd2e4, 0xd2e5, 0xd2e6, /*0xe0-0xe7*/
05411 0xd2e7, 0xd2e8, 0xd2e9, 0xd2ea, 0xd2eb, 0xd2ec, 0xd2ed, 0xd2ee, /*0xe8-0xef*/
05412 0xd2ef, 0xd2f0, 0xd2f1, 0xd2f2, 0xd2f3, 0xd2f4, 0xd2f5, 0xd2f6, /*0xf0-0xf7*/
05413 0xd2f7, 0xd2f8, 0xd2f9, 0xd2fa, 0xd2fb, 0xd2fc, 0xd2fd, 0xd2fe, /*0xf8-0xff*/
05414 /* 0x8a00 */
05415 0xd1d4, 0xd385, 0xd386, 0xd387, 0xd388, 0xd389, 0xd38a, 0xd9ea, /*0x00-0x07*/
05416 0xd38b, 0xd38c, 0xd38d, 0xd38e, 0xd38f, 0xd390, 0xd391, 0xd392, /*0x08-0x0f*/
05417 0xd393, 0xd394, 0xd395, 0xd396, 0xd397, 0xd398, 0xd399, 0xd39a, /*0x10-0x17*/
05418 0xd39b, 0xd39c, 0xd39d, 0xd39e, 0xd39f, 0xd3a0, 0xd440, 0xd441, /*0x18-0x1f*/
05419 0xd442, 0xd443, 0xd444, 0xd445, 0xd446, 0xd447, 0xd448, 0xd449, /*0x20-0x27*/
05420 0xd44a, 0xd44b, 0xd44c, 0xd44d, 0xd44e, 0xd44f, 0xd450, 0xd451, /*0x28-0x2f*/
05421 0xd452, 0xd453, 0xd454, 0xd455, 0xd456, 0xd457, 0xd458, 0xd459, /*0x30-0x37*/
05422 0xd45a, 0xd45b, 0xd45c, 0xd45d, 0xd45e, 0xd45f, 0xf6a4, 0xd460, /*0x38-0x3f*/
05423 0xd461, 0xd462, 0xd463, 0xd464, 0xd465, 0xd466, 0xd467, 0xd468, /*0x40-0x47*/
05424 0xd469, 0xd46a, 0xd46b, 0xd46c, 0xd46d, 0xd46e, 0xd46f, 0xd470, /*0x48-0x4f*/
05425 0xd471, 0xd472, 0xd473, 0xd474, 0xd475, 0xd476, 0xd477, 0xd478, /*0x50-0x57*/
05426 0xd479, 0xd47a, 0xd47b, 0xd47c, 0xd47d, 0xd47e, 0xd47f, 0xd480, /*0x58-0x5f*/
05427 0xd481, 0xd482, 0xd483, 0xd484, 0xd485, 0xd486, 0xd487, 0xd488, /*0x60-0x67*/
05428 0xd489, 0xd48a, 0xd48b, 0xd48c, 0xd48d, 0xd48e, 0xd48f, 0xd490, /*0x68-0x6f*/
05429 0xd491, 0xd492, 0xd493, 0xd494, 0xd495, 0xd496, 0xd497, 0xd498, /*0x70-0x77*/
05430 0xd499, 0xd5b2, 0xd49a, 0xd49b, 0xd49c, 0xd49d, 0xd49e, 0xd49f, /*0x78-0x7f*/
05431 0xd4a0, 0xd540, 0xd541, 0xd542, 0xd543, 0xd544, 0xd545, 0xd546, /*0x80-0x87*/
05432 0xd547, 0xd548, 0xd549, 0xd54a, 0xd54b, 0xd54c, 0xd54d, 0xd54e, /*0x88-0x8f*/
05433 0xd54f, 0xd550, 0xd551, 0xd552, 0xd553, 0xd554, 0xd555, 0xd556, /*0x90-0x97*/
05434 0xd557, 0xd558, 0xd559, 0xd55a, 0xd55b, 0xd55c, 0xd55d, 0xd55e, /*0x98-0x9f*/
05435 0xd55f, 0xd560, 0xd561, 0xd562, 0xd563, 0xd564, 0xd565, 0xd566, /*0xa0-0xaf*/
05436 0xd567, 0xd568, 0xd569, 0xd56a, 0xd56b, 0xd56c, 0xd56d, 0xd56e, /*0xb0-0xbf*/
05437 0xd56f, 0xd570, 0xd571, 0xd572, 0xd573, 0xd574, 0xd575, 0xd576, /*0xc0-0xcf*/
05438 0xd577, 0xd578, 0xd579, 0xd57a, 0xd57b, 0xd57c, 0xd57d, 0xd57e, /*0xd0-0xdf*/
05439 0xd57f, 0xd580, 0xd581, 0xd582, 0xd583, 0xd584, 0xd585, 0xd586, /*0xe0-0xef*/
05440 0xd587, 0xd588, 0xd589, 0xd58a, 0xd58b, 0xd58c, 0xd58d, 0xd58e, /*0xf0-0xf7*/
05441 0xd58f, 0xd590, 0xd591, 0xd592, 0xd593, 0xd594, 0xd595, 0xd596, /*0xf8-0xff*/
05442 0xd597, 0xd598, 0xd599, 0xd59a, 0xd59b, 0xd59c, 0xd59d, 0xd59e, /*0x00-0x07*/
05443 0xd59f, 0xd5a0, 0xd5a1, 0xd5a2, 0xd5a3, 0xd5a4, 0xd5a5, 0xd5a6, /*0x08-0x0f*/
05444 0xd5a7, 0xd5a8, 0xd5a9, 0xd5aa, 0xd5ab, 0xd5ac, 0xd5ad, 0xd5ae, /*0x10-0x17*/
05445 0xd5af, 0xd5b0, 0xd5b1, 0xd5b2, 0xd5b3, 0xd5b4, 0xd5b5, 0xd5b6, /*0x18-0x1f*/
05446 0xd5b7, 0xd5b8, 0xd5b9, 0xd5ba, 0xd5bb, 0xd5bc, 0xd5bd, 0xd5be, /*0x20-0x27*/
05447 /* 0x8b00 */
05448 0xd65c, 0xd65d, 0xd65e, 0xd65f, 0xd660, 0xd661, 0xd662, 0xe5c0, /*0x00-0x07*/
05449 0xd663, 0xd664, 0xd665, 0xd666, 0xd667, 0xd668, 0xd669, 0xd66a, /*0x08-0x0f*/
05450 0xd66b, 0xd66c, 0xd66d, 0xd66e, 0xd66f, 0xd670, 0xd671, 0xd672, /*0x10-0x17*/
05451 0xd673, 0xd674, 0xd675, 0xd676, 0xd677, 0xd678, 0xd679, 0xd67a, /*0x18-0x1f*/
05452 0xd67b, 0xd67c, 0xd67d, 0xd67e, 0xd67f, 0xd680, 0xd681, 0xf6a5, 0xd682, /*0x20-0x27*/
05453 0xd683, 0xd684, 0xd685, 0xd686, 0xd687, 0xd688, 0xd689, 0xd68a, /*0x28-0x2f*/
05454 0xd68b, 0xd68c, 0xd68d, 0xd68e, 0xd68f, 0xd690, 0xd691, 0xd692, /*0x30-0x37*/
05455 0xd693, 0xd694, 0xd695, 0xd696, 0xd697, 0xd698, 0xd699, 0xd69a, /*0x38-0x3f*/
05456 0xd69b, 0xd69c, 0xd69d, 0xd69e, 0xd69f, 0xd6a0, 0xd740, 0xd741, /*0x40-0x47*/
05457 0xd742, 0xd743, 0xd744, 0xd745, 0xd746, 0xd747, 0xd748, 0xd749, /*0x48-0x4f*/
05458 0xd74a, 0xd74b, 0xd74c, 0xd74d, 0xd74e, 0xd74f, 0xd750, 0xd751, /*0x50-0x57*/
05459 0xd752, 0xd753, 0xd754, 0xd755, 0xd756, 0xd757, 0xd758, 0xd759, /*0x58-0x5f*/
05460 0xd75a, 0xd75b, 0xd75c, 0xd75d, 0xd75e, 0xd75f, 0xb6a4, 0xd760, /*0x60-0x67*/
05461 0xd761, 0xd762, 0xd763, 0xd764, 0xc6a9, 0xd765, 0xd766, 0xd767, /*0x68-0x6f*/
05462 0xd768, 0xd769, 0xd76a, 0xd76b, 0xd76c, 0xd76d, 0xd76e, 0xd76f, /*0x70-0x77*/
05463 0xd770, 0xd771, 0xd772, 0xd773, 0xd774, 0xd775, 0xd776, 0xd777, /*0x78-0x7f*/
05464 0xd778, 0xd779, 0xd77a, 0xd77b, 0xd77c, 0xd77d, 0xd77e, 0xd77f, /*0x80-0x87*/
05465 0xd780, 0xd781, 0xd782, 0xd783, 0xd784, 0xd785, 0xd786, 0xd787, /*0x88-0x8f*/
05466 0xd788, 0xd789, 0xd78a, 0xd78b, 0xd78c, 0xd78d, 0xd78e, 0xd78f, /*0x90-0x97*/
05467 0xd790, 0xd791, 0xd792, 0xd793, 0xd794, 0xd795, 0xd796, 0xd797, /*0x98-0x9f*/
05468 0xd798, 0xd799, 0xd79a, 0xd79b, 0xd79c, 0xd79d, 0xd79e, 0xd79f, /*0xa0-0xaf*/
05469 0xd800, 0xd801, 0xd802, 0xd803, 0xd804, 0xd805, 0xd806, 0xd807, /*0xb0-0xbf*/
05470 0xd808, 0xd809, 0xd80a, 0xd80b, 0xd80c, 0xd80d, 0xd80e, 0xd80f, /*0xc0-0xcf*/
05471 0xd810, 0xd811, 0xd812, 0xd813, 0xd814, 0xd815, 0xd816, 0xd817, /*0xd0-0xdf*/
05472 0xd818, 0xd819, 0xd81a, 0xd81b, 0xd81c, 0xd81d, 0xd81e, 0xd81f, /*0xe0-0xef*/
05473 0xd820, 0xd821, 0xd822, 0xd823, 0xd824, 0xd825, 0xd826, 0xd827, /*0xf0-0xf7*/
05474 0xd828, 0xd829, 0xd82a, 0xd82b, 0xd82c, 0xd82d, 0xd82e, 0xd82f, /*0xf8-0xff*/
05475 0xd830, 0xd831, 0xd832, 0xd833, 0xd834, 0xd835, 0xd836, 0xd837, /*0x00-0x07*/
05476 0xd838, 0xd839, 0xd83a, 0xd83b, 0xd83c, 0xd83d, 0xd83e, 0xd83f, /*0x08-0x0f*/
05477 0xd840, 0xd841, 0xd842, 0xd843, 0xd844, 0xd845, 0xd846, 0xd847, /*0x10-0x17*/
05478 0xd848, 0xd849, 0xd84a, 0xd84b, 0xd84c, 0xd84d, 0xd84e, 0xd84f, /*0x18-0x1f*/
05479 0xd850, 0xd851, 0xd852, 0xd853, 0xd854, 0xd855, 0xd856, 0xd857, /*0x20-0x27*/
05480 0xd858, 0xd859, 0xd85a, 0xd85b, 0xd85c, 0xd85d, 0xd85e, 0xd85f, /*0x28-0x2f*/
05481 0xd860, 0xd861, 0xd862, 0xd863, 0xd864, 0xd865, 0xd866, 0xd867, /*0x30-0x37*/
05482 0xd868, 0xd869, 0xd86a, 0xd86b, 0xd86c, 0xd86d, 0xd86e, 0xd86f, /*0x38-0x3f*/
05483 0xd870, 0xd871, 0xd872, 0xd873, 0xd874, 0xd875, 0xd876, 0xd877, /*0x40-0x47*/
05484 0xd878, 0xd879, 0xd87a, 0xd87b, 0xd87c, 0xd87d, 0xd87e, 0xd87f, /*0x48-0x4f*/
05485 0xd880, 0xd881, 0xd882, 0xd883, 0xd884, 0xd885, 0xd886, 0xd887, /*0x50-0x57*/
05486 0xd888, 0xd889, 0xd88a, 0xd88b, 0xd88c, 0xd88d, 0xd88e, 0xd88f, /*0x58-0x5f*/

```

```
05487 0xc0be, 0xc6d7, 0xdadc, 0xdadd, 0xc7b4, 0xdade, 0xdadf, 0xb9c8, /*x30-0x37*/
05488 0xd840, 0xd841, 0xd842, 0xd843, 0xd844, 0xd845, 0xd846, 0xd847, /*x38-0x3f*/
05489 0xd848, 0xbbed, 0xd849, 0xd84a, 0xd84b, 0xd84c, 0xb6b9, 0xf4f8, /*x40-0x47*/
05490 0xd84d, 0xf4f9, 0xd84e, 0xd84f, 0xcde3, 0xd850, 0xd851, 0xd852, /*x48-0x4f*/
05491 0xd853, 0xd854, 0xd855, 0xd856, 0xd857, 0xf5b9, 0xd858, 0xd859, /*x50-0x57*/
05492 0xd85a, 0xd85b, 0xebe0, 0xd85c, 0xd85d, 0xd85e, 0xd85f, 0xd860, /*x58-0x5f*/
05493 0xd861, 0xcff3, 0xbbbf, 0xd862, 0xd863, 0xd864, 0xd865, 0xd866, /*x60-0x67*/
05494 0xd867, 0xd868, 0xbac0, 0xd4a5, 0xd869, 0xd86a, 0xd86b, 0xd86c, /*x68-0x6f*/
05495 0xd86d, 0xd86e, 0xd86f, 0xe1d9, 0xd870, 0xd871, 0xd872, 0xd873, /*x70-0x77*/
05496 0xf5f4, 0xb1aa, 0xb2f2, 0xd874, 0xd875, 0xd876, 0xd877, 0xd878, /*x78-0x7f*/
05497 0xd879, 0xd87a, 0xf5f5, 0xd87b, 0xd87c, 0xf5f7, 0xd87d, 0xd87e, /*x80-0x87*/
05498 0xd880, 0xbad1, 0xf5f6, 0xd881, 0xc3b2, 0xd882, 0xd883, 0xd884, /*x88-0x8f*/
05499 0xd885, 0xd886, 0xd887, 0xd888, 0xf5f9, 0xd889, 0xd88a, 0xd88b, /*x90-0x97*/
05500 0xf5f8, 0xd88c, 0xd88d, 0xd88e, 0xd88f, 0xd890, 0xd891, 0xd892, /*x98-0x9f*/
05501 0xd893, 0xd894, 0xd895, 0xd896, 0xd897, 0xd898, 0xd899, 0xd89a, /*xa0-xa7*/
05502 0xd89b, 0xd89c, 0xd89d, 0xd89e, 0xd89f, 0xd8a0, 0xd940, 0xd941, /*xa8-xaf*/
05503 0xd942, 0xd943, 0xd944, 0xd945, 0xd946, 0xd947, 0xd948, 0xd949, /*xb0-xb7*/
05504 0xd94a, 0xd94b, 0xd94c, 0xd94d, 0xd94e, 0xd94f, 0xd950, 0xd951, /*xb8-xbf*/
05505 0xd952, 0xd953, 0xd954, 0xd955, 0xd956, 0xd957, 0xd958, 0xd959, /*xc0-xc7*/
05506 0xd95a, 0xd95b, 0xd95c, 0xd95d, 0xd95e, 0xd95f, 0xd960, 0xd961, /*xc8-xcf*/
05507 0xd962, 0xd963, 0xd964, 0xd965, 0xd966, 0xd967, 0xd968, 0xd969, /*xd0-xd7*/
05508 0xd96a, 0xd96b, 0xd96c, 0xd96d, 0xd96e, 0xd96f, 0xd970, 0xd971, /*xd8-xdf*/
05509 0xd972, 0xd973, 0xd974, 0xd975, 0xd976, 0xd977, 0xd978, 0xd979, /*xe0-xe7*/
05510 0xd97a, 0xd97b, 0xd97c, 0xd97d, 0xd97e, 0xd980, 0xd981, 0xd982, /*xe8-xef*/
05511 0xd983, 0xd984, 0xd985, 0xd986, 0xd987, 0xd988, 0xd989, 0xd98a, /*xf0-xf7*/
05512 0xd98b, 0xd98c, 0xd98d, 0xd98e, 0xd98f, 0xd990, 0xd991, 0xd992, /*xf8-xf7*/
05513 /* 0x8d00 */
05514 0xd993, 0xd994, 0xd995, 0xd996, 0xd997, 0xd998, 0xd999, 0xd99a, /*x00-0x07*/
05515 0xd99b, 0xd99c, 0xd99d, 0xd99e, 0xd99f, 0xd9a0, 0xd9a1, 0xd9a2, /*x08-0x0f*/
05516 0xd9a3, 0xd9a4, 0xd9a5, 0xd9a6, 0xd9a7, 0xd9a8, 0xd9a9, 0xd9aa, /*x10-0x17*/
05517 0xd9ab, 0xd9ac, 0xd9ad, 0xd9ae, 0xd9af, 0xd9b0, 0xd9b1, 0xd9b2, /*x18-0x1f*/
05518 0xd9b3, 0xd9b4, 0xd9b5, 0xd9b6, 0xd9b7, 0xd9b8, 0xd9b9, 0xd9ba, /*x20-0x27*/
05519 0xd9bb, 0xd9bc, 0xd9bd, 0xd9be, 0xd9bf, 0xd9c0, 0xd9c1, 0xd9c2, /*x28-0x2f*/
05520 0xd9c3, 0xd9c4, 0xd9c5, 0xd9c6, 0xd9c7, 0xd9c8, 0xd9c9, 0xd9ca, /*x30-0x37*/
05521 0xd9cb, 0xd9cc, 0xd9cd, 0xd9ce, 0xd9cf, 0xd9d0, 0xd9d1, 0xd9d2, /*x38-0x3f*/
05522 0xd9d3, 0xd9d4, 0xd9d5, 0xd9d6, 0xd9d7, 0xd9d8, 0xd9d9, 0xd9da, /*x40-0x47*/
05523 0xd9db, 0xd9dc, 0xd9dd, 0xd9de, 0xd9df, 0xd9e0, 0xd9e1, 0xd9e2, /*x48-0x4f*/
05524 0xd9e3, 0xd9e4, 0xd9e5, 0xd9e6, 0xd9e7, 0xd9e8, 0xd9e9, 0xd9ea, /*x50-0x57*/
05525 0xd9eb, 0xd9ec, 0xd9ed, 0xd9ee, 0xd9ef, 0xd9f0, 0xd9f1, 0xd9f2, /*x58-0x5f*/
05526 0xd9f3, 0xd9f4, 0xd9f5, 0xd9f6, 0xd9f7, 0xd9f8, 0xd9f9, 0xd9fa, /*x60-0x67*/
05527 0xd9fb, 0xd9fc, 0xd9fd, 0xd9fe, 0xd9ff, 0xda00, 0xda01, 0xda02, /*x68-0x6f*/
05528 0xda03, 0xda04, 0xda05, 0xda06, 0xda07, 0xda08, 0xda09, 0xda0a, /*x70-0x77*/
05529 0xda0b, 0xda0c, 0xda0d, 0xda0e, 0xda0f, 0xda10, 0xda11, 0xda12, /*x78-0x7f*/
05530 0xda13, 0xda14, 0xda15, 0xda16, 0xda17, 0xda18, 0xda19, 0xda1a, /*x80-0x87*/
05531 0xda1b, 0xda1c, 0xda1d, 0xda1e, 0xda1f, 0xda20, 0xda21, 0xda22, /*x88-0x8f*/
05532 0xda23, 0xda24, 0xda25, 0xda26, 0xda27, 0xda28, 0xda29, 0xda2a, /*x90-0x97*/
05533 0xda2b, 0xda2c, 0xda2d, 0xda2e, 0xda2f, 0xda30, 0xda31, 0xda32, /*x98-0x9f*/
05534 0xda33, 0xda34, 0xda35, 0xda36, 0xda37, 0xda38, 0xda39, 0xda3a, /*xa0-xa7*/
05535 0xda3b, 0xda3c, 0xda3d, 0xda3e, 0xda3f, 0xda40, 0xda41, 0xda42, /*xa8-xaf*/
05536 0xda43, 0xda44, 0xda45, 0xda46, 0xda47, 0xda48, 0xda49, 0xda4a, /*xb0-xb7*/
05537 0xda4b, 0xda4c, 0xda4d, 0xda4e, 0xda4f, 0xda50, 0xda51, 0xda52, /*xb8-xbf*/
05538 0xda53, 0xda54, 0xda55, 0xda56, 0xda57, 0xda58, 0xda59, 0xda5a, /*xc0-xc7*/
05539 0xda5b, 0xda5c, 0xda5d, 0xda5e, 0xda5f, 0xda60, 0xda61, 0xda62, /*xc8-xcf*/
05540 0xda63, 0xda64, 0xda65, 0xda66, 0xda67, 0xda68, 0xda69, 0xda6a, /*xd0-xd7*/
05541 0xda6b, 0xda6c, 0xda6d, 0xda6e, 0xda6f, 0xda70, 0xda71, 0xda72, /*xd8-xdf*/
05542 0xda73, 0xda74, 0xda75, 0xda76, 0xda77, 0xda78, 0xda79, 0xda7a, /*xe0-xe7*/
05543 0xda7b, 0xda7c, 0xda7d, 0xda7e, 0xda7f, 0xda80, 0xda81, 0xda82, /*xe8-xef*/
05544 0xda83, 0xda84, 0xda85, 0xda86, 0xda87, 0xda88, 0xda89, 0xda8a, /*xf0-xf7*/
05545 0xda8b, 0xda8c, 0xda8d, 0xda8e, 0xda8f, 0xda90, 0xda91, 0xda92, /*xf8-xf7*/
05546 /* 0x8e00 */
05547 0xda93, 0xda94, 0xda95, 0xda96, 0xda97, 0xda98, 0xda99, 0xda9a, /*x00-0x07*/
05548 0xda9b, 0xda9c, 0xda9d, 0xda9e, 0xda9f, 0xdaa0, 0xdaa1, 0xdaa2, /*x08-0x0f*/
05549 0xdaa3, 0xdaa4, 0xdaa5, 0xdaa6, 0xdaa7, 0xdaa8, 0xdaa9, 0xdaaa, /*x10-0x17*/
05550 0xdaab, 0xdaac, 0xdaad, 0xdaae, 0xdaaf, 0xdab0, 0xdab1, 0xdab2, /*x18-0x1f*/
05551 0xdab3, 0xdab4, 0xdab5, 0xdab6, 0xdab7, 0xdab8, 0xdab9, 0xdaba, /*x20-0x27*/
05552 0xdabb, 0xdabc, 0xdabd, 0xdabe, 0xdabf, 0xdac0, 0xdac1, 0xdac2, /*x28-0x2f*/
05553 0xdac3, 0xdac4, 0xdac5, 0xdac6, 0xdac7, 0xdac8, 0xdac9, 0xdaca, /*x30-0x37*/
05554 0xdacb, 0xdacc, 0xdacd, 0xdace, 0xdacf, 0xdad0, 0xdad1, 0xdad2, /*x38-0x3f*/
05555 0xdad3, 0xdad4, 0xdad5, 0xdad6, 0xdad7, 0xdad8, 0xdad9, 0xdaea, /*x40-0x47*/
05556 0xdaeb, 0xdaec, 0xdaed, 0xdaee, 0xdaef, 0xdaf0, 0xdaf1, 0xdaf2, /*x48-0x4f*/
05557 0xdaf3, 0xdaf4, 0xdaf5, 0xdaf6, 0xdaf7, 0xdaf8, 0xdaf9, 0xdafa, /*x50-0x57*/
05558 0xdafb, 0xdafc, 0xdafd, 0xdafe, 0xdaff, 0xdb00, 0xdb01, 0xdb02, /*x58-0x5f*/
05559 0xdb03, 0xdb04, 0xdb05, 0xdb06, 0xdb07, 0xdb08, 0xdb09, 0xdb0a, /*x60-0x67*/
05560 0xdb0b, 0xdb0c, 0xdb0d, 0xdb0e, 0xdb0f, 0xdb10, 0xdb11, 0xdb12, /*x68-0x6f*/
05561 0xdb13, 0xdb14, 0xdb15, 0xdb16, 0xdb17, 0xdb18, 0xdb19, 0xdb1a, /*x70-0x77*/
05562 0xdb1b, 0xdb1c, 0xdb1d, 0xdb1e, 0xdb1f, 0xdb20, 0xdb21, 0xdb22, /*x78-0x7f*/
05563 0xdb23, 0xdb24, 0xdb25, 0xdb26, 0xdb27, 0xdb28, 0xdb29, 0xdb2a, /*x80-0x87*/
05564 0xdb2b, 0xdb2c, 0xdb2d, 0xdb2e, 0xdb2f, 0xdb30, 0xdb31, 0xdb32, /*x88-0x8f*/
05565 0xdb33, 0xdb34, 0xdb35, 0xdb36, 0xdb37, 0xdb38, 0xdb39, 0xdb3a, /*x90-0x97*/
05566 0xdb3b, 0xdb3c, 0xdb3d, 0xdb3e, 0xdb3f, 0xdb40, 0xdb41, 0xdb42, /*x98-0x9f*/
05567 0xdb43, 0xdb44, 0xdb45, 0xdb46, 0xdb47, 0xdb48, 0xdb49, 0xdb4a, /*xa0-xa7*/
05568 0xdb4b, 0xdb4c, 0xdb4d, 0xdb4e, 0xdb4f, 0xdb50, 0xdb51, 0xdb52, /*xa8-xaf*/
05569 0xdb53, 0xdb54, 0xdb55, 0xdb56, 0xdb57, 0xdb58, 0xdb59, 0xdb5a, /*xb0-xb7*/
05570 0xdb5b, 0xdb5c, 0xdb5d, 0xdb5e, 0xdb5f, 0xdb60, 0xdb61, 0xdb62, /*xb8-xbf*/
05571 0xdb63, 0xdb64, 0xdb65, 0xdb66, 0xdb67, 0xdb68, 0xdb69, 0xdb6a, /*xc0-xc7*/
05572 0xdb6b, 0xdb6c, 0xdb6d, 0xdb6e, 0xdb6f, 0xdb70, 0xdb71, 0xdb72, /*xc8-xcf*/
05573 0xdb73, 0xdb74, 0xdb75, 0xdb76, 0xdb77, 0xdb78, 0xdb79, 0xdb7a, /*xd0-xd7*/
```

```

05574 0xdc94, 0xdc95, 0xdc96, 0xdc97, 0xdc98, 0xdc99, 0xdc9a, 0xdc9b, /*0xd8-0xdf*/
05575 0xdc9c, 0xdc9d, 0xdc9e, 0xdc9f, 0xdca0, 0xdd40, 0xdd41, 0xdd42, /*0xe0-0xe7*/
05576 0xdd43, 0xdd44, 0xdd45, 0xdd46, 0xdd47, 0xdd48, 0xdd49, 0xdd4a, /*0xe8-0xef*/
05577 0xdd4b, 0xdd4c, 0xdd4d, 0xdd4e, 0xdd4f, 0xdd50, 0xdd51, 0xdd52, /*0xf0-0xf7*/
05578 0xdd53, 0xdd54, 0xdd55, 0xdd56, 0xdd57, 0xdd58, 0xdd59, 0xdd5a, /*0xf8-0xff*/
05579 /* 0x8f00 */
05580 0xdd5b, 0xdd5c, 0xdd5d, 0xdd5e, 0xdd5f, 0xdd60, 0xdd61, 0xdd62, /*0x00-0x07*/
05581 0xdd63, 0xdd64, 0xdd65, 0xdd66, 0xdd67, 0xdd68, 0xdd69, 0xdd6a, /*0x08-0x0f*/
05582 0xdd6b, 0xdd6c, 0xdd6d, 0xdd6e, 0xdd6f, 0xdd70, 0xdd71, 0xdd72, /*0x10-0x17*/
05583 0xdd73, 0xdd74, 0xdd75, 0xdd76, 0xdd77, 0xdd78, 0xdd79, 0xdd7a, /*0x18-0x1f*/
05584 0xdd7b, 0xdd7c, 0xdd7d, 0xdd7e, 0xdd80, 0xdd81, 0xdd82, 0xdd83, /*0x20-0x27*/
05585 0xdd84, 0xdd85, 0xdd86, 0xdd87, 0xdd88, 0xdd89, 0xdd8a, 0xdd8b, /*0x28-0x2f*/
05586 0xdd8c, 0xdd8d, 0xdd8e, 0xdd8f, 0xdd90, 0xdd91, 0xdd92, 0xdd93, /*0x30-0x37*/
05587 0xdd94, 0xdd95, 0xdd96, 0xdd97, 0xdd98, 0xdd99, 0xdd9a, 0xdd9b, /*0x38-0x3f*/
05588 0xdd9c, 0xdd9d, 0xdd9e, 0xdd9f, 0xdda0, 0xde40, 0xde41, 0xde42, /*0x40-0x47*/
05589 0xde43, 0xde44, 0xde45, 0xde46, 0xde47, 0xde48, 0xde49, 0xde4a, /*0x48-0x4f*/
05590 0xde4b, 0xde4c, 0xde4d, 0xde4e, 0xde4f, 0xde50, 0xde51, 0xde52, /*0x50-0x57*/
05591 0xde53, 0xde54, 0xde55, 0xde56, 0xde57, 0xde58, 0xde59, 0xde5a, /*0x58-0x5f*/
05592 0xde5b, 0xde5c, 0xde5d, 0xde5e, 0xde5f, 0xde60, 0xb3b5, 0xd4fe, /*0x60-0x67*/
05593 0xb9ec, 0xd0f9, 0xde61, 0xe9ed, 0xd7aa, 0xe9ee, 0xc2d6, 0xc8ed, /*0x68-0x6f*/
05594 0xbae4, 0xe9ef, 0xe9f0, 0xe9f1, 0xd6e1, 0xe9f2, 0xe9f3, 0xe9f5, /*0x70-0x7f*/
05595 0xe9f4, 0xe9f6, 0xe9f7, 0xc7e1, 0xe9f8, 0xd4d8, 0xe9f9, 0xbdcf, /*0x78-0x7f*/
05596 0xde62, 0xe9fa, 0xe9fb, 0xbdcf, 0xe9fc, 0xb8a8, 0xc1be, 0xe9fd, /*0x80-0x87*/
05597 0xb1b2, 0xbdbd, 0xb9f5, 0xe9fe, 0xde63, 0xeaa1, 0xeaa2, 0xeaa3, /*0x88-0x8f*/
05598 0xb7f8, 0xbcad, 0xde64, 0xcae4, 0xe0ce, 0xd4af, 0xcfbf, 0xd5b7, /*0x90-0x97*/
05599 0xeaa4, 0xd5de, 0xeaa5, 0xd0c1, 0xb9bc, 0xde65, 0xb4c7, 0xb1d9, /*0x98-0x9f*/
05600 0xde66, 0xde67, 0xde68, 0xc0b1, 0xde69, 0xde6a, 0xde6b, 0xde6c, /*0xa0-0xaf*/
05601 0xb1e6, 0xb1e7, 0xde6d, 0xb1e8, 0xde6e, 0xde6f, 0xde70, 0xde71, /*0xa8-0xaf*/
05602 0xb3bd, 0xc8e8, 0xde72, 0xde73, 0xde74, 0xde75, 0xe5c1, 0xde76, /*0xb0-0xb7*/
05603 0xde77, 0xb1df, 0xde78, 0xde79, 0xc1c9, 0xb4ef, 0xc1c9, 0xde7b, /*0xb8-0xbf*/
05604 0xde7c, 0xc7a8, 0xd3d8, 0xde7d, 0xc6f9, 0xd1b8, 0xde7e, 0xb9fd, /*0xc0-0xc7*/
05605 0xc2f5, 0xde80, 0xde81, 0xde82, 0xde83, 0xde84, 0xd3ad, 0xde85, /*0xc8-0xcf*/
05606 0xd4cb, 0xbdfc, 0xde86, 0xe5c2, 0xb7b5, 0xe5c3, 0xde87, 0xde88, /*0xd0-0xd7*/
05607 0xbbb9, 0xd5e2, 0xde89, 0xbdf8, 0xd4b6, 0xcea5, 0xc1ac, 0xb3d9, /*0xd8-0xdf*/
05608 0xde8a, 0xde8b, 0xccf6, 0xde8c, 0xe5c6, 0xe5c4, 0xe5c8, 0xde8d, /*0xe0-0xe7*/
05609 0xe5ca, 0xe5cf, 0xb5cf, 0xc6c8, 0xde8e, 0xb5fc, 0xe5c5, 0xde8f, /*0xe8-0xef*/
05610 0xcaf6, 0xde90, 0xde91, 0xe5c9, 0xde92, 0xde93, 0xde94, 0xc3d4, /*0xf0-0xf7*/
05611 0xb1c5, 0xbca3, 0xde95, 0xde96, 0xde97, 0xd7b7, 0xde98, 0xde99, /*0xf8-0xff*/
05612 /* 0x9000 */
05613 0xcdbc, 0xcbcd, 0xcaca, 0xccd3, 0xe5cc, 0xe5cb, 0xc4e6, 0xde9a, /*0x00-0x07*/
05614 0xde9b, 0xd1a1, 0xd1b7, 0xe5cd, 0xde9c, 0xe5d0, 0xde9d, 0xcdb8, /*0x08-0x0f*/
05615 0xd6f0, 0xe5cf, 0xb5dd, 0xde9e, 0xcdbf, 0xde9f, 0xe5d1, 0xb6ba, /*0x10-0x17*/
05616 0xdea0, 0xdf40, 0xcda8, 0xb9e4, 0xdf41, 0xcac5, 0xb3d1, 0xcdb9, /*0x18-0x1f*/
05617 0xd4ec, 0xe5d2, 0xb7ea, 0xdf42, 0xdf43, 0xdf44, 0xe5ce, 0xdf45, /*0x20-0x27*/
05618 0xdf46, 0xdf47, 0xdf48, 0xdf49, 0xdf4a, 0xe5d5, 0xb4fe, 0xe5d6, /*0x28-0x2f*/
05619 0xdf4b, 0xdf4c, 0xdf4d, 0xdf4e, 0xdf4f, 0xe5d3, 0xe5d4, 0xdf50, /*0x30-0x37*/
05620 0xd2dd, 0xdf51, 0xdf52, 0xc2df, 0xb1c6, 0xdf53, 0xd3e2, 0xdf54, /*0x38-0x3f*/
05621 0xdf55, 0xb6dd, 0xcbec, 0xdf56, 0xe5d7, 0xdf57, 0xdf58, 0xd3f6, /*0x40-0x47*/
05622 0xdf59, 0xdf5a, 0xdf5b, 0xdf5c, 0xdf5d, 0xb1e9, 0xdf5e, 0xb6f4, /*0x48-0x4f*/
05623 0xe5da, 0xe5d8, 0xe5d9, 0xb5c0, 0xdf5f, 0xdf60, 0xdf61, 0xd2c5, /*0x50-0x57*/
05624 0xe5dc, 0xdf62, 0xdf63, 0xe5de, 0xdf64, 0xdf65, 0xdf66, 0xdf67, /*0x58-0x5f*/
05625 0xdf68, 0xdf69, 0xe5dd, 0xc7b2, 0xdf6a, 0xd2a3, 0xdf6b, 0xdf6c, /*0x60-0x67*/
05626 0xe5db, 0xdf6d, 0xdf6e, 0xdf6f, 0xdf70, 0xd4e2, 0xd5da, 0xdf71, /*0x68-0x6f*/
05627 0xdf72, 0xdf73, 0xdf74, 0xdf75, 0xe5e0, 0xdf71, 0xdf76, 0xdf77, /*0x70-0x7f*/
05628 0xdf78, 0xdf79, 0xdf7a, 0xdf7b, 0xdf7c, 0xe5e1, 0xdf7d, 0xb1dc, /*0x78-0x7f*/
05629 0xd1fb, 0xdf7e, 0xe5e2, 0xe5e4, 0xdf80, 0xdf81, 0xdf82, 0xdf83, /*0x80-0x87*/
05630 0xe5e3, 0xdf84, 0xdf85, 0xe5e5, 0xdf86, 0xdf87, 0xdf88, 0xdf89, /*0x88-0x8f*/
05631 0xdf8a, 0xd2d8, 0xdf8b, 0xb5cb, 0xdf8c, 0xe7df, 0xdf8d, 0xdaf5, /*0x90-0x97*/
05632 0xdf8e, 0xdaf8, 0xdf8f, 0xdaf6, 0xdf90, 0xdaf7, 0xdf91, 0xdf92, /*0x98-0x9f*/
05633 0xdf93, 0xdafa, 0xd0cf, 0xc4c7, 0xdf94, 0xdf95, 0xb0ee, 0xdf96, /*0xa0-0xaf*/
05634 0xdf97, 0xdf98, 0xd0b0, 0xdf99, 0xdaf9, 0xdf9a, 0xd3ca, 0xbaaa, /*0xa8-0xaf*/
05635 0xdba2, 0xc7f1, 0xdf9b, 0xdafc, 0xdafb, 0xc9db, 0xdafd, 0xdf9c, /*0xb0-0xb7*/
05636 0xdba1, 0xdfde, 0xdafe, 0xc1da, 0xdf9d, 0xdf9e, 0xdba5, 0xdf9f, /*0xb8-0xbf*/
05637 0xdfa0, 0xd3f4, 0xe040, 0xe041, 0xdba7, 0xdba4, 0xe042, 0xdba8, /*0xc0-0xc7*/
05638 0xe043, 0xe044, 0xbdbf, 0xe045, 0xe046, 0xe047, 0xe0c9, 0xdba3, /*0xc8-0xcf*/
05639 0xdba6, 0xd6a3, 0xe048, 0xdba9, 0xe049, 0xe04a, 0xe04b, 0xdbad, /*0xd0-0xd7*/
05640 0xe04c, 0xe04d, 0xe04e, 0xdbae, 0xdbac, 0xbac2, 0xe04f, 0xe050, /*0xd8-0xdf*/
05641 0xe051, 0xbfa4, 0xdbab, 0xe052, 0xe053, 0xe054, 0xdbaa, 0xd4c7, /*0xe0-0xe7*/
05642 0xb2bf, 0xe055, 0xe056, 0xdbaf, 0xe057, 0xb9f9, 0xe058, 0xdbb0, /*0xe8-0xef*/
05643 0xe059, 0xe05a, 0xe05b, 0xe05c, 0xb3bb, 0xe05d, 0xe05e, 0xe05f, /*0xf0-0xf7*/
05644 0xb5a6, 0xe060, 0xe061, 0xe062, 0xe063, 0xb6bc, 0xdbb1, 0xe064, /*0xf8-0xff*/
05645 /* 0x9100 */
05646 0xe065, 0xe066, 0xb6f5, 0xe067, 0xdbb2, 0xe068, 0xe069, 0xe06a, /*0x00-0x07*/
05647 0xe06b, 0xe06c, 0xe06d, 0xe06e, 0xe06f, 0xe070, 0xe071, 0xe072, /*0x08-0x0f*/
05648 0xe073, 0xe074, 0xe075, 0xe076, 0xe077, 0xe078, 0xe079, 0xe07a, /*0x10-0x17*/
05649 0xe07b, 0xb1c9, 0xe07c, 0xe07d, 0xe07e, 0xe080, 0xdbb4, 0xe081, /*0x18-0x1f*/
05650 0xe082, 0xe083, 0xdbb3, 0xdbb5, 0xe084, 0xe085, 0xe086, 0xdbb6, /*0x20-0x27*/
05651 0xe088, 0xe089, 0xe08a, 0xe08b, 0xe08c, 0xe08d, 0xe08e, 0xdbb7, /*0x28-0x2f*/
05652 0xe08f, 0xdbb8, 0xe090, 0xe091, 0xe092, 0xe093, 0xe094, 0xe095, /*0x30-0x37*/
05653 0xe096, 0xdbb8, 0xe097, 0xe098, 0xe099, 0xe09a, 0xe09b, 0xe09c, /*0x38-0x3f*/
05654 0xe09d, 0xe09e, 0xe09f, 0xdbb9, 0xe0a0, 0xe140, 0xdbba, 0xe141, /*0x40-0x47*/
05655 0xe142, 0xd3cf, 0xf4fa, 0xc7f5, 0xd7c3, 0xc5e4, 0xf4fc, 0xf4fd, /*0x48-0x4f*/
05656 0xf4fb, 0xe143, 0xbec6, 0xe144, 0xe145, 0xe146, 0xe147, 0xd0ef, /*0x50-0x57*/
05657 0xe148, 0xe149, 0xb7d3, 0xe14a, 0xe14b, 0xd4cd, 0xccaa, 0xe14c, /*0x58-0x5f*/
05658 0xe14d, 0xf5a2, 0xf5a1, 0xbaa8, 0xf4fe, 0xcdb6, 0xe14e, 0xe14f, /*0x60-0x67*/
05659 0xe150, 0xf5a4, 0xc0d2, 0xe151, 0xb3ea, 0xe152, 0xcdaa, 0xf5a5, /*0x68-0x6f*/
05660 0xf5a3, 0xbdb4, 0xf5a8, 0xe153, 0xf5a9, 0xbdcf, 0xc3b8, 0xbfe1, /*0x70-0x77*/

```



```
05661 0xcbel, 0xf5aa, 0xe154, 0xe155, 0xe156, 0xf5a6, 0xf5a7, 0xc4f0, /*0x78-0x7f*/
05662 0xe157, 0xe158, 0xe159, 0xe15a, 0xe15b, 0xf5ac, 0xe15c, 0xb4bc, /*0x80-0x87*/
05663 0xe15d, 0xd7ed, 0xe15e, 0xb4d7, 0xf5ab, 0xf5ae, 0xe15f, 0xe160, /*0x88-0x8f*/
05664 0xf5ad, 0xf5af, 0xd0d1, 0xe161, 0xe162, 0xe163, 0xe164, 0xe165, /*0x90-0x97*/
05665 0xe166, 0xe167, 0xc3d1, 0xc8a9, 0xe168, 0xe169, 0xe16a, 0xe16b, /*0x98-0x9f*/
05666 0xe16c, 0xe16d, 0xf5b0, 0xf5b1, 0xe16e, 0xe16f, 0xe170, 0xe171, /*0xa0-0xa7*/
05667 0xe172, 0xe173, 0xf5b2, 0xe174, 0xe175, 0xf5b3, 0xf5b4, 0xf5b5, /*0xa8-0xaf*/
05668 0xe176, 0xe177, 0xe178, 0xe179, 0xf5b7, 0xf5b6, 0xe17a, 0xe17b, /*0xb0-0xb7*/
05669 0xe17c, 0xe17d, 0xf5b8, 0xe17e, 0xe180, 0xe181, 0xe182, 0xe183, /*0xb8-0xbf*/
05670 0xe184, 0xe185, 0xe186, 0xe187, 0xe188, 0xe189, 0xe18a, 0xb2c9, /*0xc0-0xc7*/
05671 0xe18b, 0xd3d4, 0xcacd, 0xe18c, 0xc0ef, 0xd6d8, 0xd2b0, 0xc1bf, /*0xc8-0xcf*/
05672 0xe18d, 0xbdf0, 0xe18e, 0xe18f, 0xe190, 0xe191, 0xe192, 0xe193, /*0xd0-0xd7*/
05673 0xe194, 0xe195, 0xe196, 0xe197, 0xb8aa, 0xe198, 0xe199, 0xe19a, /*0xd8-0xdf*/
05674 0xe19b, 0xe19c, 0xe19d, 0xe19e, 0xe19f, 0xe1a0, 0xe240, 0xe241, /*0xe0-0xe7*/
05675 0xe242, 0xe243, 0xe244, 0xe245, 0xe246, 0xe247, 0xe248, 0xe249, /*0xe8-0xef*/
05676 0xe24a, 0xe24b, 0xe24c, 0xe24d, 0xe24e, 0xe24f, 0xe250, 0xe251, /*0xf0-0xf7*/
05677 0xe252, 0xe253, 0xe254, 0xe255, 0xe256, 0xe257, 0xe258, 0xe259, /*0xf8-0xff*/
05678 /* 0x9200 */
05679 0xe25a, 0xe25b, 0xe25c, 0xe25d, 0xe25e, 0xe25f, 0xe260, 0xe261, /*0x00-0x07*/
05680 0xe262, 0xe263, 0xe264, 0xe265, 0xe266, 0xe267, 0xe268, 0xe269, /*0x08-0x0f*/
05681 0xe26a, 0xe26b, 0xe26c, 0xe26d, 0xe26e, 0xe26f, 0xe270, 0xe271, /*0x10-0x17*/
05682 0xe272, 0xe273, 0xe274, 0xe275, 0xe276, 0xe277, 0xe278, 0xe279, /*0x18-0x1f*/
05683 0xe27a, 0xe27b, 0xe27c, 0xe27d, 0xe27e, 0xe280, 0xe281, 0xe282, /*0x20-0x27*/
05684 0xe283, 0xe284, 0xe285, 0xe286, 0xe287, 0xe288, 0xe289, 0xe28a, /*0x28-0x2f*/
05685 0xe28b, 0xe28c, 0xe28d, 0xe28e, 0xe28f, 0xe290, 0xe291, 0xe292, /*0x30-0x37*/
05686 0xe293, 0xe294, 0xe295, 0xe296, 0xe297, 0xe298, 0xe299, 0xe29a, /*0x38-0x3f*/
05687 0xe29b, 0xe29c, 0xe29d, 0xe29e, 0xe29f, 0xe2a0, 0xe340, 0xe341, /*0x40-0x47*/
05688 0xe342, 0xe343, 0xe344, 0xe345, 0xe346, 0xe347, 0xe348, 0xe349, /*0x48-0x4f*/
05689 0xe34a, 0xe34b, 0xe34c, 0xe34d, 0xe34e, 0xe34f, 0xe350, 0xe351, /*0x50-0x57*/
05690 0xe352, 0xe353, 0xe354, 0xe355, 0xe356, 0xe357, 0xe358, 0xe359, /*0x58-0x5f*/
05691 0xe35a, 0xe35b, 0xe35c, 0xe35d, 0xe35e, 0xe35f, 0xe360, 0xe361, /*0x60-0x67*/
05692 0xe362, 0xe363, 0xe364, 0xe365, 0xe366, 0xe367, 0xe368, 0xe369, /*0x68-0x6f*/
05693 0xe36a, 0xe36b, 0xe36c, 0xe36d, 0xbcf8, 0xe36e, 0xe36f, 0xe370, /*0x70-0x77*/
05694 0xe371, 0xe372, 0xe373, 0xe374, 0xe375, 0xe376, 0xe377, 0xe378, /*0x78-0x7f*/
05695 0xe379, 0xe37a, 0xe37b, 0xe37c, 0xe37d, 0xe37e, 0xe380, 0xe381, /*0x80-0x87*/
05696 0xe382, 0xe383, 0xe384, 0xe385, 0xe386, 0xe387, 0xf6c6, 0xe388, /*0x88-0x8f*/
05697 0xe389, 0xe38a, 0xe38b, 0xe38c, 0xe38d, 0xe38e, 0xe38f, 0xe390, /*0x90-0x97*/
05698 0xe391, 0xe392, 0xe393, 0xe394, 0xe395, 0xe396, 0xe397, 0xe398, /*0x98-0x9f*/
05699 0xe399, 0xe39a, 0xe39b, 0xe39c, 0xe39d, 0xe39e, 0xe39f, 0xe3a0, /*0xa0-0xa7*/
05700 0xe440, 0xe441, 0xe442, 0xe443, 0xe444, 0xe445, 0xf6c7, 0xe446, /*0xa8-0xaf*/
05701 0xe447, 0xe448, 0xe449, 0xe44a, 0xe44b, 0xe44c, 0xe44d, 0xe44e, /*0xb0-0xb7*/
05702 0xe44f, 0xe450, 0xe451, 0xe452, 0xe453, 0xe454, 0xe455, 0xe456, /*0xb8-0xbf*/
05703 0xe457, 0xe458, 0xe459, 0xe45a, 0xe45b, 0xe45c, 0xe45d, 0xe45e, /*0xc0-0xc7*/
05704 0xf6c8, 0xe45f, 0xe460, 0xe461, 0xe462, 0xe463, 0xe464, 0xe465, /*0xc8-0xcf*/
05705 0xe466, 0xe467, 0xe468, 0xe469, 0xe46a, 0xe46b, 0xe46c, 0xe46d, /*0xd0-0xd7*/
05706 0xe46e, 0xe46f, 0xe470, 0xe471, 0xe472, 0xe473, 0xe474, 0xe475, /*0xd8-0xdf*/
05707 0xe476, 0xe477, 0xe478, 0xe479, 0xe47a, 0xe47b, 0xe47c, 0xe47d, /*0xe0-0xe7*/
05708 0xe47e, 0xe480, 0xe481, 0xe482, 0xe483, 0xe484, 0xe485, 0xe486, /*0xe8-0xef*/
05709 0xe487, 0xe488, 0xe489, 0xe48a, 0xe48b, 0xe48c, 0xe48d, 0xe48e, /*0xf0-0xf7*/
05710 0xe48f, 0xe490, 0xe491, 0xe492, 0xe493, 0xe494, 0xe495, 0xe496, /*0xf8-0xff*/
05711 /* 0x9300 */
05712 0xe497, 0xe498, 0xe499, 0xe49a, 0xe49b, 0xe49c, 0xe49d, 0xe49e, /*0x00-0x07*/
05713 0xe49f, 0xe4a0, 0xe4a1, 0xe4a2, 0xe4a3, 0xe4a4, 0xe4a5, 0xe4a6, /*0x08-0x0f*/
05714 0xe4a7, 0xe4a8, 0xe4a9, 0xe4aa, 0xe4ab, 0xe4ac, 0xe4ad, 0xe4ae, /*0x10-0x17*/
05715 0xe4af, 0xe4b0, 0xe4b1, 0xe4b2, 0xe4b3, 0xe4b4, 0xe4b5, 0xe4b6, /*0x18-0x1f*/
05716 0xe4b7, 0xe4b8, 0xe4b9, 0xe4ba, 0xe4bb, 0xe4bc, 0xe4bd, 0xe4be, /*0x20-0x27*/
05717 0xe4bf, 0xe4c0, 0xe4c1, 0xe4c2, 0xe4c3, 0xe4c4, 0xe4c5, 0xe4c6, /*0x28-0x2f*/
05718 0xe4c7, 0xe4c8, 0xe4c9, 0xe4ca, 0xe4cb, 0xe4cc, 0xe4cd, 0xe4ce, /*0x30-0x37*/
05719 0xe4cf, 0xe4d0, 0xe4d1, 0xe4d2, 0xe4d3, 0xf6c9, 0xe4d4, 0xe4d5, /*0x38-0x3f*/
05720 0xe4d6, 0xe4d7, 0xe4d8, 0xe4d9, 0xe4da, 0xe4db, 0xe4dc, 0xe4dd, /*0x40-0x47*/
05721 0xe4de, 0xe4df, 0xe4e0, 0xe4e1, 0xe4e2, 0xe4e3, 0xe4e4, 0xe4e5, /*0x48-0x4f*/
05722 0xe4e6, 0xe4e7, 0xe4e8, 0xe4e9, 0xe4ea, 0xe4eb, 0xe4ec, 0xe4ed, /*0x50-0x57*/
05723 0xe4ee, 0xe4ef, 0xe4f0, 0xe4f1, 0xe4f2, 0xe4f3, 0xe4f4, 0xe4f5, /*0x58-0x5f*/
05724 0xe4f6, 0xe4f7, 0xe4f8, 0xe4f9, 0xe4fa, 0xe4fb, 0xe4fc, 0xe4fd, /*0x60-0x67*/
05725 0xe4fe, 0xe4ff, 0xf6ca, 0xe5a0, 0xe5a1, 0xe5a2, 0xe5a3, 0xe5a4, 0xe5a5, /*0x68-0x6f*/
05726 0xe5a6, 0xe5a7, 0xe5a8, 0xe5a9, 0xe5aa, 0xe5ab, 0xe5ac, 0xe5ad, /*0x70-0x77*/
05727 0xe5ae, 0xe5af, 0xe5b0, 0xe5b1, 0xe5b2, 0xe5b3, 0xe5b4, 0xe5b5, /*0x78-0x7f*/
05728 0xe5b6, 0xe5b7, 0xe5b8, 0xe5b9, 0xe5ba, 0xe5bb, 0xe5bc, 0xe5bd, /*0x80-0x87*/
05729 0xe5be, 0xe5bf, 0xe5c0, 0xe5c1, 0xe5c2, 0xe5c3, 0xe5c4, 0xf6cc, /*0x88-0x8f*/
05730 0xe5cd, 0xe5ce, 0xe5cf, 0xe5d0, 0xe5d1, 0xe5d2, 0xe5d3, 0xe5d4, /*0x90-0x97*/
05731 0xe5d5, 0xe5d6, 0xe5d7, 0xe5d8, 0xe5d9, 0xe5da, 0xe5db, 0xe5dc, /*0x98-0x9f*/
05732 0xe5de, 0xe5df, 0xe5e0, 0xe5e1, 0xe5e2, 0xe5e3, 0xe5e4, 0xe5e5, /*0xa0-0xa7*/
05733 0xe5e6, 0xe5e7, 0xe5e8, 0xe5e9, 0xe5ea, 0xe5eb, 0xe5ec, 0xe5ed, /*0xa8-0xaf*/
05734 0xe5ee, 0xe5ef, 0xe5f0, 0xe5f1, 0xe5f2, 0xe5f3, 0xe5f4, 0xe5f5, /*0xb0-0xb7*/
05735 0xe5f6, 0xe5f7, 0xe5f8, 0xe5f9, 0xe5fa, 0xe5fb, 0xe5fc, 0xe5fd, /*0xb8-0xbf*/
05736 0xe5fe, 0xe5ff, 0xe600, 0xe601, 0xe602, 0xe603, 0xe604, 0xe605, /*0xc0-0xc7*/
05737 0xe606, 0xe607, 0xf6cb, 0xe609, 0xe60a, 0xe60b, 0xe60c, 0xe60d, /*0xc8-0xcf*/
05738 0xe60e, 0xe60f, 0xe610, 0xe611, 0xe612, 0xe613, 0xf7e9, 0xe614, /*0xd0-0xd7*/
05739 0xe615, 0xe616, 0xe617, 0xe618, 0xe619, 0xe61a, 0xe61b, 0xe61c, /*0xd8-0xdf*/
05740 0xe61d, 0xe61e, 0xe61f, 0xe620, 0xe621, 0xe622, 0xe623, 0xe624, /*0xe0-0xe7*/
05741 0xe625, 0xe626, 0xe627, 0xe628, 0xe629, 0xe62a, 0xe62b, 0xe62c, /*0xe8-0xef*/
05742 0xe62d, 0xe62e, 0xe62f, 0xe630, 0xe631, 0xe632, 0xe633, 0xe634, /*0xf0-0xf7*/
05743 0xe635, 0xe636, 0xe637, 0xe638, 0xe639, 0xe63a, 0xe63b, 0xe63c, /*0xf8-0xff*/
05744 /* 0x9400 */
05745 0xe63d, 0xe63e, 0xe63f, 0xe640, 0xe641, 0xe642, 0xe643, 0xe644, /*0x00-0x07*/
05746 0xe645, 0xe646, 0xe647, 0xe648, 0xe649, 0xe64a, 0xe64b, 0xe64c, /*0x08-0x0f*/
05747 0xe64d, 0xe64e, 0xe64f, 0xe650, 0xe651, 0xe652, 0xe653, 0xe654, /*0x10-0x17*/
```

```

05748 0xe78a, 0xe78b, 0xe78c, 0xe78d, 0xe78e, 0xe78f, 0xe790, 0xe791, /*0x18-0x1f*/
05749 0xe792, 0xe793, 0xe794, 0xe795, 0xe796, 0xe797, 0xe798, 0xe799, /*0x20-0x27*/
05750 0xe79a, 0xe79b, 0xe79c, 0xe79d, 0xe79e, 0xe79f, 0xe7a0, 0xe840, /*0x28-0x2f*/
05751 0xe841, 0xe842, 0xe843, 0xe844, 0xe845, 0xe846, 0xe847, 0xe848, /*0x30-0x37*/
05752 0xe849, 0xe84a, 0xe84b, 0xe84c, 0xe84d, 0xe84e, 0xf6cd, 0xe84f, /*0x38-0x3f*/
05753 0xe850, 0xe851, 0xe852, 0xe853, 0xe854, 0xe855, 0xe856, 0xe857, /*0x40-0x47*/
05754 0xe858, 0xe859, 0xe85a, 0xe85b, 0xe85c, 0xe85d, 0xe85e, 0xe85f, /*0x48-0x4f*/
05755 0xe860, 0xe861, 0xe862, 0xe863, 0xe864, 0xe865, 0xe866, 0xe867, /*0x50-0x57*/
05756 0xe868, 0xe869, 0xe86a, 0xe86b, 0xe86c, 0xe86d, 0xe86e, 0xe86f, /*0x58-0x5f*/
05757 0xe870, 0xe871, 0xe872, 0xe873, 0xe874, 0xe875, 0xe876, 0xe877, /*0x60-0x67*/
05758 0xe878, 0xe879, 0xe87a, 0xf6ce, 0xe87b, 0xe87c, 0xe87d, 0xe87e, /*0x68-0x6f*/
05759 0xe880, 0xe881, 0xe882, 0xe883, 0xe884, 0xe885, 0xe886, 0xe887, /*0x70-0x77*/
05760 0xe888, 0xe889, 0xe88a, 0xe88b, 0xe88c, 0xe88d, 0xe88e, 0xe88f, /*0x78-0x7f*/
05761 0xe890, 0xe891, 0xe892, 0xe893, 0xe894, 0xeec4, 0xeec5, 0xeec6, /*0x80-0x87*/
05762 0xd5eb, 0xb6a4, 0xeec8, 0xeec7, 0xeec9, 0xeeca, 0xc7a5, 0xeecb, /*0x88-0x8f*/
05763 0xeec4, 0xe895, 0xb7b0, 0xb5f6, 0xeecd, 0xeecf, 0xe896, 0xeece, /*0x90-0x97*/
05764 0xe897, 0xb8c6, 0xeed0, 0xeed1, 0xeed2, 0xb6db, 0xb3ae, 0xd6d3, /*0x98-0x9f*/
05765 0xc4c6, 0xb1b5, 0xb8d6, 0xeed3, 0xeed4, 0xd4bf, 0xc7d5, 0xbefb, /*0xa0-0xaf*/
05766 0xcd9, 0xb9b3, 0xeed6, 0xeed5, 0xeed8, 0xeed7, 0xc5a5, 0xeed9, /*0xa8-0xaf*/
05767 0xeeda, 0xc7ae, 0xeedb, 0xc7af, 0xeedc, 0xb2a7, 0xeedd, 0xeede, /*0xb0-0xbf*/
05768 0xeedf, 0xee01, 0xee02, 0xee03, 0xee04, 0xee05, 0xbcd8, 0xee06, /*0xb8-0xbf*/
05769 0xd3cb, 0xccfa, 0xb2ac, 0xc1e5, 0xee05, 0xc7a6, 0xc3ad, 0xe898, /*0xc0-0xcf*/
05770 0xee06, 0xee07, 0xee08, 0xee09, 0xee0a, 0xee0b, 0xee0c, 0xe899, /*0xc8-0xcf*/
05771 0xee0d, 0xee0e, 0xee0f, 0xee10, 0xee11, 0xee12, 0xee13, 0xee14, /*0xd0-0xdf*/
05772 0xee15, 0xee16, 0xee17, 0xee18, 0xee19, 0xee1a, 0xee1b, 0xee1c, /*0xd8-0xdf*/
05773 0xee1d, 0xd5a1, 0xee1f, 0xcfb3, 0xee1a, 0xee1b, 0xe89d, 0xee1c, /*0xe0-0xef*/
05774 0xee1d, 0xfaf1, 0xee1e, 0xfaf2, 0xc3fa, 0xfaf3, 0xfaf4, 0xfaf5, /*0xe8-0xef*/
05775 0xbdc2, 0xd2bf, 0xb2f9, 0xfaf5, 0xfaf6, 0xfaf7, 0xd2f8, 0xfaf8, /*0xf0-0xf7*/
05776 0xd6fd, 0xfafa, 0xc6cc, 0xe89e, 0xfafa, 0xfafb, 0xc1b4, 0xfaf9, /*0xf8-0xff*/
05777 /* 0x9500 */
05778 0xcffa, 0xcbf8, 0xfaf6, 0xfaf7, 0xb3fa, 0xb9f8, 0xfaf6, 0xfaf7, /*0x00-0x07*/
05779 0xd0e2, 0xfefb1, 0xfefb2, 0xb7e6, 0xd0bf, 0xfefb3, 0xfefb4, 0xfefb5, /*0x08-0x0f*/
05780 0xc8f1, 0xcc0e, 0xfefb6, 0xfefb7, 0xfefb8, 0xfefb9, 0xfefba, 0xd5e0, /*0x10-0x17*/
05781 0xfefbb, 0xb4ed, 0xc3aa, 0xfefbc, 0xe89f, 0xfefbd, 0xfefbe, 0xfefbf, /*0x18-0x1f*/
05782 0xe8a0, 0xcfcf, 0xfefc0, 0xc2e0, 0xb4b8, 0xd7b6, 0xbdf5, 0xe940, /*0x20-0x27*/
05783 0xcfcf, 0xfefc3, 0xfefc1, 0xfefc2, 0xfefc4, 0xb6a7, 0xbcf6, 0xbef2, /*0x28-0x2f*/
05784 0xc3cc, 0xfefc5, 0xfefc6, 0xe941, 0xfefc7, 0xfefc8, 0xfefc9, 0xfefca, /*0x30-0x37*/
05785 0xfefcb, 0xc7c2, 0xfefc1, 0xb6cd, 0xfefcb, 0xe942, 0xfefcc, 0xfefcd, /*0x38-0x3f*/
05786 0xb6cd, 0xfefc3, 0xfefc4, 0xfefc5, 0xfefc6, 0xfefc7, 0xfefc8, 0xd5f2, /*0x40-0x47*/
05787 0xe944, 0xfefd3, 0xc4f7, 0xe945, 0xfefd4, 0xc4f8, 0xfefd5, 0xfefd6, /*0x48-0x4f*/
05788 0xb8e4, 0xb0f7, 0xfefd7, 0xfefd8, 0xfefd9, 0xe946, 0xfefda, 0xfefdb, /*0x50-0x57*/
05789 0xfefdc, 0xfefde, 0xfefde, 0xbef5, 0xfefe1, 0xfefdf, 0xfefe0, 0xfefe1, /*0x58-0x5f*/
05790 0xe948, 0xfefe2, 0xfefe3, 0xc1cd, 0xfefe4, 0xfefe5, 0xfefe6, 0xfefe7, /*0x60-0x67*/
05791 0xfefe8, 0xfefe9, 0xfefea, 0xfefeb, 0xfefec, 0xc0d8, 0xe949, 0xfefed, /*0x68-0x6f*/
05792 0xc1ad, 0xfefee, 0xfefef, 0xfefff, 0xe94a, 0xe94b, 0xcfe2, 0xe94c, /*0x70-0x77*/
05793 0xe94d, 0xe94e, 0xe94f, 0xe950, 0xe951, 0xe952, 0xe953, 0xb3a4, /*0x78-0x7f*/
05794 0xe954, 0xe955, 0xe956, 0xe957, 0xe958, 0xe959, 0xe95a, 0xe95b, /*0x80-0x87*/
05795 0xe95c, 0xe95d, 0xe95e, 0xe95f, 0xe960, 0xe961, 0xe962, 0xe963, /*0x88-0x8f*/
05796 0xe964, 0xe965, 0xe966, 0xe967, 0xe968, 0xe969, 0xe96a, 0xe96b, /*0x90-0x97*/
05797 0xe96c, 0xe96d, 0xe96e, 0xe96f, 0xe970, 0xe971, 0xe972, 0xe973, /*0x98-0x9f*/
05798 0xe974, 0xe975, 0xe976, 0xe977, 0xe978, 0xe979, 0xe97a, 0xe97b, /*0xa0-0xaf*/
05799 0xe97c, 0xe97d, 0xe97e, 0xe980, 0xe981, 0xe982, 0xe983, 0xe984, /*0xa8-0xaf*/
05800 0xe985, 0xe986, 0xe987, 0xe988, 0xe989, 0xe98a, 0xe98b, 0xe98c, /*0xb0-0xbf*/
05801 0xe98d, 0xe98e, 0xe98f, 0xe990, 0xe991, 0xe992, 0xe993, 0xe994, /*0xb8-0xbf*/
05802 0xe995, 0xe996, 0xe997, 0xe998, 0xe999, 0xe99a, 0xe99b, 0xe99c, /*0xc0-0xcf*/
05803 0xe99d, 0xe99e, 0xe99f, 0xe9a0, 0xe9a1, 0xe9a2, 0xe9a3, 0xe9a4, /*0xc8-0xcf*/
05804 0xe9a5, 0xe9a6, 0xe9a7, 0xe9a8, 0xe9a9, 0xe9aa, 0xe9ab, 0xe9ac, /*0xd0-0xdf*/
05805 0xe9ad, 0xe9ae, 0xe9af, 0xe9b0, 0xe9b1, 0xe9b2, 0xe9b3, 0xe9b4, /*0xd8-0xdf*/
05806 0xe9b5, 0xe9b6, 0xe9b7, 0xe9b8, 0xe9b9, 0xe9ba, 0xe9bb, 0xe9bc, /*0xe0-0xef*/
05807 0xc3c5, 0xe9bc, 0xc3c6, 0xe9bd, 0xe9be, 0xc3c7, 0xc3c8, 0xc3c9, /*0xe8-0xef*/
05808 0xc8f2, 0xe9bf, 0xc3c7, 0xc3c8, 0xbce4, 0xe9c9, 0xe9ca, 0xc3c6, /*0xf0-0xf7*/
05809 0xd5a2, 0xc4d6, 0xb9eb, 0xc3c5, 0xe9cb, 0xc3f6, 0xe9cc, 0xe9cd, /*0xf8-0xff*/
05810 /* 0x9600 */
05811 0xb7a7, 0xb8f3, 0xbad2, 0xe3cd, 0xe3ce, 0xd4c4, 0xe3cf, 0xe9ae, /*0x00-0x07*/
05812 0xe3d0, 0xd1cb, 0xe3d1, 0xe3d2, 0xe3d3, 0xe3d4, 0xe3d5, 0xe3d6, /*0x08-0x0f*/
05813 0xb2fb, 0xc0bb, 0xe3d6, 0xe3d7, 0xc0ab, 0xe3d7, 0xe3d8, 0xe3d9, /*0x10-0x17*/
05814 0xe9ad, 0xe9ae, 0xe9af, 0xe9b0, 0xb8b7, 0xdae2, 0xe9b1, 0xb6d3, /*0x18-0x1f*/
05815 0xe9b2, 0xdae3, 0xdae4, 0xdae5, 0xe9b3, 0xe9b4, 0xe9b5, 0xe9b6, /*0x20-0x27*/
05816 0xe9b7, 0xdae6, 0xdae7, 0xe9b8, 0xe9b9, 0xe9ba, 0xc8ee, 0xe9bb, /*0x28-0x2f*/
05817 0xe9bc, 0xdae8, 0xb7c0, 0xd1f4, 0xd2f5, 0xd5f3, 0xbdd7, 0xe9bd, /*0x30-0x37*/
05818 0xe9be, 0xe9bf, 0xe9c0, 0xd7e8, 0xdae9, 0xdaea, 0xe9bf, 0xb0a2, /*0x38-0x3f*/
05819 0xcdd3, 0xe9c1, 0xdaea, 0xe9c2, 0xb8bd, 0xbcca, 0xc2bd, 0xc2a4, /*0x40-0x47*/
05820 0xb3c2, 0xdaea, 0xe9c3, 0xc2aa, 0xc4b0, 0xbdb5, 0xe9c4, 0xe9c5, /*0x48-0x4f*/
05821 0xcfd6, 0xe9c6, 0xe9c7, 0xe9c8, 0xc9c2, 0xe9c9, 0xe9ca, 0xe9cb, /*0x50-0x57*/
05822 0xe9cc, 0xe9cd, 0xe9ce, 0xb1dd, 0xe9cf, 0xe9d0, 0xe9d1, 0xdaec, /*0x58-0x5f*/
05823 0xe9d2, 0xb6b8, 0xd4ba, 0xe9d3, 0xb3fd, 0xe9d4, 0xe9d5, 0xdaed, /*0x60-0x67*/
05824 0xd4c9, 0xcfd5, 0xc5e3, 0xe9d6, 0xdaee, 0xe9d7, 0xe9d8, 0xe9d9, /*0x68-0x6f*/
05825 0xe9da, 0xe9db, 0xdaef, 0xe9dc, 0xdaf0, 0xc1ea, 0xcdd5, 0xcfd6, /*0x70-0x77*/
05826 0xe9de, 0xe9df, 0xe9e0, 0xe9e1, 0xe9e2, 0xe9e3, 0xe9e4, 0xe9e5, /*0x78-0x7f*/
05827 0xe9e6, 0xe9e7, 0xe9e8, 0xe9e9, 0xe9ea, 0xe9eb, 0xe9ec, 0xe9ed, /*0x80-0x87*/
05828 0xdaf1, 0xe9ef, 0xe9f0, 0xcbe5, 0xeb40, 0xdaf2, 0xeb41, 0xcbe6, /*0x88-0x8f*/
05829 0xd2fe, 0xeb42, 0xeb43, 0xeb44, 0xb8f4, 0xeb45, 0xeb46, 0xdaf3, /*0x90-0x97*/
05830 0xb0af, 0xcfb6, 0xeb47, 0xeb48, 0xd5cf, 0xeb49, 0xeb4a, 0xeb4b, /*0x98-0x9f*/
05831 0xeb4c, 0xeb4d, 0xeb4e, 0xeb4f, 0xeb50, 0xeb51, 0xeb52, 0xcbed, /*0xa0-0xaf*/
05832 0xeb53, 0xeb54, 0xeb55, 0xeb56, 0xeb57, 0xeb58, 0xeb59, 0xeb5a, /*0xa8-0xaf*/
05833 0xdaf4, 0xeb5b, 0xeb5c, 0xeb5d, 0xeb5e, 0xc1a5, 0xeb5f, 0xeb60, /*0xb0-0xbf*/
05834 0xeb61, 0xf6bf, 0xeb62, 0xeb63, 0xf6c0, 0xf6c1, 0xc4d1, 0xeb64, /*0xb8-0xbf*/

```

```
05835 0xc8b8, 0xd1e3, 0xeb64, 0xeb65, 0xd0db, 0xd1c5, 0xbcaf, 0xb9cd, /*0xc0-0xc7*/
05836 0xeb66, 0xff4, 0xeb67, 0xeb68, 0xb4c6, 0xd3ba, 0xf6c2, 0xb3fb, /*0xc8-0xcf*/
05837 0xeb69, 0xeb6a, 0xf6c3, 0xeb6b, 0xeb6c, 0xb5f1, 0xeb6d, 0xeb6e, /*0xd0-0xd7*/
05838 0xeb6f, 0xeb70, 0xeb71, 0xeb72, 0xeb73, 0xeb74, 0xeb75, 0xeb76, /*0xd8-0xdf*/
05839 0xf6c5, 0xeb77, 0xeb78, 0xeb79, 0xeb7a, 0xeb7b, 0xeb7c, 0xeb7d, /*0xe0-0xe7*/
05840 0xd3ea, 0xf6a7, 0xd1a9, 0xeb7e, 0xeb80, 0xeb81, 0xeb82, 0xf6a9, /*0xe8-0xef*/
05841 0xeb83, 0xeb84, 0xeb85, 0xf6a8, 0xeb86, 0xeb87, 0xc1e3, 0xc0d7, /*0xf0-0xf7*/
05842 0xeb88, 0xb1a2, 0xeb89, 0xeb8a, 0xeb8b, 0xeb8c, 0xceed, 0xeb8d, /*0xf8-0xff*/
05843 /* 0x9700 */
05844 0xd0e8, 0xf6ab, 0xeb8e, 0xeb8f, 0xcff6, 0xeb90, 0xf6aa, 0xd5f0, /*0x00-0x07*/
05845 0xf6ac, 0xc3b9, 0xeb91, 0xeb92, 0xeb93, 0xbbf4, 0xf6ae, 0xf6ad, /*0x08-0x0f*/
05846 0xeb94, 0xeb95, 0xeb96, 0xc4de, 0xeb97, 0xeb98, 0xc1d8, 0xeb99, /*0x10-0x17*/
05847 0xeb9a, 0xeb9b, 0xeb9c, 0xeb9d, 0xcbaa, 0xeb9e, 0xcfbf, 0xeb9f, /*0x18-0x1f*/
05848 0xeba0, 0xec40, 0xec41, 0xec42, 0xec43, 0xec44, 0xec45, 0xec46, /*0x20-0x27*/
05849 0xec47, 0xec48, 0xf6af, 0xec49, 0xec4a, 0xf6b0, 0xec4b, 0xec4c, /*0x28-0x2f*/
05850 0xf6b1, 0xec4d, 0xc2b6, 0xec4e, 0xec4f, 0xec50, 0xec51, 0xec52, /*0x30-0x37*/
05851 0xb0d4, 0xc5f9, 0xec53, 0xec54, 0xec55, 0xec56, 0xf6b2, 0xec57, /*0x38-0x3f*/
05852 0xec58, 0xec59, 0xec5a, 0xec5b, 0xec5c, 0xec5d, 0xec5e, 0xec5f, /*0x40-0x4f*/
05853 0xec60, 0xec61, 0xec62, 0xec63, 0xec64, 0xec65, 0xec66, 0xec67, /*0x48-0x4f*/
05854 0xec68, 0xec69, 0xc7e0, 0xf6a6, 0xec6a, 0xec6b, 0xebb8, 0xec6c, /*0x50-0x57*/
05855 0xec6d, 0xebb2, 0xec6e, 0xb5e5, 0xec6f, 0xec70, 0xb7c7, 0xec71, /*0x58-0x5f*/
05856 0xbfbf, 0xc3d2, 0xc3e6, 0xec72, 0xec73, 0xd8cc, 0xec74, 0xec75, /*0x60-0x67*/
05857 0xec76, 0xb8ef, 0xec77, 0xec78, 0xec79, 0xec7a, 0xec7b, 0xec7c, /*0x68-0x6f*/
05858 0xec7d, 0xec7e, 0xec80, 0xbdf9, 0xd1a5, 0xec81, 0xb0d0, 0xec82, /*0x70-0x7f*/
05859 0xec83, 0xec84, 0xec85, 0xec86, 0xf7b0, 0xec87, 0xec88, 0xec89, /*0x78-0x7f*/
05860 0xec8a, 0xec8b, 0xec8c, 0xec8d, 0xec8e, 0xf7b1, 0xec8f, 0xec90, /*0x80-0x87*/
05861 0xec91, 0xec92, 0xec93, 0xd0ac, 0xec94, 0xb0b0, 0xec95, 0xec96, /*0x88-0x8f*/
05862 0xec97, 0xf7b2, 0xf7b3, 0xec98, 0xf7b4, 0xec99, 0xec9a, 0xec9b, /*0x90-0x97*/
05863 0xc7ca, 0xec9c, 0xec9d, 0xec9e, 0xec9f, 0xeca0, 0xed40, 0xed41, /*0x98-0x9f*/
05864 0xbecf, 0xed42, 0xed43, 0xf7b7, 0xed44, 0xed45, 0xed46, 0xed47, /*0xa0-0xaf*/
05865 0xed48, 0xed49, 0xed4a, 0xf7b6, 0xed4b, 0xb1de, 0xed4c, 0xf7b5, /*0xa8-0xaf*/
05866 0xed4d, 0xed4e, 0xf7b8, 0xed4f, 0xf7b9, 0xed50, 0xed51, 0xed52, /*0xb0-0xb7*/
05867 0xed53, 0xed54, 0xed55, 0xed56, 0xed57, 0xed58, 0xed59, 0xed5a, /*0xb8-0xbf*/
05868 0xed5b, 0xed5c, 0xed5d, 0xed5e, 0xed5f, 0xed60, 0xed61, 0xed62, /*0xc0-0xc7*/
05869 0xed63, 0xed64, 0xed65, 0xed66, 0xed67, 0xed68, 0xed69, 0xed6a, /*0xc8-0xcf*/
05870 0xed6b, 0xed6c, 0xed6d, 0xed6e, 0xed6f, 0xed70, 0xed71, 0xed72, /*0xd0-0xd7*/
05871 0xed73, 0xed74, 0xed75, 0xed76, 0xed77, 0xed78, 0xed79, 0xed7a, /*0xd8-0xdf*/
05872 0xed7b, 0xed7c, 0xed7d, 0xed7e, 0xed80, 0xed81, 0xcea4, 0xc8cd, /*0xe0-0xe7*/
05873 0xed82, 0xbaab, 0xebb8, 0xebb9, 0xebb2, 0xed83, 0xed84, /*0xe8-0xef*/
05874 0xed85, 0xed86, 0xed87, 0xd2f4, 0xed88, 0xd4cf, 0xc9d8, 0xed89, /*0xf0-0xf7*/
05875 0xed8a, 0xed8b, 0xed8c, 0xed8d, 0xed8e, 0xed8f, 0xed90, 0xed91, /*0xf8-0xff*/
05876 /* 0x9800 */
05877 0xed92, 0xed93, 0xed94, 0xed95, 0xed96, 0xed97, 0xed98, 0xed99, /*0x00-0x07*/
05878 0xed9a, 0xed9b, 0xed9c, 0xed9d, 0xed9e, 0xed9f, 0xeda0, 0xee40, /*0x08-0x0f*/
05879 0xee41, 0xee42, 0xee43, 0xee44, 0xee45, 0xee46, 0xee47, 0xee48, /*0x10-0x17*/
05880 0xee49, 0xee4a, 0xee4b, 0xee4c, 0xee4d, 0xee4e, 0xee4f, 0xee50, /*0x18-0x1f*/
05881 0xee51, 0xee52, 0xee53, 0xee54, 0xee55, 0xee56, 0xee57, 0xee58, /*0x20-0x27*/
05882 0xee59, 0xee5a, 0xee5b, 0xee5c, 0xee5d, 0xee5e, 0xee5f, 0xee60, /*0x28-0x2f*/
05883 0xee61, 0xee62, 0xee63, 0xee64, 0xee65, 0xee66, 0xee67, 0xee68, /*0x30-0x37*/
05884 0xee69, 0xee6a, 0xee6b, 0xee6c, 0xee6d, 0xee6e, 0xee6f, 0xee70, /*0x38-0x3f*/
05885 0xee71, 0xee72, 0xee73, 0xee74, 0xee75, 0xee76, 0xee77, 0xee78, /*0x40-0x47*/
05886 0xee79, 0xee7a, 0xee7b, 0xee7c, 0xee7d, 0xee7e, 0xee80, 0xee81, /*0x48-0x4f*/
05887 0xee82, 0xee83, 0xee84, 0xee85, 0xee86, 0xee87, 0xee88, 0xee89, /*0x50-0x57*/
05888 0xee8a, 0xee8b, 0xee8c, 0xee8d, 0xee8e, 0xee8f, 0xee90, 0xee91, /*0x58-0x5f*/
05889 0xee92, 0xee93, 0xee94, 0xee95, 0xee96, 0xee97, 0xee98, 0xee99, /*0x60-0x67*/
05890 0xee9a, 0xee9b, 0xee9c, 0xee9d, 0xee9e, 0xee9f, 0xeea0, 0xfef4, /*0x68-0x6f*/
05891 0xfef41, 0xfef42, 0xfef43, 0xfef44, 0xfef45, 0xd2b3, 0xb6a5, 0xc7ea, /*0x70-0x77*/
05892 0xf1fc, 0xcfee, 0xcbb3, 0xd0eb, 0xe7ef, 0xcde7, 0xb9cb, 0xb6d9, /*0x78-0x7f*/
05893 0xf1fd, 0xb0e4, 0xcbbc, 0xf1fe, 0xd4a4, 0xc2ad, 0xc1ec, 0xc6c4, /*0x80-0x87*/
05894 0xb6b1, 0xf2a1, 0xbcd5, 0xfef46, 0xf2a2, 0xf2a3, 0xfef47, 0xf2a4, /*0x88-0x8f*/
05895 0xd2c3, 0xc6b5, 0xfef48, 0xcdc7, 0xf2a5, 0xfef49, 0xd3b1, 0xbfc5, /*0x90-0x97*/
05896 0xcce2, 0xfef4a, 0xf2a6, 0xf2a7, 0xd1d5, 0xb6ee, 0xf2a8, 0xf2a9, /*0x98-0x9f*/
05897 0xb5df, 0xf2aa, 0xf2ab, 0xfef4b, 0xb2fc, 0xf2ac, 0xf2ad, 0xc8a7, /*0xa0-0xaf*/
05898 0xfef4c, 0xfef4d, 0xfef4e, 0xfef4f, 0xfef50, 0xfef51, 0xfef52, 0xfef53, /*0xa8-0xaf*/
05899 0xfef54, 0xfef55, 0xfef56, 0xfef57, 0xfef58, 0xfef59, 0xfef5a, 0xfef5b, /*0xb0-0xb7*/
05900 0xfef5c, 0xfef5d, 0xfef5e, 0xfef5f, 0xfef60, 0xfef61, 0xfef62, 0xfef63, /*0xb8-0xbf*/
05901 0xfef64, 0xfef65, 0xfef66, 0xfef67, 0xfef68, 0xfef69, 0xfef6a, 0xfef6b, /*0xc0-0xc7*/
05902 0xfef6c, 0xfef6d, 0xfef6e, 0xfef6f, 0xfef70, 0xfef71, 0xb7e7, 0xfef72, /*0xc8-0xcf*/
05903 0xfef73, 0xecaa, 0xecab, 0xfef74, 0xecac, 0xfef75, 0xfef76, /*0xd0-0xd7*/
05904 0xc6ae, 0xecad, 0xecae, 0xfef77, 0xfef78, 0xfef79, 0xb7c9, 0xcab3, /*0xd8-0xdf*/
05905 0xfef7a, 0xfef7b, 0xfef7c, 0xfef7d, 0xfef7e, 0xfef80, 0xfef81, 0xe2b8, /*0xe0-0xe7*/
05906 0xf7cf, 0xfef82, 0xfef83, 0xfef84, 0xfef85, 0xfef86, 0xfef87, 0xfef88, /*0xe8-0xef*/
05907 0xfef89, 0xfef8a, 0xfef8b, 0xfef8c, 0xfef8d, 0xfef8e, 0xfef8f, 0xfef90, /*0xf0-0xf7*/
05908 0xfef91, 0xfef92, 0xfef93, 0xfef94, 0xfef95, 0xfef96, 0xfef97, 0xfef98, /*0xf8-0xff*/
05909 /* 0x9900 */
05910 0xfef99, 0xfef9a, 0xfef9b, 0xfef9c, 0xfef9d, 0xfef9e, 0xfef9f, 0xfefa0, /*0x00-0x07*/
05911 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x08-0x0f*/
05912 0xb2cd, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x10-0x17*/
05913 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x18-0x1f*/
05914 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x20-0x27*/
05915 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x28-0x2f*/
05916 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x30-0x37*/
05917 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x38-0x3f*/
05918 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x40-0x47*/
05919 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x48-0x4f*/
05920 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x50-0x57*/
05921 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, 0xfef9f, /*0x58-0x5f*/
```

```
05922 0xf094, 0xf095, 0xf096, 0xe2bb, 0xf097, 0xbca2, 0xf098, 0xe2bc, /*0x60-0x67*/
05923 0xe2bd, 0xe2be, 0xe2bf, 0xe2c0, 0xe2c1, 0xb7b9, 0xd2fb, 0xbda4, /*0x68-0x6f*/
05924 0xcace, 0xb1a5, 0xcbc7, 0xf099, 0xe2c2, 0xb6fc, 0xc8c4, 0xe2c3, /*0x70-0x77*/
05925 0xf09a, 0xf09b, 0xbdc8, 0xf09c, 0xb1fd, 0xe2c4, 0xf09d, 0xb6f6, /*0x78-0x7f*/
05926 0xe2c5, 0xc4d9, 0xf09e, 0xf09f, 0xe2c6, 0xcfd4, 0xb9dd, 0xe2c7, /*0x80-0x87*/
05927 0xc0a1, 0xf0a0, 0xe2c8, 0xb2f6, 0xf140, 0xe2c9, 0xf141, 0xc1f3, /*0x88-0x8f*/
05928 0xe2ca, 0xe2cb, 0xc2f8, 0xe2cc, 0xe2cd, 0xe2ce, 0xcad7, 0xd8b8, /*0x90-0x97*/
05929 0xd9e5, 0xcfe3, 0xf142, 0xf143, 0xf144, 0xf145, 0xf146, 0xf147, /*0x98-0x9f*/
05930 0xf148, 0xf149, 0xf14a, 0xf14b, 0xf14c, 0xf0a5, 0xf14d, 0xf14e, /*0xa0-0xa7*/
05931 0xdc0b, 0xf14f, 0xf150, 0xf151, 0xf152, 0xf153, 0xf154, 0xf155, /*0xa8-0xaf*/
05932 0xf156, 0xf157, 0xf158, 0xf159, 0xf15a, 0xf15b, 0xf15c, 0xf15d, /*0xb0-0xb7*/
05933 0xf15e, 0xf15f, 0xf160, 0xf161, 0xf162, 0xf163, 0xf164, 0xf165, /*0xb8-0xbf*/
05934 0xf166, 0xf167, 0xf168, 0xf169, 0xf16a, 0xf16b, 0xf16c, 0xf16d, /*0xc0-0xc7*/
05935 0xf16e, 0xf16f, 0xf170, 0xf171, 0xf172, 0xf173, 0xf174, 0xf175, /*0xc8-0xcf*/
05936 0xf176, 0xf177, 0xf178, 0xf179, 0xf17a, 0xf17b, 0xf17c, 0xf17d, /*0xd0-0xd7*/
05937 0xf17e, 0xf180, 0xf181, 0xf182, 0xf183, 0xf184, 0xf185, 0xf186, /*0xd8-0xdf*/
05938 0xf187, 0xf188, 0xf189, 0xf18a, 0xf18b, 0xf18c, 0xf18d, 0xf18e, /*0xe0-0xe7*/
05939 0xf18f, 0xf190, 0xf191, 0xf192, 0xf193, 0xf194, 0xf195, 0xf196, /*0xe8-0xef*/
05940 0xf197, 0xf198, 0xf199, 0xf19a, 0xf19b, 0xf19c, 0xf19d, 0xf19e, /*0xf0-0xf7*/
05941 0xf19f, 0xf1a0, 0xf240, 0xf241, 0xf242, 0xf243, 0xf244, 0xf245, /*0xf8-0xff*/
05942 /* 0x9a00 */
05943 0xf246, 0xf247, 0xf248, 0xf249, 0xf24a, 0xf24b, 0xf24c, 0xf24d, /*0x00-0x07*/
05944 0xf24e, 0xf24f, 0xf250, 0xf251, 0xf252, 0xf253, 0xf254, 0xf255, /*0x08-0x0f*/
05945 0xf256, 0xf257, 0xf258, 0xf259, 0xf25a, 0xf25b, 0xf25c, 0xf25d, /*0x10-0x17*/
05946 0xf25e, 0xf25f, 0xf260, 0xf261, 0xf262, 0xf263, 0xf264, 0xf265, /*0x18-0x1f*/
05947 0xf266, 0xf267, 0xf268, 0xf269, 0xf26a, 0xf26b, 0xf26c, 0xf26d, /*0x20-0x27*/
05948 0xf26e, 0xf26f, 0xf270, 0xf271, 0xf272, 0xf273, 0xf274, 0xf275, /*0x28-0x2f*/
05949 0xf276, 0xf277, 0xf278, 0xf279, 0xf27a, 0xf27b, 0xf27c, 0xf27d, /*0x30-0x37*/
05950 0xf27e, 0xf280, 0xf281, 0xf282, 0xf283, 0xf284, 0xf285, 0xf286, /*0x38-0x3f*/
05951 0xf287, 0xf288, 0xf289, 0xf28a, 0xf28b, 0xf28c, 0xf28d, 0xf28e, /*0x40-0x47*/
05952 0xf28f, 0xf290, 0xf291, 0xf292, 0xf293, 0xf294, 0xf295, 0xf296, /*0x48-0x4f*/
05953 0xf297, 0xf298, 0xf299, 0xf29a, 0xf29b, 0xf29c, 0xf29d, 0xf29e, /*0x50-0x57*/
05954 0xf29f, 0xf2a0, 0xf340, 0xf341, 0xf342, 0xf343, 0xf344, 0xf345, /*0x58-0x5f*/
05955 0xf346, 0xf347, 0xf348, 0xf349, 0xf34a, 0xf34b, 0xf34c, 0xf34d, /*0x60-0x67*/
05956 0xf34e, 0xf34f, 0xf350, 0xf351, 0xc2ed, 0xd4a6, 0xcdd4, 0xd1b1, /*0x68-0x6f*/
05957 0xb3db, 0xc7fd, 0xf352, 0xb2b5, 0xc2bf, 0xe6e0, 0xcabb, 0xe6e1, /*0x70-0x77*/
05958 0xe6e2, 0xbcd4, 0xe6e3, 0xd7a4, 0xcdd5, 0xe6e5, 0xbcd5, 0xe6e4, /*0x78-0x7f*/
05959 0xe6e6, 0xe6e7, 0xc2ee, 0xf353, 0xbdb5, 0xe6e8, 0xc2e6, 0xbaa7, /*0x80-0x87*/
05960 0xe6e9, 0xf354, 0xe6ea, 0xb3d2, 0xd1e9, 0xf355, 0xf356, 0xbfa5, /*0x88-0x8f*/
05961 0xe6eb, 0xc6ef, 0xe6ec, 0xe6ed, 0xf357, 0xf358, 0xe6ee, 0xc6ad, /*0x90-0x97*/
05962 0xe6ef, 0xf359, 0xc9a7, 0xe6f0, 0xe6f1, 0xe6f2, 0xe5b9, 0xe6f3, /*0x98-0x9f*/
05963 0xe6f4, 0xc2e2, 0xe6f5, 0xe6f6, 0xd6e8, 0xe6f7, 0xf35a, 0xe6f8, /*0xa0-0xaf*/
05964 0xb9c7, 0xf35b, 0xf35c, 0xf35d, 0xf35e, 0xf35f, 0xf360, 0xf361, /*0xa8-0xaf*/
05965 0xf7bb, 0xf7ba, 0xf362, 0xf363, 0xf364, 0xf365, 0xf7be, 0xf7bc, /*0xb0-0xb7*/
05966 0xbaa1, 0xf366, 0xf7bf, 0xf367, 0xf7c0, 0xf368, 0xf369, 0xf36a, /*0xb8-0xbf*/
05967 0xf7c2, 0xf7c1, 0xf7c4, 0xf36b, 0xf36c, 0xf7c3, 0xf36d, 0xf36e, /*0xc0-0xc7*/
05968 0xf36f, 0xf370, 0xf371, 0xf7c5, 0xf7c6, 0xf372, 0xf373, 0xf374, /*0xc8-0xcf*/
05969 0xf375, 0xf376, 0xf377, 0xcbe8, 0xf377, 0xf378, 0xf379, 0xf37a, /*0xd0-0xd7*/
05970 0xb8df, 0xf37b, 0xf37c, 0xf37d, 0xf37e, 0xf380, 0xf381, 0xf7d4, /*0xd8-0xdf*/
05971 0xf382, 0xf7d5, 0xf383, 0xf384, 0xf385, 0xf386, 0xf7d6, 0xf387, /*0xe0-0xe7*/
05972 0xf388, 0xf389, 0xf38a, 0xf7d8, 0xf38b, 0xf7da, 0xf38c, 0xf7d7, /*0xe8-0xef*/
05973 0xf38d, 0xf38e, 0xf38f, 0xf390, 0xf391, 0xf392, 0xf393, 0xf394, /*0xf0-0xf7*/
05974 0xf395, 0xf7db, 0xf396, 0xf7d9, 0xf397, 0xf398, 0xf399, 0xf39a, /*0xf8-0xff*/
05975 /* 0x9b00 */
05976 0xf39b, 0xf39c, 0xf39d, 0xd7d7, 0xf39e, 0xf39f, 0xf3a0, 0xf440, /*0x00-0x07*/
05977 0xf7dc, 0xf441, 0xf442, 0xf443, 0xf444, 0xf445, 0xf446, 0xf7dd, /*0x08-0x0f*/
05978 0xf447, 0xf448, 0xf449, 0xf7de, 0xf44a, 0xf44b, 0xf44c, 0xf44d, /*0x10-0x17*/
05979 0xf44e, 0xf44f, 0xf450, 0xf451, 0xf452, 0xf453, 0xf454, 0xf7df, /*0x18-0x1f*/
05980 0xf455, 0xf456, 0xf457, 0xf7e0, 0xf458, 0xf459, 0xf45a, 0xf45b, /*0x20-0x27*/
05981 0xf45c, 0xf45d, 0xf45e, 0xf45f, 0xf460, 0xf461, 0xf462, 0xdbcb, /*0x28-0x2f*/
05982 0xf463, 0xf464, 0xd8aa, 0xf465, 0xf466, 0xf467, 0xf468, 0xf469, /*0x30-0x37*/
05983 0xf46a, 0xf46b, 0xf46c, 0xe5f7, 0xb9ed, 0xf46d, 0xf46e, 0xf46f, /*0x38-0x3f*/
05984 0xf470, 0xbfffd, 0xbbea, 0xf7c9, 0xc6c7, 0xf7c8, 0xf471, 0xf7ca, /*0x40-0x47*/
05985 0xf7cc, 0xf7cb, 0xf472, 0xf473, 0xf474, 0xf7cd, 0xf475, 0xcbea, /*0x48-0x4f*/
05986 0xf476, 0xf7ce, 0xf477, 0xf478, 0xc4a7, 0xf479, 0xf47a, 0xf47b, /*0x50-0x57*/
05987 0xf47c, 0xf47d, 0xf47e, 0xf480, 0xf481, 0xf482, 0xf483, 0xf484, /*0x58-0x5f*/
05988 0xf485, 0xf486, 0xf487, 0xf488, 0xf489, 0xf48a, 0xf48b, 0xf48c, /*0x60-0x67*/
05989 0xf48d, 0xf48e, 0xf48f, 0xf490, 0xf491, 0xf492, 0xf493, 0xf494, /*0x68-0x6f*/
05990 0xf495, 0xf496, 0xf497, 0xf498, 0xf499, 0xf49a, 0xf49b, 0xf49c, /*0x70-0x77*/
05991 0xf49d, 0xf49e, 0xf49f, 0xf540, 0xf540, 0xf541, 0xf542, 0xf543, /*0x78-0x7f*/
05992 0xf544, 0xf545, 0xf546, 0xf547, 0xf548, 0xf549, 0xf54a, 0xf54b, /*0x80-0x87*/
05993 0xf54c, 0xf54d, 0xf54e, 0xf54f, 0xf550, 0xf551, 0xf552, 0xf553, /*0x88-0x8f*/
05994 0xf554, 0xf555, 0xf556, 0xf557, 0xf558, 0xf559, 0xf55a, 0xf55b, /*0x90-0x97*/
05995 0xf55c, 0xf55d, 0xf55e, 0xf55f, 0xf560, 0xf561, 0xf562, 0xf563, /*0x98-0x9f*/
05996 0xf564, 0xf565, 0xf566, 0xf567, 0xf568, 0xf569, 0xf56a, 0xf56b, /*0xa0-0xaf*/
05997 0xf56c, 0xf56d, 0xf56e, 0xf56f, 0xf570, 0xf571, 0xf572, 0xf573, /*0xa8-0xaf*/
05998 0xf574, 0xf575, 0xf576, 0xf577, 0xf578, 0xf579, 0xf57a, 0xf57b, /*0xb0-0xb7*/
05999 0xf57c, 0xf57d, 0xf57e, 0xf580, 0xf581, 0xf582, 0xf583, 0xf584, /*0xb8-0xbf*/
06000 0xf585, 0xf586, 0xf587, 0xf588, 0xf589, 0xf58a, 0xf58b, 0xf58c, /*0xc0-0xc7*/
06001 0xf58d, 0xf58e, 0xf58f, 0xf590, 0xf591, 0xf592, 0xf593, 0xf594, /*0xc8-0xcf*/
06002 0xf595, 0xf596, 0xf597, 0xf598, 0xf599, 0xf59a, 0xf59b, 0xf59c, /*0xd0-0xd7*/
06003 0xf59d, 0xf59e, 0xf59f, 0xf5a0, 0xf640, 0xf641, 0xf642, 0xf643, /*0xd8-0xdf*/
06004 0xf644, 0xf645, 0xf646, 0xf647, 0xf648, 0xf649, 0xf64a, 0xf64b, /*0xe0-0xe7*/
06005 0xf64c, 0xf64d, 0xf64e, 0xf64f, 0xf650, 0xf651, 0xf652, 0xf653, /*0xe8-0xef*/
06006 0xf654, 0xf655, 0xf656, 0xf657, 0xf658, 0xf659, 0xf65a, 0xf65b, /*0xf0-0xf7*/
06007 0xf65c, 0xf65d, 0xf65e, 0xf65f, 0xf660, 0xf661, 0xf662, 0xf663, /*0xf8-0xff*/
06008 /* 0x9c00 */
```

```
06009 0xf664, 0xf665, 0xf666, 0xf667, 0xf668, 0xf669, 0xf66a, 0xf66b, /*0x00-0x07*/
06010 0xf66c, 0xf66d, 0xf66e, 0xf66f, 0xf670, 0xf671, 0xf672, 0xf673, /*0x08-0x0f*/
06011 0xf674, 0xf675, 0xf676, 0xf677, 0xf678, 0xf679, 0xf67a, 0xf67b, /*0x10-0x1f*/
06012 0xf67c, 0xf67d, 0xf67e, 0xf680, 0xf681, 0xf682, 0xf683, 0xf684, /*0x18-0x1f*/
06013 0xf685, 0xf686, 0xf687, 0xf688, 0xf689, 0xf68a, 0xf68b, 0xf68c, /*0x20-0x27*/
06014 0xf68d, 0xf68e, 0xf68f, 0xf690, 0xf691, 0xf692, 0xf693, 0xf694, /*0x28-0x2f*/
06015 0xf695, 0xf696, 0xf697, 0xf698, 0xf699, 0xf69a, 0xf69b, 0xf69c, /*0x30-0x37*/
06016 0xf69d, 0xf69e, 0xf69f, 0xf6a0, 0xf740, 0xf741, 0xf742, 0xf743, /*0x38-0x3f*/
06017 0xf744, 0xf745, 0xf746, 0xf747, 0xf748, 0xf749, 0xf74a, 0xf74b, /*0x40-0x4f*/
06018 0xf74c, 0xf74d, 0xf74e, 0xf74f, 0xf750, 0xf751, 0xf752, 0xf753, /*0x48-0x4f*/
06019 0xf754, 0xf755, 0xf756, 0xf757, 0xf758, 0xf759, 0xf75a, 0xf75b, /*0x50-0x5f*/
06020 0xf75c, 0xf75d, 0xf75e, 0xf75f, 0xf760, 0xf761, 0xf762, 0xf763, /*0x58-0x5f*/
06021 0xf764, 0xf765, 0xf766, 0xf767, 0xf768, 0xf769, 0xf76a, 0xf76b, /*0x60-0x6f*/
06022 0xf76c, 0xf76d, 0xf76e, 0xf76f, 0xf770, 0xf771, 0xf772, 0xf773, /*0x68-0x6f*/
06023 0xf774, 0xf775, 0xf776, 0xf777, 0xf778, 0xf779, 0xf77a, 0xf77b, /*0x70-0x7f*/
06024 0xf77c, 0xf77d, 0xf77e, 0xf780, 0xd3e3, 0xf781, 0xf782, 0xf6cf, /*0x78-0x7f*/
06025 0xf783, 0xc2b3, 0xf6d0, 0xf784, 0xf785, 0xf6d1, 0xf6d2, 0xf6d3, /*0x80-0x8f*/
06026 0xf6d4, 0xf786, 0xf787, 0xf6d6, 0xf788, 0xb1ab, 0xf6d7, 0xf789, /*0x88-0x8f*/
06027 0xf6d8, 0xf6d9, 0xf6da, 0xf78a, 0xf6db, 0xf6dc, 0xf78b, 0xf78c, /*0x90-0x9f*/
06028 0xf78d, 0xf78e, 0xf6dd, 0xf6de, 0xcfcf, 0xf78f, 0xf6de, 0xf6e0, /*0x98-0x9f*/
06029 0xf6e1, 0xf6e2, 0xf6e3, 0xf6e4, 0xc0f0, 0xf6e5, 0xf6e6, 0xf6e7, /*0xa0-0xaf*/
06030 0xf6e8, 0xf6e9, 0xf790, 0xf6ea, 0xf791, 0xf6eb, 0xf6ec, 0xf792, /*0xa8-0xaf*/
06031 0xf6ed, 0xf6ee, 0xf6ef, 0xf6f0, 0xf6f1, 0xf6f2, 0xf6f3, 0xf6f4, /*0xb0-0xbf*/
06032 0xb6ea, 0xf793, 0xf6f5, 0xf6f6, 0xf6f7, 0xf6f8, 0xf794, 0xf795, /*0xb8-0xbf*/
06033 0xf796, 0xf797, 0xf798, 0xc8fa, 0xf6f9, 0xf6fa, 0xf6fb, 0xf6fc, /*0xc0-0xcf*/
06034 0xf799, 0xf79a, 0xf6fd, 0xf6fe, 0xf7a1, 0xf7a2, 0xf7a3, 0xf7a4, /*0xc8-0xcf*/
06035 0xf7a5, 0xf79b, 0xf79c, 0xf7a6, 0xf7a7, 0xf7a8, 0xb1ee, 0xf7a9, /*0xd0-0xdf*/
06036 0xf7aa, 0xf7ab, 0xf79d, 0xf79e, 0xf7ac, 0xf7ad, 0xc1db, 0xf7ae, /*0xd8-0xdf*/
06037 0xf79f, 0xf7a0, 0xf7af, 0xf840, 0xf841, 0xf842, 0xf843, 0xf844, /*0xe0-0xef*/
06038 0xf845, 0xf846, 0xf847, 0xf848, 0xf849, 0xf84a, 0xf84b, 0xf84c, /*0xe8-0xef*/
06039 0xf84d, 0xf84e, 0xf84f, 0xf850, 0xf851, 0xf852, 0xf853, 0xf854, /*0xf0-0xf7*/
06040 0xf855, 0xf856, 0xf857, 0xf858, 0xf859, 0xf85a, 0xf85b, 0xf85c, /*0xf8-0xff*/
06041 /* 0x9d00 */
06042 0xf85d, 0xf85e, 0xf85f, 0xf860, 0xf861, 0xf862, 0xf863, 0xf864, /*0x00-0x07*/
06043 0xf865, 0xf866, 0xf867, 0xf868, 0xf869, 0xf86a, 0xf86b, 0xf86c, /*0x08-0x0f*/
06044 0xf86d, 0xf86e, 0xf86f, 0xf870, 0xf871, 0xf872, 0xf873, 0xf874, /*0x10-0x1f*/
06045 0xf875, 0xf876, 0xf877, 0xf878, 0xf879, 0xf87a, 0xf87b, 0xf87c, /*0x18-0x1f*/
06046 0xf87d, 0xf87e, 0xf880, 0xf881, 0xf882, 0xf883, 0xf884, 0xf885, /*0x20-0x27*/
06047 0xf886, 0xf887, 0xf888, 0xf889, 0xf88a, 0xf88b, 0xf88c, 0xf88d, /*0x28-0x2f*/
06048 0xf88e, 0xf88f, 0xf890, 0xf891, 0xf892, 0xf893, 0xf894, 0xf895, /*0x30-0x37*/
06049 0xf896, 0xf897, 0xf898, 0xf899, 0xf89a, 0xf89b, 0xf89c, 0xf89d, /*0x38-0x3f*/
06050 0xf89e, 0xf89f, 0xf8a0, 0xf940, 0xf941, 0xf942, 0xf943, 0xf944, /*0x40-0x4f*/
06051 0xf945, 0xf946, 0xf947, 0xf948, 0xf949, 0xf94a, 0xf94b, 0xf94c, /*0x48-0x4f*/
06052 0xf94d, 0xf94e, 0xf94f, 0xf950, 0xf951, 0xf952, 0xf953, 0xf954, /*0x50-0x5f*/
06053 0xf955, 0xf956, 0xf957, 0xf958, 0xf959, 0xf95a, 0xf95b, 0xf95c, /*0x58-0x5f*/
06054 0xf95d, 0xf95e, 0xf95f, 0xf960, 0xf961, 0xf962, 0xf963, 0xf964, /*0x60-0x6f*/
06055 0xf965, 0xf966, 0xf967, 0xf968, 0xf969, 0xf96a, 0xf96b, 0xf96c, /*0x68-0x6f*/
06056 0xf96d, 0xf96e, 0xf96f, 0xf970, 0xf971, 0xf972, 0xf973, 0xf974, /*0x70-0x7f*/
06057 0xf975, 0xf976, 0xf977, 0xf978, 0xf979, 0xf97a, 0xf97b, 0xf97c, /*0x78-0x7f*/
06058 0xf97d, 0xf97e, 0xf980, 0xf981, 0xf982, 0xf983, 0xf984, 0xf985, /*0x80-0x8f*/
06059 0xf986, 0xf987, 0xf988, 0xf989, 0xf98a, 0xf98b, 0xf98c, 0xf98d, /*0x88-0x8f*/
06060 0xf98e, 0xf98f, 0xf990, 0xf991, 0xf992, 0xf993, 0xf994, 0xf995, /*0x90-0x9f*/
06061 0xf996, 0xf997, 0xf998, 0xf999, 0xf99a, 0xf99b, 0xf99c, 0xf99d, /*0x98-0x9f*/
06062 0xf99e, 0xf99f, 0xf9a0, 0xfa40, 0xfa41, 0xfa42, 0xfa43, 0xfa44, /*0xa0-0xaf*/
06063 0xfa45, 0xfa46, 0xfa47, 0xfa48, 0xfa49, 0xfa4a, 0xfa4b, 0xfa4c, /*0xa8-0xaf*/
06064 0xfa4d, 0xfa4e, 0xfa4f, 0xfa50, 0xfa51, 0xfa52, 0xfa53, 0xfa54, /*0xb0-0xbf*/
06065 0xfa55, 0xfa56, 0xfa57, 0xfa58, 0xfa59, 0xfa5a, 0xfa5b, 0xfa5c, /*0xb8-0xbf*/
06066 0xfa5d, 0xfa5e, 0xfa5f, 0xfa60, 0xfa61, 0xfa62, 0xfa63, 0xfa64, /*0xc0-0xcf*/
06067 0xfa65, 0xfa66, 0xfa67, 0xfa68, 0xfa69, 0xfa6a, 0xfa6b, 0xfa6c, /*0xc8-0xcf*/
06068 0xfa6d, 0xfa6e, 0xfa6f, 0xfa70, 0xfa71, 0xfa72, 0xfa73, 0xfa74, /*0xd0-0xdf*/
06069 0xfa75, 0xfa76, 0xfa77, 0xfa78, 0xfa79, 0xfa7a, 0xfa7b, 0xfa7c, /*0xd8-0xdf*/
06070 0xfa7d, 0xfa7e, 0xfa80, 0xfa81, 0xfa82, 0xfa83, 0xfa84, 0xfa85, /*0xe0-0xef*/
06071 0xfa86, 0xfa87, 0xfa88, 0xfa89, 0xfa8a, 0xfa8b, 0xfa8c, 0xfa8d, /*0xe8-0xef*/
06072 0xfa8e, 0xfa8f, 0xfa90, 0xfa91, 0xfa92, 0xfa93, 0xfa94, 0xfa95, /*0xf0-0xf7*/
06073 0xfa96, 0xfa97, 0xfa98, 0xfa99, 0xfa9a, 0xfa9b, 0xfa9c, 0xfa9d, /*0xf8-0xff*/
06074 /* 0x9e00 */
06075 0xfa9e, 0xfa9f, 0xfaa0, 0xfb40, 0xfb41, 0xfb42, 0xfb43, 0xfb44, /*0x00-0x07*/
06076 0xfb45, 0xfb46, 0xfb47, 0xfb48, 0xfb49, 0xfb4a, 0xfb4b, 0xfb4c, /*0x08-0x0f*/
06077 0xfb4d, 0xfb4e, 0xfb4f, 0xfb50, 0xfb51, 0xfb52, 0xfb53, 0xfb54, /*0x10-0x1f*/
06078 0xfb55, 0xfb56, 0xfb57, 0xfb58, 0xfb59, 0xfb5a, 0xfb5b, 0xc4f1, /*0x18-0x1f*/
06079 0xf0af, 0xbca6, 0xf0b0, 0xc3f9, 0xfb5c, 0xc5b8, 0xd1bb, 0xfb5d, /*0x20-0x2f*/
06080 0xf0b1, 0xf0b2, 0xf0b3, 0xf0b4, 0xf0b5, 0xd1bc, 0xfb5e, 0xd1ec, /*0x28-0x2f*/
06081 0xfb5f, 0xf0b7, 0xf0b6, 0xd4a7, 0xfb60, 0xcd2, 0xf0b8, 0xf0ba, /*0x30-0x3f*/
06082 0xf0b9, 0xf0bb, 0xf0bc, 0xfb61, 0xfb62, 0xb8eb, 0xf0bd, 0xbae8, /*0x38-0x3f*/
06083 0xfb63, 0xf0be, 0xf0bf, 0xb8ee, 0xf0c0, 0xb8ec, 0xf0c1, 0xf0c2, /*0x40-0x4f*/
06084 0xf0c3, 0xf0c4, 0xc8b5, 0xf0c5, 0xf0c6, 0xfb64, 0xf0c7, 0xc5f4, /*0x48-0x4f*/
06085 0xfb65, 0xf0c8, 0xfb66, 0xfb67, 0xfb68, 0xf0c9, 0xfb69, 0xf0ca, /*0x50-0x5f*/
06086 0xf7bd, 0xfb6a, 0xf0cb, 0xf0cc, 0xf0cd, 0xfb6b, 0xf0ce, 0xfb6c, /*0x58-0x5f*/
06087 0xfb6d, 0xfb6e, 0xfb6f, 0xf0cf, 0xbad7, 0xfb70, 0xf0d0, 0xf0d1, /*0x60-0x6f*/
06088 0xf0d2, 0xf0d3, 0xf0d4, 0xf0d5, 0xf0d6, 0xf0d8, 0xfb71, 0xfb72, /*0x68-0x6f*/
06089 0xd3a5, 0xf0d7, 0xfb73, 0xf0d9, 0xfb74, 0xfb75, 0xfb76, 0xfb77, /*0x70-0x7f*/
06090 0xfb78, 0xfb79, 0xfb7a, 0xfb7b, 0xfb7c, 0xfb7d, 0xf5ba, 0xc2b9, /*0x78-0x7f*/
06091 0xfb7e, 0xfb80, 0xf7e4, 0xfb81, 0xfb82, 0xfb83, 0xfb84, 0xf7e5, /*0x80-0x8f*/
06092 0xf7e6, 0xfb85, 0xfb86, 0xf7e7, 0xfb87, 0xfb88, 0xfb89, 0xfb8a, /*0x88-0x8f*/
06093 0xfb8b, 0xfb8c, 0xf7e8, 0xc2b4, 0xfb8d, 0xfb8e, 0xfb8f, 0xfb90, /*0x90-0x9f*/
06094 0xfb91, 0xfb92, 0xfb93, 0xfb94, 0xfb95, 0xf7ea, 0xfb96, 0xf7eb, /*0x98-0x9f*/
06095 0xfb97, 0xfb98, 0xfb99, 0xfb9a, 0xfb9b, 0xfb9c, 0xc2f3, 0xfb9d, /*0xa0-0xaf*/
```



```

06096 0xfb9e, 0xfb9f, 0xfb0, 0xfc40, 0xfc41, 0xfc42, 0xfc43, 0xfc44, /*0xa8-0xaf*/
06097 0xfc45, 0xfc46, 0xfc47, 0xfc48, 0xf4f0, 0xfc49, 0xfc4a, 0xfc4b, /*0xb0-0xb7*/
06098 0xf4ef, 0xf4c, 0xf4c, 0xc2e9, 0xfc4e, 0xf7e1, 0xf7e2, 0xfc4f, /*0xb8-0xbf*/
06099 0xfc50, 0xfc51, 0xfc52, 0xfc53, 0xbbc6, 0xfc54, 0xfc55, 0xfc56, /*0xc0-0xc7*/
06100 0xfc57, 0xd9e4, 0xfc58, 0xfc59, 0xfc5a, 0xcaf2, 0xc0e8, 0xf0a4, /*0xc8-0xcf*/
06101 0xfc5b, 0xbada, 0xfc5c, 0xfc5d, 0xc7ad, 0xfc5e, 0xfc5f, 0xfc60, /*0xd0-0xd7*/
06102 0xc4ac, 0xfc61, 0xfc62, 0xf7ec, 0xf7ed, 0xf7ee, 0xfc63, 0xf7f0, /*0xd8-0xdf*/
06103 0xf7ef, 0xfc64, 0xf7f1, 0xfc65, 0xfc66, 0xf7f4, 0xfc67, 0xf7f3, /*0xe0-0xe7*/
06104 0xfc68, 0xf7f2, 0xf7f5, 0xfc69, 0xfc6a, 0xfc6b, 0xfc6c, 0xf7f6, /*0xe8-0xef*/
06105 0xfc6d, 0xfc6e, 0xfc6f, 0xfc70, 0xfc71, 0xfc72, 0xfc73, 0xfc74, /*0xf0-0xf7*/
06106 0xfc75, 0xede9, 0xfc76, 0xede, 0xede, 0xfc77, 0xf6bc, 0xfc78, /*0xf8-0xff*/
06107 /* 0x9f00 */
06108 0xfc79, 0xfc7a, 0xfc7b, 0xfc7c, 0xfc7d, 0xfc7e, 0xfc80, 0xfc81, /*0x00-0x07*/
06109 0xfc82, 0xfc83, 0xfc84, 0xf6bd, 0xfc85, 0xf6be, 0xb6a6, 0xfc86, /*0x08-0x0f*/
06110 0xd8be, 0xfc87, 0xfc88, 0xb9c4, 0xfc89, 0xfc8a, 0xfc8b, 0xd8bb, /*0x10-0x17*/
06111 0xfc8c, 0xdcbl, 0xfc8d, 0xfc8e, 0xfc8f, 0xfc90, 0xfc91, 0xfc92, /*0x18-0x1f*/
06112 0xcaf3, 0xfc93, 0xf7f7, 0xfc94, 0xfc95, 0xfc96, 0xfc97, 0xfc98, /*0x20-0x27*/
06113 0xfc99, 0xfc9a, 0xfc9b, 0xfc9c, 0xf7f8, 0xfc9d, 0xfc9e, 0xf7f9, /*0x28-0x2f*/
06114 0xfc9f, 0xfca0, 0xfd40, 0xfd41, 0xfd42, 0xfd43, 0xfd44, 0xf7fb, /*0x30-0x37*/
06115 0xfd45, 0xf7fa, 0xfd46, 0xb1c7, 0xfd47, 0xf7fc, 0xf7fd, 0xfd48, /*0x38-0x3f*/
06116 0xfd49, 0xfd4a, 0xfd4b, 0xfd4c, 0xf7fe, 0xfd4d, 0xfd4e, 0xfd4f, /*0x40-0x47*/
06117 0xfd50, 0xfd51, 0xfd52, 0xfd53, 0xfd54, 0xfd55, 0xfd56, 0xfd57, /*0x48-0x4f*/
06118 0xc6eb, 0xecb4, 0xfd58, 0xfd59, 0xfd5a, 0xfd5b, 0xfd5c, 0xfd5d, /*0x50-0x57*/
06119 0xfd5e, 0xfd5f, 0xfd60, 0xfd61, 0xfd62, 0xfd63, 0xfd64, 0xfd65, /*0x58-0x5f*/
06120 0xfd66, 0xfd67, 0xfd68, 0xfd69, 0xfd6a, 0xfd6b, 0xfd6c, 0xfd6d, /*0x60-0x6f*/
06121 0xfd6e, 0xfd6f, 0xfd70, 0xfd71, 0xfd72, 0xfd73, 0xfd74, 0xfd75, /*0x68-0x6f*/
06122 0xfd76, 0xfd77, 0xfd78, 0xfd79, 0xfd7a, 0xfd7b, 0xfd7c, 0xfd7d, /*0x70-0x77*/
06123 0xfd7e, 0xfd80, 0xfd81, 0xfd82, 0xfd83, 0xfd84, 0xfd85, 0xb3dd, /*0x78-0x7f*/
06124 0xf6b3, 0xfd86, 0xfd87, 0xf6b4, 0xc1e4, 0xf6b5, 0xf6b6, 0xf6b7, /*0x80-0x87*/
06125 0xf6b8, 0xf6b9, 0xf6ba, 0xc8a3, 0xf6bb, 0xfd88, 0xfd89, 0xfd8a, /*0x88-0x8f*/
06126 0xfd8b, 0xfd8c, 0xfd8d, 0xfd8e, 0xfd8f, 0xfd90, 0xfd91, 0xfd92, /*0x90-0x97*/
06127 0xfd93, 0xc1fa, 0xb9a8, 0xede8, 0xfd94, 0xfd95, 0xfd96, 0xb9ea, /*0x98-0x9f*/
06128 0xd9df, 0xfd97, 0xfd98, 0xfd99, 0xfd9a, 0x0000, 0x0000, 0x0000, /*0xa0-0xaf*/
06129 };
06130 static const unsigned short cp936ext_page1f2f[32] = {
06131 0x0000, 0xfd9d, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x78-0x7f*/
06132 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x80-0x87*/
06133 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x88-0x8f*/
06134 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x90-0x97*/
06135 };
06136 static const unsigned short cp936ext_page1f3c[24] = {
06137 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xfd9f, /*0xe0-0xe7*/
06138 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xe8-0xef*/
06139 0x0000, 0xfda0, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0xf0-0xf7*/
06140 };
06141 static const unsigned short cp936ext_page1f41[40] = {
06142 0x0000, 0x0000, 0x0000, 0x0000, 0xfe40, 0xfe41, 0xfe42, 0xfe43, /*0x08-0x0f*/
06143 0x0000, 0xfe44, 0x0000, 0xfe45, 0xfe46, 0x0000, 0x0000, 0x0000, /*0x10-0x17*/
06144 0xfe47, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xfe48, /*0x18-0x1f*/
06145 0xfe49, 0xfe4a, 0x0000, 0xfe4b, 0xfe4c, 0x0000, 0x0000, 0xfe4d, /*0x20-0x27*/
06146 0xfe4e, 0xfe4f, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, /*0x28-0x2f*/
06147 };
06148 static const unsigned short cp936ext_page1fc6[64] = {
06149 0xa955, 0xa6f2, 0x0000, 0xa6f4, 0xa6f5, 0xa6e0, 0xa6e1, 0xa6f0, /*0x30-0x37*/
06150 0xa6f1, 0xa6e2, 0xa6e3, 0xa6ee, 0xa6ef, 0xa6e6, 0xa6e7, 0xa6e4, /*0x38-0x3f*/
06151 0xa6e5, 0xa6e8, 0xa6e9, 0xa6ea, 0xa6eb, 0x0000, 0x0000, 0x0000, /*0x40-0x47*/
06152 0x0000, 0xa968, 0xa969, 0xa96a, 0xa96b, 0xa96c, 0xa96d, 0xa96e, /*0x48-0x4f*/
06153 0xa96f, 0xa970, 0xa971, 0x0000, 0xa972, 0xa973, 0xa974, 0xa975, /*0x50-0x57*/
06154 0x0000, 0xa976, 0xa977, 0xa978, 0xa979, 0xa97a, 0xa97b, 0xa97c, /*0x58-0x5f*/
06155 0xa97d, 0xa97e, 0xa980, 0xa981, 0xa982, 0xa983, 0xa984, 0x0000, /*0x60-0x6f*/
06156 0xa985, 0xa986, 0xa987, 0xa988, 0x0000, 0x0000, 0x0000, 0x0000, /*0x68-0x6f*/
06157 };
06158 static const unsigned short cp936ext_page1fe0[96] = {
06159 0x0000, 0xa3a1, 0xa3a2, 0xa3a3, 0xa1e7, 0xa3a5, 0xa3a6, 0xa3a7, /*0x00-0x07*/
06160 0xa3a8, 0xa3a9, 0xa3aa, 0xa3ab, 0xa3ac, 0xa3ad, 0xa3ae, 0xa3af, /*0x08-0x0f*/
06161 0xa3b0, 0xa3b1, 0xa3b2, 0xa3b3, 0xa3b4, 0xa3b5, 0xa3b6, 0xa3b7, /*0x10-0x17*/
06162 0xa3b8, 0xa3b9, 0xa3ba, 0xa3bb, 0xa3bc, 0xa3bd, 0xa3be, 0xa3bf, /*0x18-0x1f*/
06163 0xa3c0, 0xa3c1, 0xa3c2, 0xa3c3, 0xa3c4, 0xa3c5, 0xa3c6, 0xa3c7, /*0x20-0x27*/
06164 0xa3c8, 0xa3c9, 0xa3ca, 0xa3cb, 0xa3cc, 0xa3cd, 0xa3ce, 0xa3cf, /*0x28-0x2f*/
06165 0xa3d0, 0xa3d1, 0xa3d2, 0xa3d3, 0xa3d4, 0xa3d5, 0xa3d6, 0xa3d7, /*0x30-0x37*/
06166 0xa3d8, 0xa3d9, 0xa3da, 0xa3db, 0xa3dc, 0xa3dd, 0xa3de, 0xa3df, /*0x38-0x3f*/
06167 0xa3e0, 0xa3e1, 0xa3e2, 0xa3e3, 0xa3e4, 0xa3e5, 0xa3e6, 0xa3e7, /*0x40-0x47*/
06168 0xa3e8, 0xa3e9, 0xa3ea, 0xa3eb, 0xa3ec, 0xa3ed, 0xa3ee, 0xa3ef, /*0x48-0x4f*/
06169 0xa3f0, 0xa3f1, 0xa3f2, 0xa3f3, 0xa3f4, 0xa3f5, 0xa3f6, 0xa3f7, /*0x50-0x57*/
06170 0xa3f8, 0xa3f9, 0xa3fa, 0xa3fb, 0xa3fc, 0xa3fd, 0xalab, 0x0000, /*0x58-0x5f*/
06171 };
06172 static const unsigned short cp936ext_page1ffc[8] = {
06173 0xale9, 0xalea, 0xa956, 0xa3fe, 0xa957, 0xa3a4, 0x0000, 0x0000, /*0xe0-0xe7*/
06174 };
06175
06176 static int
06177 cp936ext_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
06178 {
06179     if (n >= 2) {
06180         unsigned short c = 0;
06181         if (wc >= 0x00a0 && wc < 0x0170)
06182             c = cp936ext_page0014[wc-0x00a0];

```

```

06183     else if (wc >= 0x01c8 && wc < 0x01e0)
06184         c = cp936ext_page0039[wc-0x01c8];
06185     else if (wc >= 0x0250 && wc < 0x0268)
06186         c = cp936ext_page004a[wc-0x0250];
06187     else if (wc >= 0x02c0 && wc < 0x02e0)
06188         c = cp936ext_page0058[wc-0x02c0];
06189     else if (wc >= 0x0390 && wc < 0x03d0)
06190         c = cp936ext_page0072[wc-0x0390];
06191     else if (wc >= 0x0400 && wc < 0x0458)
06192         c = cp936ext_page0080[wc-0x0400];
06193     else if (wc >= 0x2010 && wc < 0x2040)
06194         c = cp936ext_page0402[wc-0x2010];
06195     else if (wc >= 0x2100 && wc < 0x21a0)
06196         c = cp936ext_page0420[wc-0x2100];
06197     else if (wc >= 0x2208 && wc < 0x22c0)
06198         c = cp936ext_page0441[wc-0x2208];
06199     else if (wc == 0x2312)
06200         c = 0xald0;
06201     else if (wc >= 0x2460 && wc < 0x24a0)
06202         c = cp936ext_page048c[wc-0x2460];
06203     else if (wc >= 0x2500 && wc < 0x25e8)
06204         c = cp936ext_page04a0[wc-0x2500];
06205     else if (wc >= 0x2600 && wc < 0x2648)
06206         c = cp936ext_page04c0[wc-0x2600];
06207     else if (wc >= 0x3000 && wc < 0x3130)
06208         c = cp936ext_page0600[wc-0x3000];
06209     else if (wc >= 0x3220 && wc < 0x3238)
06210         c = cp936ext_page0644[wc-0x3220];
06211     else if (wc == 0x32a3)
06212         c = 0xa949;
06213     else if (wc >= 0x3388 && wc < 0x33d8)
06214         c = cp936ext_page0671[wc-0x3388];
06215     else if (wc >= 0x4e00 && wc < 0x9fa8)
06216         c = cp936ext_page09c0[wc-0x4e00];
06217     else if (wc == 0xf92c)
06218         c = 0xfd9c;
06219     else if (wc >= 0xf978 && wc < 0xf998)
06220         c = cp936ext_page1f2f[wc-0xf978];
06221     else if (wc >= 0xf9e0 && wc < 0xf9f8)
06222         c = cp936ext_page1f3c[wc-0xf9e0];
06223     else if (wc >= 0xfa08 && wc < 0xfa30)
06224         c = cp936ext_page1f41[wc-0xfa08];
06225     else if (wc >= 0xfe30 && wc < 0xfe70)
06226         c = cp936ext_page1fc6[wc-0xfe30];
06227     else if (wc >= 0xff00 && wc < 0xff60)
06228         c = cp936ext_page1fe0[wc-0xff00];
06229     else if (wc >= 0xffe0 && wc < 0xffe8)
06230         c = cp936ext_page1ffc[wc-0xffe0];
06231     if (c != 0) {
06232         r[0] = (c >> 8); r[1] = (c & 0xff);
06233         return 2;
06234     }
06235     return RET_ILSEQ;
06236 }
06237 return RET_TOOSMALL;
06238 }
06239 #endif /* NEED_TOMB */
06240
06241 #endif /* CP936 */

```

12.266 gb2312.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/gb2312.h,v 1.5 2003/05/27 22:26:29 tsi Exp $ */
00002
00003 /*
00004  * GB2312.1980-0
00005  */
00006 #ifdef NEED_TOWC
00007 static const unsigned short gb2312_2uni_page21[831] = {
00008     /* 0x21 */
00009     0x3000, 0x3001, 0x3002, 0x30fb, 0x02c9, 0x02c7, 0x00a8, 0x3003,
00010     0x3005, 0x2015, 0xff5e, 0x2016, 0x2026, 0x2018, 0x2019, 0x201c,
00011     0x201d, 0x3014, 0x3015, 0x3008, 0x3009, 0x300a, 0x300b, 0x300c,
00012     0x300d, 0x300e, 0x300f, 0x3016, 0x3017, 0x3010, 0x3011, 0x00b1,
00013     0x00d7, 0x00f7, 0x2236, 0x2227, 0x2228, 0x2211, 0x220f, 0x222a,
00014     0x2229, 0x2208, 0x2237, 0x221a, 0x22a5, 0x2225, 0x2220, 0x2312,
00015     0x2299, 0x222b, 0x222e, 0x2261, 0x224c, 0x2248, 0x223d, 0x221d,
00016     0x2260, 0x226e, 0x226f, 0x2264, 0x2265, 0x221e, 0x2235, 0x2234,
00017     0x2642, 0x2640, 0x00b0, 0x2032, 0x2033, 0x2103, 0xff04, 0x00a4,
00018     0xffe0, 0xffe1, 0x2030, 0x00a7, 0x2116, 0x2606, 0x2605, 0x25cb,
00019     0x25cf, 0x25ce, 0x25c7, 0x25c6, 0x25a1, 0x25a0, 0x25b3, 0x25b2,
00020     0x203b, 0x2192, 0x2190, 0x2191, 0x2193, 0x3013,
00021     /* 0x22 */
00022     0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00023     0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,

```

```
00024 0x2488, 0x2489, 0x248a, 0x248b, 0x248c, 0x248d, 0x248e, 0x248f,
00025 0x2490, 0x2491, 0x2492, 0x2493, 0x2494, 0x2495, 0x2496, 0x2497,
00026 0x2498, 0x2499, 0x249a, 0x249b, 0x249c, 0x249d, 0x249e, 0x249f,
00027 0x2478, 0x2479, 0x247a, 0x247b, 0x247c, 0x247d, 0x247e, 0x247f,
00028 0x2480, 0x2481, 0x2482, 0x2483, 0x2484, 0x2485, 0x2486, 0x2487,
00029 0x2460, 0x2461, 0x2462, 0x2463, 0x2464, 0x2465, 0x2466, 0x2467,
00030 0x2468, 0x2469, 0xffffd, 0xffffd, 0x3220, 0x3221, 0x3222, 0x3223,
00031 0x3224, 0x3225, 0x3226, 0x3227, 0x3228, 0x3229, 0xffffd, 0xffffd,
00032 0x2160, 0x2161, 0x2162, 0x2163, 0x2164, 0x2165, 0x2166, 0x2167,
00033 0x2168, 0x2169, 0x216a, 0x216b, 0xffffd, 0xffffd,
00034 /* 0x23 */
00035 0xff01, 0xff02, 0xff03, 0xff05, 0xff05, 0xff06, 0xff07, 0xff08,
00036 0xff09, 0xff0a, 0xff0b, 0xff0c, 0xff0d, 0xff0e, 0xff0f, 0xff10,
00037 0xff11, 0xff12, 0xff13, 0xff14, 0xff15, 0xff16, 0xff17, 0xff18,
00038 0xff19, 0xff1a, 0xff1b, 0xff1c, 0xff1d, 0xff1e, 0xff1f, 0xff20,
00039 0xff21, 0xff22, 0xff23, 0xff24, 0xff25, 0xff26, 0xff27, 0xff28,
00040 0xff29, 0xff2a, 0xff2b, 0xff2c, 0xff2d, 0xff2e, 0xff2f, 0xff30,
00041 0xff31, 0xff32, 0xff33, 0xff34, 0xff35, 0xff36, 0xff37, 0xff38,
00042 0xff39, 0xff3a, 0xff3b, 0xff3c, 0xff3d, 0xff3e, 0xff3f, 0xff40,
00043 0xff41, 0xff42, 0xff43, 0xff44, 0xff45, 0xff46, 0xff47, 0xff48,
00044 0xff49, 0xff4a, 0xff4b, 0xff4c, 0xff4d, 0xff4e, 0xff4f, 0xff50,
00045 0xff51, 0xff52, 0xff53, 0xff54, 0xff55, 0xff56, 0xff57, 0xff58,
00046 0xff59, 0xff5a, 0xff5b, 0xff5c, 0xff5d, 0xff5e,
00047 /* 0x24 */
00048 0x3041, 0x3042, 0x3043, 0x3044, 0x3045, 0x3046, 0x3047, 0x3048,
00049 0x3049, 0x304a, 0x304b, 0x304c, 0x304d, 0x304e, 0x304f, 0x3050,
00050 0x3051, 0x3052, 0x3053, 0x3054, 0x3055, 0x3056, 0x3057, 0x3058,
00051 0x3059, 0x305a, 0x305b, 0x305c, 0x305d, 0x305e, 0x305f, 0x3060,
00052 0x3061, 0x3062, 0x3063, 0x3064, 0x3065, 0x3066, 0x3067, 0x3068,
00053 0x3069, 0x306a, 0x306b, 0x306c, 0x306d, 0x306e, 0x306f, 0x3070,
00054 0x3071, 0x3072, 0x3073, 0x3074, 0x3075, 0x3076, 0x3077, 0x3078,
00055 0x3079, 0x307a, 0x307b, 0x307c, 0x307d, 0x307e, 0x307f, 0x3080,
00056 0x3081, 0x3082, 0x3083, 0x3084, 0x3085, 0x3086, 0x3087, 0x3088,
00057 0x3089, 0x308a, 0x308b, 0x308c, 0x308d, 0x308e, 0x308f, 0x3090,
00058 0x3091, 0x3092, 0x3093, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00059 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00060 /* 0x25 */
00061 0x30a1, 0x30a2, 0x30a3, 0x30a4, 0x30a5, 0x30a6, 0x30a7, 0x30a8,
00062 0x30a9, 0x30aa, 0x30ab, 0x30ac, 0x30ad, 0x30ae, 0x30af, 0x30b0,
00063 0x30b1, 0x30b2, 0x30b3, 0x30b4, 0x30b5, 0x30b6, 0x30b7, 0x30b8,
00064 0x30b9, 0x30ba, 0x30bb, 0x30bc, 0x30bd, 0x30be, 0x30bf, 0x30c0,
00065 0x30c1, 0x30c2, 0x30c3, 0x30c4, 0x30c5, 0x30c6, 0x30c7, 0x30c8,
00066 0x30c9, 0x30ca, 0x30cb, 0x30cc, 0x30cd, 0x30ce, 0x30cf, 0x30d0,
00067 0x30d1, 0x30d2, 0x30d3, 0x30d4, 0x30d5, 0x30d6, 0x30d7, 0x30d8,
00068 0x30d9, 0x30da, 0x30db, 0x30dc, 0x30dd, 0x30de, 0x30df, 0x30e0,
00069 0x30e1, 0x30e2, 0x30e3, 0x30e4, 0x30e5, 0x30e6, 0x30e7, 0x30e8,
00070 0x30e9, 0x30ea, 0x30eb, 0x30ec, 0x30ed, 0x30ee, 0x30ef, 0x30f0,
00071 0x30f1, 0x30f2, 0x30f3, 0x30f4, 0x30f5, 0x30f6, 0xffffd, 0xffffd,
00072 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00073 /* 0x26 */
00074 0x0391, 0x0392, 0x0393, 0x0394, 0x0395, 0x0396, 0x0397, 0x0398,
00075 0x0399, 0x039a, 0x039b, 0x039c, 0x039d, 0x039e, 0x039f, 0x03a0,
00076 0x03a1, 0x03a2, 0x03a3, 0x03a4, 0x03a5, 0x03a6, 0x03a7, 0x03a8, 0x03a9,
00077 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00078 0x03b1, 0x03b2, 0x03b3, 0x03b4, 0x03b5, 0x03b6, 0x03b7, 0x03b8,
00079 0x03b9, 0x03ba, 0x03bb, 0x03bc, 0x03bd, 0x03be, 0x03bf, 0x03c0,
00080 0x03c1, 0x03c2, 0x03c3, 0x03c4, 0x03c5, 0x03c6, 0x03c7, 0x03c8,
00081 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00082 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00083 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00084 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00085 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00086 /* 0x27 */
00087 0x0410, 0x0411, 0x0412, 0x0413, 0x0414, 0x0415, 0x0416, 0x0417,
00088 0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e, 0x041f,
00089 0x0420, 0x0421, 0x0422, 0x0423, 0x0424, 0x0425, 0x0426, 0x0427,
00090 0x0428, 0x0429, 0x042a, 0x042b, 0x042c, 0x042d, 0x042e, 0x042f,
00091 0x0430, 0x0431, 0x0432, 0x0433, 0x0434, 0x0435, 0x0436, 0x0437,
00092 0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e, 0x043f,
00093 0x0440, 0x0441, 0x0442, 0x0443, 0x0444, 0x0445, 0x0446, 0x0447,
00094 0x0448, 0x0449, 0x044a, 0x044b, 0x044c, 0x044d, 0x044e, 0x044f,
00095 0x0450, 0x0451, 0x0452, 0x0453, 0x0454, 0x0455, 0x0456, 0x0457,
00096 0x0458, 0x0459, 0x045a, 0x045b, 0x045c, 0x045d, 0x045e, 0x045f,
00097 0x0460, 0x0461, 0x0462, 0x0463, 0x0464, 0x0465, 0x0466, 0x0467,
00098 0x0468, 0x0469, 0x046a, 0x046b, 0x046c, 0x046d, 0x046e, 0x046f,
00099 /* 0x28 */
00100 0x0101, 0x00e1, 0x01ce, 0x00e0, 0x0113, 0x00e9, 0x011b, 0x00e8,
00101 0x012b, 0x00ed, 0x01d0, 0x00ec, 0x014d, 0x00f3, 0x01d2, 0x00f2,
00102 0x016b, 0x00fa, 0x01d4, 0x00f9, 0x01d6, 0x01d8, 0x01da, 0x01dc,
00103 0x00fc, 0x00ea, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00104 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0x3105, 0x3106, 0x3107, 0x3108,
00105 0x3109, 0x310a, 0x310b, 0x310c, 0x310d, 0x310e, 0x310f, 0x3110,
00106 0x3111, 0x3112, 0x3113, 0x3114, 0x3115, 0x3116, 0x3117, 0x3118,
00107 0x3119, 0x311a, 0x311b, 0x311c, 0x311d, 0x311e, 0x311f, 0x3120,
00108 0x3121, 0x3122, 0x3123, 0x3124, 0x3125, 0x3126, 0x3127, 0x3128,
00109 0x3129, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00110 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
```



```
00111 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00112 /* 0x29 */
00113 0xffffd, 0xffffd, 0xffffd, 0x2500, 0x2501, 0x2502, 0x2503, 0x2504,
00114 0x2505, 0x2506, 0x2507, 0x2508, 0x2509, 0x250a, 0x250b, 0x250c,
00115 0x250d, 0x250e, 0x250f, 0x2510, 0x2511, 0x2512, 0x2513, 0x2514,
00116 0x2515, 0x2516, 0x2517, 0x2518, 0x2519, 0x251a, 0x251b, 0x251c,
00117 0x251d, 0x251e, 0x251f, 0x2520, 0x2521, 0x2522, 0x2523, 0x2524,
00118 0x2525, 0x2526, 0x2527, 0x2528, 0x2529, 0x252a, 0x252b, 0x252c,
00119 0x252d, 0x252e, 0x252f, 0x2530, 0x2531, 0x2532, 0x2533, 0x2534,
00120 0x2535, 0x2536, 0x2537, 0x2538, 0x2539, 0x253a, 0x253b, 0x253c,
00121 0x253d, 0x253e, 0x253f, 0x2540, 0x2541, 0x2542, 0x2543, 0x2544,
00122 0x2545, 0x2546, 0x2547, 0x2548, 0x2549, 0x254a, 0x254b,
00123 };
00124 static const unsigned short gb2312_2uni_page30[6768] = {
00125 /* 0x30 */
00126 0x554a, 0x963f, 0x57c3, 0x6328, 0x54ce, 0x5509, 0x54c0, 0x7691,
00127 0x764c, 0x853c, 0x77ee, 0x827e, 0x788d, 0x7231, 0x9698, 0x978d,
00128 0x6c28, 0x5b89, 0x4ffa, 0x6309, 0x6697, 0x5cb8, 0x80fa, 0x6848,
00129 0x80ae, 0x6602, 0x76ce, 0x51f9, 0x6556, 0x71ac, 0x7ff1, 0x8884,
00130 0x50b2, 0x5965, 0x61ca, 0x6fb3, 0x82ad, 0x634c, 0x6252, 0x53ed,
00131 0x5427, 0x7b06, 0x516b, 0x75a4, 0x5df4, 0x62d4, 0x8dcb, 0x9776,
00132 0x628a, 0x8019, 0x575d, 0x9738, 0x7f62, 0x7238, 0x767d, 0x67cf,
00133 0x767e, 0x6446, 0x4f70, 0x8d25, 0x62dc, 0x7a17, 0x6591, 0x73ed,
00134 0x642c, 0x6273, 0x822c, 0x9881, 0x677f, 0x7248, 0x626e, 0x62cc,
00135 0x4f34, 0x74e3, 0x534a, 0x529e, 0x7eca, 0x90a6, 0x5e2e, 0x6886,
00136 0x699c, 0x8180, 0x7ed1, 0x68d2, 0x78c5, 0x868c, 0x9551, 0x508d,
00137 0x8c24, 0x82de, 0x80de, 0x5305, 0x8912, 0x5265,
00138 /* 0x31 */
00139 0x8584, 0x96f9, 0x4fdd, 0x5821, 0x9971, 0x5b9d, 0x62b1, 0x62a5,
00140 0x66b4, 0x8c79, 0x9c8d, 0x7206, 0x676f, 0x7891, 0x60b2, 0x5351,
00141 0x5317, 0x8f88, 0x80cc, 0x8d1d, 0x94a1, 0x500d, 0x72c8, 0x5907,
00142 0x60eb, 0x7119, 0x88ab, 0x5954, 0x82ef, 0x672c, 0x7b28, 0x5d29,
00143 0x7ef7, 0x752d, 0x6cf5, 0x8e66, 0x8ff8, 0x903c, 0x9f3b, 0x6bd4,
00144 0x9119, 0x7b14, 0x5f7c, 0x78a7, 0x84d6, 0x853d, 0x6bd5, 0x6bd9,
00145 0x6bd6, 0x5e01, 0x5e87, 0x75f9, 0x95ed, 0x655d, 0x5f0a, 0x5fc5,
00146 0x8f9f, 0x58c1, 0x81c2, 0x907f, 0x965b, 0x97ad, 0x8fb9, 0x7f16,
00147 0x8d2c, 0x6241, 0x4fbf, 0x53d8, 0x535e, 0x8fa8, 0x8fa9, 0x8fab,
00148 0x904d, 0x6807, 0x5f6a, 0x8198, 0x8868, 0x9cd6, 0x618b, 0x522b,
00149 0x762a, 0x5f6c, 0x658c, 0x6fd2, 0x6ee8, 0x5bbe, 0x6448, 0x5175,
00150 0x51b0, 0x67c4, 0x4e19, 0x79c9, 0x997c, 0x70b3,
00151 /* 0x32 */
00152 0x75c5, 0x5e76, 0x73bb, 0x83e0, 0x64ad, 0x62e8, 0x94b5, 0x6ce2,
00153 0x535a, 0x52c3, 0x640f, 0x94c2, 0x7b94, 0x4f2f, 0x5e1b, 0x8236,
00154 0x8116, 0x818a, 0x6e24, 0x6cca, 0x9a73, 0x6355, 0x535c, 0x54fa,
00155 0x8865, 0x57e0, 0x4e0d, 0x5e03, 0x6b65, 0x7c3f, 0x90e8, 0x6016,
00156 0x64e6, 0x731c, 0x88c1, 0x6750, 0x624d, 0x8d22, 0x776c, 0x8e29,
00157 0x91c7, 0x5f69, 0x83dc, 0x8521, 0x9910, 0x53c2, 0x8695, 0x6b8b,
00158 0x60ed, 0x60e8, 0x707f, 0x82cd, 0x8231, 0x4ed3, 0x6ca7, 0x85cf,
00159 0x64cd, 0x7cd9, 0x69fd, 0x66f9, 0x8349, 0x5395, 0x7b56, 0x4fa7,
00160 0x518c, 0x6d4b, 0x5c42, 0x8e6d, 0x63d2, 0x53c9, 0x832c, 0x8336,
00161 0x67e5, 0x78b4, 0x643d, 0x5bdf, 0x5c94, 0x5dee, 0x8be7, 0x62c6,
00162 0x67f4, 0x8c7a, 0x6400, 0x63ba, 0x8749, 0x998b, 0x8c17, 0x7f20,
00163 0x94f2, 0x4ea7, 0x9610, 0x98a4, 0x660c, 0x7316,
00164 /* 0x33 */
00165 0x573a, 0x5c1d, 0x5e38, 0x957f, 0x507f, 0x80a0, 0x5382, 0x655e,
00166 0x7545, 0x5531, 0x5021, 0x8d85, 0x6284, 0x949e, 0x671d, 0x5632,
00167 0x6f6e, 0x57de, 0x5435, 0x7092, 0x8f66, 0x626f, 0x64a4, 0x63a3,
00168 0x5f7b, 0x6f88, 0x90f4, 0x81e3, 0x8fb0, 0x5c18, 0x6668, 0x5ff1,
00169 0x6c89, 0x9648, 0x8d81, 0x886c, 0x6491, 0x79f0, 0x57ce, 0x6a59,
00170 0x6210, 0x5448, 0x4e58, 0x7a0b, 0x60e9, 0x6f84, 0x8bda, 0x627f,
00171 0x901e, 0x9a8b, 0x79e4, 0x5403, 0x75f4, 0x6301, 0x5319, 0x6c60,
00172 0x8f8f, 0x5f1b, 0x9a70, 0x803b, 0x9f7f, 0x4f88, 0x5c3a, 0x8d64,
00173 0x7fc5, 0x65a5, 0x70bd, 0x5145, 0x51b2, 0x866b, 0x5d07, 0x5ba0,
00174 0x62bd, 0x916c, 0x7574, 0x8e0c, 0x7a20, 0x6101, 0x7b79, 0x4ec7,
00175 0x7ef8, 0x7785, 0x4e11, 0x81ed, 0x521d, 0x51fa, 0x6a71, 0x53a8,
00176 0x8e87, 0x9504, 0x96cf, 0x6ec1, 0x9664, 0x695a,
00177 /* 0x34 */
00178 0x7840, 0x50a8, 0x77d7, 0x6410, 0x89e6, 0x5904, 0x63e3, 0x5ddd,
00179 0x7a7f, 0x693d, 0x4f20, 0x8239, 0x5598, 0x4e32, 0x75ae, 0x7a97,
00180 0x5e62, 0x5e8a, 0x95ef, 0x521b, 0x5439, 0x708a, 0x6376, 0x9524,
00181 0x5782, 0x6625, 0x693f, 0x9187, 0x5507, 0x6df3, 0x7eaf, 0x8822,
00182 0x6233, 0x7ef0, 0x75b5, 0x8328, 0x78c1, 0x96cc, 0x8f9e, 0x6148,
00183 0x74f7, 0x8bcd, 0x6b64, 0x523a, 0x8d50, 0x6b21, 0x806a, 0x8471,
00184 0x56f1, 0x5306, 0x4ece, 0x4e1b, 0x51d1, 0x7c97, 0x918b, 0x7c07,
00185 0x4fc3, 0x8e7f, 0x7be1, 0x7a9c, 0x6467, 0x5d14, 0x50ac, 0x8106,
00186 0x7601, 0x7cb9, 0x6dec, 0x7fe0, 0x6751, 0x5b58, 0x5bf8, 0x78cb,
00187 0x64ae, 0x6413, 0x63aa, 0x632b, 0x9519, 0x642d, 0x8f8e, 0x7b54,
00188 0x7629, 0x6253, 0x5927, 0x5446, 0x6b79, 0x50a3, 0x6234, 0x5e26,
00189 0x6b86, 0x4ee3, 0x8d37, 0x888b, 0x5f85, 0x902e,
00190 /* 0x35 */
00191 0x6020, 0x803d, 0x62c5, 0x4e39, 0x5355, 0x90f8, 0x63b8, 0x80c6,
00192 0x65e6, 0x6c2e, 0x4f46, 0x60ee, 0x6de1, 0x8bde, 0x5f39, 0x86cb,
00193 0x5f53, 0x6321, 0x515a, 0x8361, 0x6863, 0x5200, 0x6363, 0x8e48,
00194 0x5012, 0x5c9b, 0x7977, 0x5bfc, 0x5230, 0x7a3b, 0x60bc, 0x9053,
00195 0x76d7, 0x5fb7, 0x5f97, 0x7684, 0x8e6c, 0x706f, 0x767b, 0x7b49,
00196 0x77aa, 0x51f3, 0x9093, 0x5824, 0x4f4e, 0x6ef4, 0x8fea, 0x654c,
00197 0x7b1b, 0x72c4, 0x6da4, 0x7fdf, 0x5ae1, 0x62b5, 0x5e95, 0x5730,
```

```

00198 0x8482, 0x7b2c, 0x5e1d, 0x5f1f, 0x9012, 0x7f14, 0x98a0, 0x6382,
00199 0x6ec7, 0x7898, 0x70b9, 0x5178, 0x975b, 0x57ab, 0x7535, 0x4f43,
00200 0x7538, 0x5e97, 0x60e6, 0x5960, 0x6dc0, 0x6bbf, 0x7889, 0x53fc,
00201 0x96d5, 0x51cb, 0x5201, 0x6389, 0x540a, 0x9493, 0x8c03, 0x8dcc,
00202 0x7239, 0x789f, 0x8776, 0x8fed, 0x8c0d, 0x53e0,
00203 /* 0x36 */
00204 0x4e01, 0x76ef, 0x53ee, 0x9489, 0x9876, 0x9f0e, 0x952d, 0x5b9a,
00205 0x8ba2, 0x4e22, 0x4e1c, 0x51ac, 0x8463, 0x61c2, 0x52a8, 0x680b,
00206 0x4f97, 0x606b, 0x51bb, 0x6d1e, 0x515c, 0x6296, 0x6597, 0x9661,
00207 0x8c46, 0x9017, 0x75d8, 0x90fd, 0x7763, 0x6bd2, 0x728a, 0x72ec,
00208 0x8bfb, 0x5835, 0x7779, 0x8d4c, 0x675c, 0x9540, 0x809a, 0x5ea6,
00209 0x6e21, 0x5992, 0x7aef, 0x77ed, 0x953b, 0x6bb5, 0x65ad, 0x7f0e,
00210 0x5806, 0x5151, 0x961f, 0x5bf9, 0x58a9, 0x5428, 0x8e72, 0x6566,
00211 0x987f, 0x56e4, 0x949d, 0x76fe, 0x9041, 0x6387, 0x54c6, 0x591a,
00212 0x593a, 0x579b, 0x8eb2, 0x6735, 0x8dfa, 0x8235, 0x5241, 0x60f0,
00213 0x5815, 0x86fe, 0x5ce8, 0x9e45, 0x4fc4, 0x989d, 0x8bb9, 0x5a25,
00214 0x6076, 0x5384, 0x627c, 0x904f, 0x9102, 0x997f, 0x6069, 0x800c,
00215 0x513f, 0x8033, 0x5c14, 0x9975, 0x6d31, 0x4e8c,
00216 /* 0x37 */
00217 0x8d30, 0x53d1, 0x7f5a, 0x7b4f, 0x4f10, 0x4e4f, 0x9600, 0x6cd5,
00218 0x73d0, 0x85e9, 0x5e06, 0x756a, 0x7ffb, 0x6a0a, 0x77fe, 0x9492,
00219 0x7e41, 0x51e1, 0x70e6, 0x53cd, 0x8fd4, 0x8303, 0x8d29, 0x72af,
00220 0x996d, 0x6cdb, 0x574a, 0x82b3, 0x65b9, 0x80aa, 0x623f, 0x9632,
00221 0x59a8, 0x4eff, 0x8bbf, 0x7eba, 0x653e, 0x83f2, 0x975e, 0x5561,
00222 0x98de, 0x80a5, 0x532a, 0x8bfd, 0x5420, 0x80ba, 0x5e9f, 0x6cb8,
00223 0x8d39, 0x82ac, 0x915a, 0x5429, 0x6c1b, 0x5206, 0x7eb7, 0x575f,
00224 0x711a, 0x6c7e, 0x7c89, 0x594b, 0x4efd, 0x5fff, 0x6124, 0x7caa,
00225 0x4e30, 0x5c01, 0x67ab, 0x8702, 0x5cf0, 0x950b, 0x98ce, 0x75af,
00226 0x70fd, 0x9022, 0x51af, 0x7f1d, 0x8bbd, 0x5949, 0x51e4, 0x4f5b,
00227 0x5426, 0x592b, 0x6577, 0x80a4, 0x5b75, 0x6276, 0x62c2, 0x8f90,
00228 0x5e45, 0x6c1f, 0x7b26, 0x4f0f, 0x4fd8, 0x670d,
00229 /* 0x38 */
00230 0x6d6e, 0x6daa, 0x798f, 0x88b1, 0x5f17, 0x752b, 0x629a, 0x8f85,
00231 0x4fef, 0x91dc, 0x65a7, 0x812f, 0x8151, 0x5e9c, 0x8150, 0x8d74,
00232 0x526f, 0x8986, 0x8d4b, 0x590d, 0x5085, 0x4ed8, 0x961c, 0x7236,
00233 0x8179, 0x8d1f, 0x5bcc, 0x8ba3, 0x9644, 0x5987, 0x7f1a, 0x5490,
00234 0x5676, 0x560e, 0x8be5, 0x6539, 0x6982, 0x9499, 0x76d6, 0x6e89,
00235 0x5e72, 0x7518, 0x6746, 0x67d1, 0x7aff, 0x809d, 0x8d76, 0x611f,
00236 0x79c6, 0x6562, 0x8d63, 0x5188, 0x521a, 0x94a2, 0x7f38, 0x809b,
00237 0x7eb2, 0x5c97, 0x6e2f, 0x6760, 0x7bd9, 0x768b, 0x9ad8, 0x818f,
00238 0x7f94, 0x7cd5, 0x641e, 0x9550, 0x7a3f, 0x544a, 0x54e5, 0x6b4c,
00239 0x6401, 0x6208, 0x9e3d, 0x80f3, 0x7599, 0x5272, 0x9769, 0x845b,
00240 0x683c, 0x86e4, 0x9601, 0x9694, 0x94ec, 0x4e2a, 0x5404, 0x7ed9,
00241 0x6839, 0x8ddf, 0x8015, 0x66f4, 0x5e9a, 0x7fb9,
00242 /* 0x39 */
00243 0x57c2, 0x803f, 0x6897, 0x5de5, 0x653b, 0x529f, 0x606d, 0x9f9a,
00244 0x4f9b, 0x8eac, 0x516c, 0x5bab, 0x5f13, 0x5de9, 0x6c5e, 0x62f1,
00245 0x8d21, 0x5171, 0x94a9, 0x52fe, 0x6c9f, 0x82df, 0x72d7, 0x57a2,
00246 0x6784, 0x8d2d, 0x591f, 0x8f9c, 0x83c7, 0x5495, 0x7b8d, 0x4f30,
00247 0x6cbd, 0x5b64, 0x59d1, 0x9f13, 0x53e4, 0x86ca, 0x9aa8, 0x8c37,
00248 0x80a1, 0x6545, 0x987e, 0x56fa, 0x96c7, 0x522e, 0x74dc, 0x5250,
00249 0x5be1, 0x6302, 0x8902, 0x4e56, 0x62d0, 0x602a, 0x68fa, 0x5173,
00250 0x5b98, 0x51a0, 0x89c2, 0x7ba1, 0x9986, 0x7f50, 0x60ef, 0x704c,
00251 0x8d2f, 0x5149, 0x5e7f, 0x901b, 0x7470, 0x89c4, 0x572d, 0x7845,
00252 0x5f52, 0x9f9f, 0x95fa, 0x8f68, 0x9b3c, 0x8be1, 0x7678, 0x6842,
00253 0x67dc, 0x8dea, 0x8d35, 0x523d, 0x8f8a, 0x6eda, 0x68cd, 0x9505,
00254 0x90ed, 0x56fd, 0x679c, 0x88f9, 0x8fc7, 0x54c8,
00255 /* 0x3a */
00256 0x9ab8, 0x5b69, 0x6d77, 0x6c26, 0x4ea5, 0x5bb3, 0x9a87, 0x9163,
00257 0x61a8, 0x90af, 0x97e9, 0x542b, 0x6db5, 0x5bd2, 0x51fd, 0x558a,
00258 0x7f55, 0x7ff0, 0x64bc, 0x634d, 0x65f1, 0x61be, 0x608d, 0x710a,
00259 0x6c57, 0x6c49, 0x592f, 0x676d, 0x822a, 0x58d5, 0x568e, 0x8c6a,
00260 0x6beb, 0x90dd, 0x597d, 0x8017, 0x53f7, 0x6d69, 0x5475, 0x559d,
00261 0x8377, 0x83cf, 0x6838, 0x79be, 0x548c, 0x4f55, 0x5408, 0x76d2,
00262 0x8c89, 0x9602, 0x6cb3, 0x6db8, 0x8d6b, 0x8910, 0x9e64, 0x8d3a,
00263 0x563f, 0x9ed1, 0x75d5, 0x5f88, 0x72e0, 0x6068, 0x54fc, 0x4ea8,
00264 0x6a2a, 0x8861, 0x6052, 0x8f70, 0x54c4, 0x70d8, 0x8679, 0x9e3f,
00265 0x6d2a, 0x5b8f, 0x5f18, 0x7ea2, 0x5589, 0x4faf, 0x7334, 0x543c,
00266 0x539a, 0x5019, 0x540e, 0x547c, 0x4e4e, 0x5ffd, 0x745a, 0x58f6,
00267 0x846b, 0x80e1, 0x8774, 0x72d0, 0x7cca, 0x6e56,
00268 /* 0x3b */
00269 0x5f27, 0x864e, 0x552c, 0x62a4, 0x4e92, 0x6caa, 0x6237, 0x82b1,
00270 0x54d7, 0x534e, 0x733e, 0x6ed1, 0x753b, 0x5212, 0x5316, 0x8bdd,
00271 0x69d0, 0x5f8a, 0x6000, 0x6dee, 0x574f, 0x6b22, 0x73af, 0x6853,
00272 0x8fd8, 0x7f13, 0x6362, 0x60a3, 0x5524, 0x75ea, 0x8c62, 0x7115,
00273 0x6da3, 0x5ba6, 0x5e7b, 0x8352, 0x614c, 0x9ec4, 0x78fa, 0x8757,
00274 0x7c27, 0x7687, 0x51f0, 0x60f6, 0x714c, 0x6643, 0x5e4c, 0x604d,
00275 0x8c0e, 0x7070, 0x6325, 0x8f89, 0x5fbd, 0x6062, 0x86d4, 0x56de,
00276 0x6bc1, 0x6094, 0x6167, 0x5349, 0x60e0, 0x6666, 0x8d3f, 0x79fd,
00277 0x4f1a, 0x70e9, 0x6c47, 0x8bb3, 0x8bf2, 0x7ed8, 0x8364, 0x660f,
00278 0x5a5a, 0x9b42, 0x6d51, 0x6df7, 0x8c41, 0x6d3b, 0x4f19, 0x706b,
00279 0x83b7, 0x6216, 0x60d1, 0x970d, 0x8d27, 0x7978, 0x51fb, 0x573e,
00280 0x57fa, 0x673a, 0x7578, 0x7a3d, 0x79ef, 0x7b95,
00281 /* 0x3c */
00282 0x808c, 0x9965, 0x8ff9, 0x6fc0, 0x8ba5, 0x9e21, 0x59ec, 0x7ee9,
00283 0x7f09, 0x5409, 0x6781, 0x68d8, 0x8f91, 0x7c4d, 0x96c6, 0x53ca,
00284 0x6025, 0x75be, 0x6c72, 0x5373, 0x5ac9, 0x7ea7, 0x6324, 0x51e0,

```

```
00285 0x810a, 0x5df1, 0x84df, 0x6280, 0x5180, 0x5b63, 0x4f0e, 0x796d,
00286 0x5242, 0x60b8, 0x6d4e, 0x5bc4, 0x5bc2, 0x8ba1, 0x8bb0, 0x65e2,
00287 0x5fcc, 0x9645, 0x5993, 0x7ee7, 0x7eaa, 0x5609, 0x67b7, 0x5939,
00288 0x4f73, 0x5bb6, 0x52a0, 0x835a, 0x988a, 0x8d3e, 0x7532, 0x94be,
00289 0x5047, 0x7a3c, 0x4ef7, 0x67b6, 0x9a7e, 0x5ac1, 0x6b7c, 0x76d1,
00290 0x575a, 0x5c16, 0x7b3a, 0x95f4, 0x714e, 0x517c, 0x80a9, 0x8270,
00291 0x5978, 0x7f04, 0x8327, 0x68c0, 0x67ec, 0x78b1, 0x7877, 0x62e3,
00292 0x6361, 0x7b80, 0x4fed, 0x526a, 0x51cf, 0x8350, 0x69db, 0x9274,
00293 0x8df5, 0x8d31, 0x89c1, 0x952e, 0x7bad, 0x4ef6,
00294 /* 0x3d */
00295 0x5065, 0x8230, 0x5251, 0x996f, 0x6e10, 0x6e85, 0x6da7, 0x5efa,
00296 0x50f5, 0x59dc, 0x5c06, 0x6d46, 0x6c5f, 0x7586, 0x848b, 0x6868,
00297 0x5956, 0x8bb2, 0x5320, 0x9171, 0x964d, 0x8549, 0x6912, 0x7901,
00298 0x7126, 0x80f6, 0x4ea4, 0x90ca, 0x6d47, 0x9a84, 0x5a07, 0x56bc,
00299 0x6405, 0x94f0, 0x77eb, 0x4fa5, 0x811a, 0x72e1, 0x89d2, 0x997a,
00300 0x7f34, 0x7ede, 0x527f, 0x6559, 0x9175, 0x8f7f, 0x8f83, 0x53eb,
00301 0x7a96, 0x63ed, 0x63a5, 0x7686, 0x79f8, 0x8857, 0x9636, 0x622a,
00302 0x52ab, 0x8282, 0x6854, 0x6770, 0x6377, 0x776b, 0x7aed, 0x6d01,
00303 0x7ed3, 0x89e3, 0x59d0, 0x6212, 0x85c9, 0x82a5, 0x754c, 0x501f,
00304 0x4ecb, 0x75a5, 0x8beb, 0x5c4a, 0x5dfc, 0x7b4b, 0x65a4, 0x91d1,
00305 0x4eca, 0x6d25, 0x895f, 0x7d27, 0x9526, 0x4ec5, 0x8c28, 0x8fdb,
00306 0x9773, 0x664b, 0x7981, 0x8fd1, 0x70ec, 0x6d78,
00307 /* 0x3e */
00308 0x5c3d, 0x52b2, 0x8346, 0x5162, 0x830e, 0x775b, 0x6676, 0x9cb8,
00309 0x4eac, 0x60ca, 0x7cbe, 0x7cb3, 0x7ecf, 0x4e95, 0x8b66, 0x666f,
00310 0x9888, 0x9759, 0x5883, 0x656c, 0x955c, 0x5f84, 0x75c9, 0x9756,
00311 0x7adf, 0x7ade, 0x51c0, 0x70af, 0x7a98, 0x63ea, 0x7a7e, 0x7ea0,
00312 0x7396, 0x97ed, 0x4e45, 0x7078, 0x4e5d, 0x9152, 0x53a9, 0x6551,
00313 0x65e7, 0x81fc, 0x8205, 0x548e, 0x5c31, 0x759a, 0x97a0, 0x62d8,
00314 0x72d9, 0x75bd, 0x5c45, 0x9a79, 0x83ca, 0x5c40, 0x5480, 0x77e9,
00315 0x4e3e, 0x6cae, 0x805a, 0x62d2, 0x636e, 0x5de8, 0x5177, 0x8ddd,
00316 0x8e1e, 0x952f, 0x4ff1, 0x53e5, 0x60e7, 0x70ac, 0x5267, 0x6350,
00317 0x9e43, 0x5a1f, 0x5026, 0x7737, 0x5377, 0x7ee2, 0x6485, 0x652b,
00318 0x6289, 0x6398, 0x5014, 0x7235, 0x89c9, 0x51b3, 0x8bc0, 0x7edd,
00319 0x5747, 0x83cc, 0x94a7, 0x519b, 0x541b, 0x5cfb,
00320 /* 0x3f */
00321 0x4fca, 0x7ae3, 0x6d5a, 0x90e1, 0x9a8f, 0x5580, 0x5496, 0x5361,
00322 0x54af, 0x5f00, 0x63e9, 0x6977, 0x51ef, 0x6168, 0x520a, 0x582a,
00323 0x52d8, 0x574e, 0x780d, 0x770b, 0x5eb7, 0x6177, 0x7ce0, 0x625b,
00324 0x6297, 0x4ea2, 0x7095, 0x8003, 0x62f7, 0x70e4, 0x9760, 0x5777,
00325 0x82db, 0x67ef, 0x68f5, 0x78d5, 0x9897, 0x79d1, 0x58f3, 0x54b3,
00326 0x53ef, 0x6e34, 0x514b, 0x523b, 0x5ba2, 0x8bfe, 0x80af, 0x5543,
00327 0x57a6, 0x6073, 0x5751, 0x542d, 0x7a7a, 0x6050, 0x5b54, 0x63a7,
00328 0x62a0, 0x53e3, 0x6263, 0x5bc7, 0x67af, 0x54ed, 0x7a9f, 0x82e6,
00329 0x9177, 0x5e93, 0x88e4, 0x5938, 0x57ae, 0x630e, 0x8de8, 0x80ef,
00330 0x5757, 0x7b77, 0x4fa9, 0x5feb, 0x5bbd, 0x6b3e, 0x5321, 0x7b50,
00331 0x72c2, 0x6846, 0x77ff, 0x7736, 0x65f7, 0x51b5, 0x4e8f, 0x76d4,
00332 0x5cbf, 0x7aa5, 0x8475, 0x594e, 0x9b41, 0x5080,
00333 /* 0x40 */
00334 0x9988, 0x6127, 0x6e83, 0x5764, 0x6606, 0x6346, 0x56f0, 0x62ec,
00335 0x6269, 0x5ed3, 0x9614, 0x5783, 0x62c9, 0x5587, 0x8721, 0x814a,
00336 0x8fa3, 0x5566, 0x83b1, 0x6765, 0x8d56, 0x84dd, 0x5a6a, 0x680f,
00337 0x62e6, 0x7bee, 0x9611, 0x5170, 0x6f9c, 0x8c30, 0x63fd, 0x89c8,
00338 0x61d2, 0x7f06, 0x70c2, 0x6ee5, 0x7405, 0x6994, 0x72fc, 0x5eca,
00339 0x90ce, 0x6717, 0x6d6a, 0x635e, 0x52b3, 0x7262, 0x8001, 0x4f6c,
00340 0x59e5, 0x916a, 0x70d9, 0x6d9d, 0x52d2, 0x4e50, 0x96f7, 0x956d,
00341 0x857e, 0x78ca, 0x7d2f, 0x5121, 0x5792, 0x64c2, 0x808b, 0x7c7b,
00342 0x6cea, 0x68f1, 0x695e, 0x51b7, 0x5398, 0x68a8, 0x7281, 0x9ece,
00343 0x7bf1, 0x72f8, 0x79bb, 0x6f13, 0x7406, 0x674e, 0x91cc, 0x9ca4,
00344 0x793c, 0x8389, 0x8354, 0x540f, 0x6817, 0x4e3d, 0x5389, 0x52b1,
00345 0x783e, 0x5386, 0x5229, 0x5088, 0x4f8b, 0x4fd0,
00346 /* 0x41 */
00347 0x75e2, 0x7acb, 0x7c92, 0x6ca5, 0x96b6, 0x529b, 0x7483, 0x54e9,
00348 0x4fe9, 0x8054, 0x83b2, 0x8fde, 0x9570, 0x5ec9, 0x601c, 0x6d9f,
00349 0x5e18, 0x655b, 0x8138, 0x94fe, 0x604b, 0x70bc, 0x7ec3, 0x7cae,
00350 0x51c9, 0x6881, 0x7cb1, 0x826f, 0x4e24, 0x8f86, 0x91cf, 0x667e,
00351 0x4eae, 0x8c05, 0x64a9, 0x804a, 0x50da, 0x7597, 0x71ce, 0x5be5,
00352 0x8fbd, 0x6f66, 0x4e86, 0x6482, 0x9563, 0x5ed6, 0x6599, 0x5217,
00353 0x88c2, 0x70c8, 0x52a3, 0x730e, 0x7433, 0x6797, 0x78f7, 0x9716,
00354 0x4e34, 0x90bb, 0x9cde, 0x6dcb, 0x51db, 0x8d41, 0x541d, 0x62ce,
00355 0x73b2, 0x83f1, 0x96f6, 0x9f84, 0x94c3, 0x4f36, 0x7f9a, 0x51cc,
00356 0x7075, 0x9675, 0x5cad, 0x9886, 0x53e6, 0x4ee4, 0x6e9c, 0x7409,
00357 0x69b4, 0x786b, 0x998f, 0x7559, 0x5218, 0x7624, 0x6d41, 0x67f3,
00358 0x516d, 0x9f99, 0x804b, 0x5499, 0x7b3c, 0x7abf,
00359 /* 0x42 */
00360 0x9686, 0x5784, 0x62e2, 0x9647, 0x697c, 0x5a04, 0x6402, 0x7bd3,
00361 0x6f0f, 0x964b, 0x82a6, 0x5362, 0x9885, 0x5e90, 0x7089, 0x63b3,
00362 0x5364, 0x864f, 0x9c81, 0x9e93, 0x788c, 0x9732, 0x8de7, 0x8d42,
00363 0x9e7f, 0x6f5e, 0x7984, 0x5f55, 0x9646, 0x622e, 0x9a74, 0x5415,
00364 0x94dd, 0x4fa3, 0x65c5, 0x5c65, 0x5c61, 0x7f15, 0x8651, 0x6c2f,
00365 0x5f8b, 0x7387, 0x6ee4, 0x7eff, 0x5ce6, 0x631b, 0x5b6a, 0x6ee6,
00366 0x5375, 0x4e71, 0x63a0, 0x7565, 0x62a1, 0x8f6e, 0x4f26, 0x4ed1,
00367 0x6ca6, 0x7eb6, 0x8bba, 0x841d, 0x87ba, 0x7f57, 0x903b, 0x9523,
00368 0x7ba9, 0x9aa1, 0x88f8, 0x843d, 0x6dlb, 0x9a86, 0x7edc, 0x5988,
00369 0x9ebb, 0x739b, 0x7801, 0x8682, 0x9a6c, 0x9a82, 0x561b, 0x5417,
00370 0x57cb, 0x4e70, 0x9ea6, 0x5356, 0x8fc8, 0x8109, 0x7792, 0x9992,
00371 0x86ee, 0x6ee1, 0x8513, 0x66fc, 0x6162, 0x6f2b,
```

```

00372  /* 0x43 */
00373  0x8c29, 0x8292, 0x832b, 0x76f2, 0x6c13, 0x5fd9, 0x83bd, 0x732b,
00374  0x8305, 0x951a, 0x6bdb, 0x77db, 0x94c6, 0x536f, 0x8302, 0x5192,
00375  0x5e3d, 0x8c8c, 0x8d38, 0x4e48, 0x73ab, 0x679a, 0x6885, 0x9176,
00376  0x9709, 0x7164, 0x6ca1, 0x7709, 0x5a92, 0x9541, 0x6bcf, 0x7f8e,
00377  0x6627, 0x5bd0, 0x59b9, 0x5a9a, 0x95e8, 0x95f7, 0x4eec, 0x840c,
00378  0x8499, 0x6aac, 0x76df, 0x9530, 0x731b, 0x68a6, 0x5b5f, 0x772f,
00379  0x919a, 0x9761, 0x7cdc, 0x8ff7, 0x8c1c, 0x5f25, 0x7c73, 0x79d8,
00380  0x89c5, 0x6ccc, 0x871c, 0x5bc6, 0x5e42, 0x68c9, 0x7720, 0x7ef5,
00381  0x5195, 0x514d, 0x52c9, 0x5a29, 0x7f05, 0x9762, 0x82d7, 0x63cf,
00382  0x7784, 0x85d0, 0x79d2, 0x6e3a, 0x5e99, 0x5999, 0x8511, 0x706d,
00383  0x6c11, 0x62bf, 0x76bf, 0x654f, 0x60af, 0x95fd, 0x660e, 0x879f,
00384  0x9e23, 0x94ed, 0x540d, 0x547d, 0x8c2c, 0x6478,
00385  /* 0x44 */
00386  0x6479, 0x8611, 0x6a21, 0x819c, 0x78e8, 0x6469, 0x9b54, 0x62b9,
00387  0x672b, 0x83ab, 0x58a8, 0x9ed8, 0x6cab, 0x6f20, 0x5bde, 0x964c,
00388  0x8c0b, 0x725f, 0x67d0, 0x62c7, 0x7261, 0x4ea9, 0x59c6, 0x6bcd,
00389  0x5893, 0x66ae, 0x5e55, 0x52df, 0x6155, 0x6728, 0x776e, 0x7766,
00390  0x7267, 0x7a46, 0x62ff, 0x54ea, 0x5450, 0x94a0, 0x90a3, 0x5a1c,
00391  0x7eb3, 0x6c16, 0x4e43, 0x5976, 0x8010, 0x5948, 0x5357, 0x7537,
00392  0x96be, 0x56ca, 0x6320, 0x8111, 0x607c, 0x95f9, 0x6dd6, 0x5462,
00393  0x9981, 0x5185, 0x5ae9, 0x80fd, 0x59ae, 0x9713, 0x502a, 0x6ce5,
00394  0x5c3c, 0x62df, 0x4f60, 0x533f, 0x817b, 0x9006, 0x6eba, 0x852b,
00395  0x62c8, 0x5e74, 0x78be, 0x64b5, 0x637b, 0x5ff5, 0x5a18, 0x917f,
00396  0x9e1f, 0x5c3f, 0x634f, 0x8042, 0x5b7d, 0x556e, 0x954a, 0x954d,
00397  0x6d85, 0x60a8, 0x67e0, 0x72de, 0x51dd, 0x5b81,
00398  /* 0x45 */
00399  0x62e7, 0x6cde, 0x725b, 0x626d, 0x94ae, 0x7ebd, 0x8113, 0x6d53,
00400  0x519c, 0x5f04, 0x5974, 0x52aa, 0x6012, 0x5973, 0x6696, 0x8650,
00401  0x759f, 0x632a, 0x61e6, 0x7cef, 0x8bfa, 0x54e6, 0x6b27, 0x9e25,
00402  0x6bb4, 0x85d5, 0x5455, 0x5076, 0x6ca4, 0x556a, 0x8db4, 0x722c,
00403  0x5e15, 0x6015, 0x7436, 0x62cd, 0x6392, 0x724c, 0x5f98, 0x6e43,
00404  0x6d3e, 0x6500, 0x6f58, 0x76d8, 0x78d0, 0x76fc, 0x7554, 0x5224,
00405  0x53db, 0x4e53, 0x5e9e, 0x65c1, 0x802a, 0x80d6, 0x629b, 0x5486,
00406  0x5228, 0x70ae, 0x888d, 0x8dd1, 0x6ce1, 0x5478, 0x80da, 0x57f9,
00407  0x88f4, 0x8d54, 0x966a, 0x914d, 0x4f69, 0x6c9b, 0x55b7, 0x76c6,
00408  0x7830, 0x62a8, 0x70f9, 0x6f8e, 0x5f6d, 0x84ec, 0x68da, 0x787c,
00409  0x7bf7, 0x81a8, 0x670b, 0x9e4f, 0x6367, 0x78b0, 0x576f, 0x7812,
00410  0x9739, 0x6279, 0x62ab, 0x5288, 0x7435, 0x6bd7,
00411  /* 0x46 */
00412  0x5564, 0x813e, 0x75b2, 0x76ae, 0x5339, 0x75de, 0x50fb, 0x5c41,
00413  0x8b6c, 0x7bc7, 0x504f, 0x7247, 0x9a97, 0x98d8, 0x6f02, 0x74e2,
00414  0x7968, 0x6487, 0x77a5, 0x62fc, 0x9891, 0x8d2b, 0x54c1, 0x8058,
00415  0x4e52, 0x576a, 0x82f9, 0x840d, 0x5e73, 0x51ed, 0x74f6, 0x8bc4,
00416  0x5c4f, 0x5761, 0x6cfc, 0x9887, 0x5a46, 0x7834, 0x9b44, 0x8feb,
00417  0x7c95, 0x5256, 0x6251, 0x94fa, 0x4ec6, 0x8386, 0x8461, 0x83e9,
00418  0x84b2, 0x57d4, 0x6734, 0x5703, 0x666e, 0x6d66, 0x8c31, 0x66dd,
00419  0x7011, 0x671f, 0x6b3a, 0x6816, 0x621a, 0x59bb, 0x4e03, 0x51c4,
00420  0x6f06, 0x67d2, 0x6c8f, 0x5176, 0x68cb, 0x5947, 0x6b67, 0x7566,
00421  0x5d0e, 0x8110, 0x9f50, 0x65d7, 0x7948, 0x7941, 0x9a91, 0x8d77,
00422  0x5c82, 0x4e5e, 0x4f01, 0x542f, 0x5951, 0x780c, 0x5668, 0x6c14,
00423  0x8fc4, 0x5f03, 0x6c7d, 0x6ce3, 0x8bab, 0x6390,
00424  /* 0x47 */
00425  0x6070, 0x6d3d, 0x7275, 0x6266, 0x948e, 0x94c5, 0x5343, 0x8fc1,
00426  0x7b7e, 0x4edf, 0x8c26, 0x4e7e, 0x9ed4, 0x94b1, 0x94b3, 0x524d,
00427  0x6f5c, 0x9063, 0x6d45, 0x8c34, 0x5811, 0x5d4c, 0x6b20, 0x6b49,
00428  0x67aa, 0x545b, 0x8154, 0x7f8c, 0x5899, 0x8537, 0x5f3a, 0x62a2,
00429  0x6a47, 0x9539, 0x6572, 0x6084, 0x6865, 0x77a7, 0x4e54, 0x4fa8,
00430  0x5de7, 0x9798, 0x64ac, 0x7fd8, 0x5ced, 0x4fcf, 0x7a8d, 0x5207,
00431  0x8304, 0x4e14, 0x602f, 0x7a83, 0x94a6, 0x4fb5, 0x4eb2, 0x79e6,
00432  0x7434, 0x52e4, 0x82b9, 0x64d2, 0x79bd, 0x5bdd, 0x6c81, 0x9752,
00433  0x8f7b, 0x6c22, 0x503e, 0x537f, 0x6e05, 0x64ce, 0x6674, 0x6c30,
00434  0x60c5, 0x9877, 0x8bf7, 0x5e86, 0x743c, 0x7a77, 0x79cb, 0x4e18,
00435  0x90b1, 0x7403, 0x6c42, 0x56da, 0x914b, 0x6cc5, 0x8d8b, 0x533a,
00436  0x86c6, 0x66f2, 0x8eaf, 0x5c48, 0x9a71, 0x6e20,
00437  /* 0x48 */
00438  0x53d6, 0x5a36, 0x9f8b, 0x8da3, 0x53bb, 0x5708, 0x98a7, 0x6743,
00439  0x919b, 0x6cc9, 0x5168, 0x75ca, 0x62f3, 0x72ac, 0x5238, 0x529d,
00440  0x7f3a, 0x7094, 0x7638, 0x5374, 0x9e4a, 0x69b7, 0x786e, 0x96c0,
00441  0x88d9, 0x7fa4, 0x7136, 0x71c3, 0x5189, 0x67d3, 0x74e4, 0x58e4,
00442  0x6518, 0x56b7, 0x8ba9, 0x9976, 0x6270, 0x7ed5, 0x60f9, 0x70ed,
00443  0x58ec, 0x4ec1, 0x4eba, 0x5fcd, 0x97e7, 0x4efb, 0x8ba4, 0x5203,
00444  0x598a, 0x7eab, 0x6254, 0x4ecd, 0x65e5, 0x620e, 0x8338, 0x84c9,
00445  0x8363, 0x878d, 0x7194, 0x6eb6, 0x5bb9, 0x7ed2, 0x5197, 0x63c9,
00446  0x67d4, 0x8089, 0x8339, 0x8815, 0x5112, 0x5b7a, 0x5982, 0x8fb1,
00447  0x4e73, 0x6c5d, 0x5165, 0x8925, 0x8f6f, 0x962e, 0x854a, 0x745e,
00448  0x9510, 0x95f0, 0x6da6, 0x82e5, 0x5f31, 0x6492, 0x6d12, 0x8428,
00449  0x816e, 0x9cc3, 0x585e, 0x8d5b, 0x4e09, 0x53c1,
00450  /* 0x49 */
00451  0x4f1e, 0x6563, 0x6851, 0x55d3, 0x4e27, 0x6414, 0x9a9a, 0x626b,
00452  0x5ac2, 0x745f, 0x8272, 0x6da9, 0x68ee, 0x50e7, 0x838e, 0x7802,
00453  0x6740, 0x5239, 0x6c99, 0x7eb1, 0x50bb, 0x5565, 0x715e, 0x7b5b,
00454  0x6652, 0x73ca, 0x82eb, 0x6749, 0x5c71, 0x5220, 0x717d, 0x886b,
00455  0x95ea, 0x9655, 0x64c5, 0x8d61, 0x81b3, 0x5584, 0x6c55, 0x6247,
00456  0x7f2e, 0x5892, 0x4f24, 0x5546, 0x8d4f, 0x664c, 0x4e0a, 0x5c1a,
00457  0x88f3, 0x68a2, 0x634e, 0x7a0d, 0x70e7, 0x828d, 0x52fa, 0x97f6,
00458  0x5c11, 0x54e8, 0x90b5, 0x7ecd, 0x5962, 0x8d4a, 0x86c7, 0x820c,

```

```
00459 0x820d, 0x8d66, 0x6444, 0x5c04, 0x6151, 0x6d89, 0x793e, 0x8bbe,
00460 0x7837, 0x7533, 0x547b, 0x4f38, 0x8eab, 0x6df1, 0x5a20, 0x7ec5,
00461 0x795e, 0x6c88, 0x5ba1, 0x5a76, 0x751a, 0x80be, 0x614e, 0x6e17,
00462 0x58f0, 0x751f, 0x7525, 0x7272, 0x5347, 0x7ef3,
00463 /* 0x4a */
00464 0x7701, 0x76db, 0x5269, 0x80dc, 0x5723, 0x5e08, 0x5931, 0x72ee,
00465 0x65bd, 0x6e7f, 0x8bd7, 0x5c38, 0x8671, 0x5341, 0x77f3, 0x62fe,
00466 0x65f6, 0x4ec0, 0x98df, 0x8680, 0x5b9e, 0x8bc6, 0x53f2, 0x77e2,
00467 0x4f7f, 0x5c4e, 0x9a76, 0x59cb, 0x5f0f, 0x793a, 0x58eb, 0x4e16,
00468 0x67ff, 0x4e8b, 0x62ed, 0x8a93, 0x901d, 0x52bf, 0x662f, 0x55dc,
00469 0x566c, 0x9002, 0x4ed5, 0x4f8d, 0x91ca, 0x9970, 0x6c0f, 0x5e02,
00470 0x6043, 0x5ba4, 0x89c6, 0x8bd5, 0x6536, 0x624b, 0x9996, 0x5b88,
00471 0x5bff, 0x6388, 0x552e, 0x53d7, 0x7626, 0x517d, 0x852c, 0x67a2,
00472 0x68b3, 0x6b8a, 0x6292, 0x8f93, 0x53d4, 0x8212, 0x6dd1, 0x758f,
00473 0x4e66, 0x8d4e, 0x5b70, 0x719f, 0x85af, 0x6691, 0x66d9, 0x7f72,
00474 0x8700, 0x9ecd, 0x9f20, 0x5c5e, 0x672f, 0x8ff0, 0x6811, 0x675f,
00475 0x620d, 0x7ad6, 0x5885, 0x5eb6, 0x6570, 0x6f31,
00476 /* 0x4b */
00477 0x6055, 0x5237, 0x800d, 0x6454, 0x8870, 0x7529, 0x5e05, 0x6813,
00478 0x62f4, 0x971c, 0x53cc, 0x723d, 0x8c01, 0x6c34, 0x7761, 0x7a0e,
00479 0x542e, 0x77ac, 0x987a, 0x821c, 0x8bf4, 0x7855, 0x6714, 0x70c1,
00480 0x65af, 0x6495, 0x5636, 0x601d, 0x79c1, 0x53f8, 0x4e1d, 0x6b7b,
00481 0x8086, 0x5bf5, 0x55e3, 0x56db, 0x4f3a, 0x4f3c, 0x9972, 0x5df3,
00482 0x677e, 0x8038, 0x6002, 0x9882, 0x9001, 0x5b8b, 0x8bbc, 0x8bf5,
00483 0x641c, 0x8258, 0x64de, 0x55fd, 0x82cf, 0x9165, 0x4fd7, 0x7d20,
00484 0x901f, 0x7c9f, 0x50f3, 0x5851, 0x6eaf, 0x5bbf, 0x8bc9, 0x8083,
00485 0x9178, 0x849c, 0x7b97, 0x867d, 0x968b, 0x968f, 0x7ee5, 0x9ad3,
00486 0x788e, 0x5c81, 0x7a57, 0x9042, 0x96a7, 0x795f, 0x5b59, 0x635f,
00487 0x7b0b, 0x84d1, 0x68ad, 0x5506, 0x7f29, 0x7410, 0x7d22, 0x9501,
00488 0x6240, 0x584c, 0x4ed6, 0x5b83, 0x5979, 0x5854,
00489 /* 0x4c */
00490 0x736d, 0x631e, 0x8e4b, 0x8e0f, 0x80ce, 0x82d4, 0x62ac, 0x53f0,
00491 0x6cf0, 0x915e, 0x592a, 0x6001, 0x6c70, 0x574d, 0x644a, 0x8d2a,
00492 0x762b, 0x6ee9, 0x575b, 0x6a80, 0x75f0, 0x6f6d, 0x8c2d, 0x8c08,
00493 0x5766, 0x6bef, 0x8892, 0x78b3, 0x63a2, 0x53f9, 0x70ad, 0x6c64,
00494 0x5858, 0x642a, 0x5802, 0x68e0, 0x819b, 0x5510, 0x7cd6, 0x5018,
00495 0x8eba, 0x6dcc, 0x8d9f, 0x70eb, 0x638f, 0x6d9b, 0x6ed4, 0x7ee6,
00496 0x8404, 0x6843, 0x9003, 0x6dd8, 0x9676, 0x8ba8, 0x5957, 0x7279,
00497 0x85e4, 0x817e, 0x75bc, 0x8a8a, 0x68af, 0x5254, 0x8e22, 0x9511,
00498 0x63d0, 0x9898, 0x8e44, 0x557c, 0x4f53, 0x66ff, 0x568f, 0x60d5,
00499 0x6d95, 0x5243, 0x5c49, 0x5929, 0x6dfb, 0x586b, 0x7530, 0x751c,
00500 0x606c, 0x8214, 0x8146, 0x6311, 0x6761, 0x8fe2, 0x773a, 0x8df3,
00501 0x8d34, 0x94c1, 0x5e16, 0x5385, 0x542c, 0x70c3,
00502 /* 0x4d */
00503 0x6c40, 0x5ef7, 0x505c, 0x4ead, 0x5ead, 0x633a, 0x8247, 0x901a,
00504 0x6850, 0x916e, 0x77b3, 0x540c, 0x94dc, 0x5f64, 0x7ae5, 0x6876,
00505 0x6345, 0x7b52, 0x7edf, 0x75db, 0x5077, 0x6295, 0x5934, 0x900f,
00506 0x51f8, 0x79c3, 0x7a81, 0x56fe, 0x5f92, 0x9014, 0x6d82, 0x5c60,
00507 0x571f, 0x5410, 0x5154, 0x6e4d, 0x56e2, 0x63a8, 0x9893, 0x817f,
00508 0x8715, 0x892a, 0x9000, 0x541e, 0x5c6f, 0x81c0, 0x62d6, 0x6258,
00509 0x8131, 0x9e35, 0x9640, 0x9a6e, 0x9a7c, 0x692d, 0x59a5, 0x62d3,
00510 0x553e, 0x6316, 0x54c7, 0x86d9, 0x6d3c, 0x5a03, 0x74e6, 0x889c,
00511 0x6b6a, 0x5916, 0x8c4c, 0x5f2f, 0x6e7e, 0x73a9, 0x987d, 0x4e38,
00512 0x70f7, 0x5b8c, 0x7897, 0x633d, 0x665a, 0x7696, 0x60cb, 0x5b9b,
00513 0x5a49, 0x4e07, 0x8155, 0x6c6a, 0x738b, 0x4ea1, 0x6789, 0x7f51,
00514 0x5f80, 0x65fa, 0x671b, 0x5fd8, 0x5984, 0x5a01,
00515 /* 0x4e */
00516 0x5dcd, 0x5fae, 0x5371, 0x97e6, 0x8fdd, 0x6845, 0x56f4, 0x552f,
00517 0x60df, 0x4e3a, 0x6f4d, 0x7ef4, 0x82c7, 0x840e, 0x59d4, 0x4f1f,
00518 0x4f2a, 0x5c3c, 0x7eac, 0x672a, 0x851a, 0x5473, 0x754f, 0x80c3,
00519 0x5582, 0x9b4f, 0x4f4d, 0x6e2d, 0x8c13, 0x5c09, 0x6170, 0x536b,
00520 0x761f, 0x6e29, 0x868a, 0x6587, 0x95fb, 0x7eb9, 0x543b, 0x7a33,
00521 0x7d0a, 0x95ee, 0x55e1, 0x7fc1, 0x74ee, 0x631d, 0x8717, 0x6da1,
00522 0x7a9d, 0x6211, 0x65a1, 0x5367, 0x63e1, 0x6c83, 0x5deb, 0x545c,
00523 0x94a8, 0x4e4c, 0x6c61, 0x8bec, 0x5c4b, 0x65e0, 0x829c, 0x68a7,
00524 0x543e, 0x5434, 0x6bcb, 0x6b66, 0x4e94, 0x6342, 0x5348, 0x821e,
00525 0x4f0d, 0x4fae, 0x575e, 0x620a, 0x96fe, 0x6664, 0x7269, 0x52ff,
00526 0x52a1, 0x609f, 0x8bef, 0x6614, 0x7199, 0x6790, 0x897f, 0x7852,
00527 0x77fd, 0x6670, 0x563b, 0x5438, 0x9521, 0x727a,
00528 /* 0x4f */
00529 0x7a00, 0x606f, 0x5e0c, 0x6089, 0x819d, 0x5915, 0x60dc, 0x7184,
00530 0x70ef, 0x6eaa, 0x6c50, 0x7280, 0x6a84, 0x88ad, 0x5e2d, 0x4e60,
00531 0x5ab3, 0x559c, 0x94e3, 0x6d17, 0x7cfb, 0x9699, 0x620f, 0x7ec6,
00532 0x778e, 0x867e, 0x5323, 0x971e, 0x8f96, 0x6687, 0x5ce1, 0x4fa0,
00533 0x72ed, 0x4e0b, 0x53a6, 0x590f, 0x5413, 0x6380, 0x9528, 0x5148,
00534 0x4ed9, 0x9c9c, 0x7ea4, 0x54b8, 0x8d24, 0x8854, 0x8237, 0x95f2,
00535 0x6d8e, 0x5f26, 0x5acc, 0x663e, 0x9669, 0x73b0, 0x732e, 0x53bf,
00536 0x817a, 0x9985, 0x7fa1, 0x5baa, 0x9677, 0x9650, 0x7ebf, 0x76f8,
00537 0x53a2, 0x9576, 0x9999, 0x7bb1, 0x8944, 0x6e58, 0x4e61, 0x7fd4,
00538 0x7965, 0x8be6, 0x60f3, 0x54cd, 0x4eab, 0x9879, 0x5df7, 0x6a61,
00539 0x50cf, 0x5411, 0x8c61, 0x8427, 0x785d, 0x9704, 0x524a, 0x54ee,
00540 0x56a3, 0x9500, 0x6d88, 0x5bb5, 0x6dc6, 0x6653,
00541 /* 0x50 */
00542 0x5c0f, 0x5b5d, 0x6821, 0x8096, 0x5578, 0x7b11, 0x6548, 0x6954,
00543 0x4e9b, 0x6b47, 0x874e, 0x978b, 0x534f, 0x631f, 0x643a, 0x90aa,
00544 0x659c, 0x80c1, 0x8c10, 0x5199, 0x68b0, 0x5378, 0x87f9, 0x61c8,
00545 0x6cc4, 0x6cfb, 0x8c22, 0x5c51, 0x85aa, 0x82af, 0x950c, 0x6b23,
```

```
00546 0x8f9b, 0x65b0, 0x5ffb, 0x5fc3, 0x4fe1, 0x8845, 0x661f, 0x8165,
00547 0x7329, 0x60fa, 0x5174, 0x5211, 0x578b, 0x5f62, 0x90a2, 0x884c,
00548 0x9192, 0x5e78, 0x674f, 0x6027, 0x59d3, 0x5144, 0x51f6, 0x80f8,
00549 0x5308, 0x6c79, 0x96c4, 0x718a, 0x4f11, 0x4fee, 0x7f9e, 0x673d,
00550 0x55c5, 0x9508, 0x79c0, 0x8896, 0x7ee3, 0x589f, 0x620c, 0x9700,
00551 0x865a, 0x5618, 0x987b, 0x5f90, 0x8bb8, 0x84c4, 0x9157, 0x53d9,
00552 0x65ed, 0x5e8f, 0x755c, 0x6064, 0x7d6e, 0x5a7f, 0x7eea, 0x7eed,
00553 0x8f69, 0x55a7, 0x5ba3, 0x60ac, 0x65cb, 0x7384,
00554 /* 0x51 */
00555 0x9009, 0x7663, 0x7729, 0x7eda, 0x9774, 0x859b, 0x5b66, 0x7a74,
00556 0x96ea, 0x8840, 0x52cb, 0x718f, 0x5faa, 0x65ec, 0x8be2, 0x5bfb,
00557 0x9a6f, 0x5de1, 0x6b89, 0x6c5b, 0x8bad, 0x8baf, 0x900a, 0x8fc5,
00558 0x538b, 0x62bc, 0x9e26, 0x9e2d, 0x5440, 0x4e2b, 0x82bd, 0x7259,
00559 0x869c, 0x5d16, 0x8859, 0x6daf, 0x96c5, 0x54d1, 0x4e9a, 0x8bb6,
00560 0x7109, 0x54bd, 0x9609, 0x70df, 0x6df9, 0x76d0, 0x4e25, 0x7814,
00561 0x8712, 0x5ca9, 0x5ef6, 0x8a00, 0x989c, 0x960e, 0x708e, 0x6cbf,
00562 0x5944, 0x63a9, 0x773c, 0x884d, 0x6f14, 0x8273, 0x5830, 0x71d5,
00563 0x538c, 0x781a, 0x96c1, 0x5501, 0x5f66, 0x7130, 0x5bb4, 0x8c1a,
00564 0x9a8c, 0x6b83, 0x592e, 0x9e2f, 0x79e7, 0x6768, 0x626c, 0x4f6f,
00565 0x75a1, 0x7f8a, 0x6d0b, 0x9633, 0x6c27, 0x4ef0, 0x75d2, 0x517b,
00566 0x6837, 0x6f3e, 0x9080, 0x8170, 0x5996, 0x7476,
00567 /* 0x52 */
00568 0x6447, 0x5c27, 0x9065, 0x7a91, 0x8c23, 0x59da, 0x54ac, 0x8200,
00569 0x836f, 0x8981, 0x8000, 0x6930, 0x564e, 0x8036, 0x7237, 0x91ce,
00570 0x51b6, 0x4e5f, 0x9875, 0x6396, 0x4e1a, 0x53f6, 0x66f3, 0x814b,
00571 0x591c, 0x6db2, 0x4e00, 0x58f9, 0x533b, 0x63d6, 0x94f1, 0x4f9d,
00572 0x4f0a, 0x8863, 0x9890, 0x5937, 0x9057, 0x79fb, 0x4eea, 0x80f0,
00573 0x7591, 0x6c82, 0x5b9c, 0x59e8, 0x5f5d, 0x6905, 0x8681, 0x501a,
00574 0x5df2, 0x4e59, 0x77e3, 0x4ee5, 0x827a, 0x6291, 0x6613, 0x9091,
00575 0x5c79, 0x4ebf, 0x5f79, 0x81c6, 0x9038, 0x8084, 0x75ab, 0x4ea6,
00576 0x88d4, 0x610f, 0x6bc5, 0x5fc6, 0x4e49, 0x76ca, 0x6ea2, 0x8be3,
00577 0x8bae, 0x8c0a, 0x8bd1, 0x5f02, 0x7ffc, 0x7fcc, 0x7ece, 0x8335,
00578 0x836b, 0x56e0, 0x6bb7, 0x97f3, 0x9634, 0x59fb, 0x541f, 0x94f6,
00579 0x6deb, 0x5bc5, 0x996e, 0x5c39, 0x5f15, 0x9690,
00580 /* 0x53 */
00581 0x5370, 0x82f1, 0x6a31, 0x5a74, 0x9e70, 0x5e94, 0x7f28, 0x83b9,
00582 0x8424, 0x8425, 0x8367, 0x8747, 0x8fce, 0x8d62, 0x76c8, 0x5f71,
00583 0x9896, 0x786c, 0x6620, 0x54df, 0x62e5, 0x4f63, 0x81c3, 0x75c8,
00584 0x5eb8, 0x96cd, 0x8e0a, 0x86f9, 0x548f, 0x6cf3, 0x6d8c, 0x6c38,
00585 0x607f, 0x52c7, 0x7528, 0x5e7d, 0x4f18, 0x60a0, 0x5fe7, 0x5c24,
00586 0x7531, 0x90ae, 0x94c0, 0x72b9, 0x6cb9, 0x6e38, 0x9149, 0x6709,
00587 0x53cb, 0x53f3, 0x4f51, 0x91c9, 0x8bf1, 0x53c8, 0x5e7c, 0x8fc2,
00588 0x6de4, 0x4e8e, 0x76c2, 0x6986, 0x865e, 0x611a, 0x8206, 0x4f59,
00589 0x4fde, 0x903e, 0x9c7c, 0x6109, 0x6e1d, 0x6e14, 0x9685, 0x4e88,
00590 0x5a31, 0x96e8, 0x4e0e, 0x5c7f, 0x79b9, 0x5b87, 0x8bed, 0x7fbd,
00591 0x7389, 0x57df, 0x828b, 0x90c1, 0x5401, 0x9047, 0x55bb, 0x5cea,
00592 0x5fa1, 0x6108, 0x6b32, 0x72f1, 0x80b2, 0x8a89,
00593 /* 0x54 */
00594 0x6d74, 0x5bd3, 0x88d5, 0x9884, 0x8c6b, 0x9a6d, 0x9e33, 0x6e0a,
00595 0x51a4, 0x5143, 0x57a3, 0x8881, 0x539f, 0x63f4, 0x8f95, 0x56ed,
00596 0x5458, 0x5706, 0x733f, 0x6e90, 0x7f18, 0x8fdc, 0x82d1, 0x613f,
00597 0x6028, 0x9662, 0x66f0, 0x7ea6, 0x8d8a, 0x8dc3, 0x94a5, 0x5cb3,
00598 0x7ca4, 0x6708, 0x60a6, 0x9605, 0x8018, 0x4e91, 0x90e7, 0x5300,
00599 0x9668, 0x5141, 0x8fd0, 0x8574, 0x915d, 0x6655, 0x97f5, 0x5b55,
00600 0x531d, 0x7838, 0x6742, 0x683d, 0x54c9, 0x707e, 0x5bb0, 0x8ff7,
00601 0x518d, 0x5728, 0x54b1, 0x6512, 0x6682, 0x8d5e, 0x8d43, 0x810f,
00602 0x846c, 0x906d, 0x7cdf, 0x51ff, 0x85fb, 0x67a3, 0x65e9, 0x6fa1,
00603 0x86a4, 0x8e81, 0x566a, 0x9020, 0x7682, 0x7076, 0x71e5, 0x8d23,
00604 0x62e9, 0x5219, 0x6cfd, 0x8d3c, 0x600e, 0x589e, 0x618e, 0x66fe,
00605 0x8d60, 0x624e, 0x55b3, 0x6e23, 0x672d, 0x8f67,
00606 /* 0x55 */
00607 0x94e1, 0x95f8, 0x7728, 0x6805, 0x69a8, 0x548b, 0x4e4d, 0x70b8,
00608 0x8bc8, 0x6458, 0x658b, 0x5b85, 0x7a84, 0x503a, 0x5be8, 0x77bb,
00609 0x6be1, 0x8a79, 0x7c98, 0x6cbe, 0x76cf, 0x65a9, 0x8f97, 0x5d2d,
00610 0x5c55, 0x8638, 0x6808, 0x5360, 0x6218, 0x7ad9, 0x6e5b, 0x7efd,
00611 0x6a1f, 0x7ae0, 0x5f70, 0x6f33, 0x5f20, 0x638c, 0x6da8, 0x6756,
00612 0x4e08, 0x5e10, 0x8d26, 0x4ed7, 0x80c0, 0x7634, 0x969c, 0x62db,
00613 0x662d, 0x627e, 0x6cbc, 0x8d75, 0x7167, 0x7f69, 0x5146, 0x8087,
00614 0x53ec, 0x906e, 0x6298, 0x54f2, 0x86f0, 0x8f99, 0x8005, 0x9517,
00615 0x8517, 0x8fd9, 0x6d59, 0x73cd, 0x659f, 0x771f, 0x7504, 0x7827,
00616 0x81fb, 0x8d1e, 0x9488, 0x4fa6, 0x6795, 0x75b9, 0x8bca, 0x9707,
00617 0x632f, 0x9547, 0x9635, 0x84b8, 0x6323, 0x7741, 0x5f81, 0x72f0,
00618 0x4e89, 0x6014, 0x6574, 0x62ef, 0x6b63, 0x653f,
00619 /* 0x56 */
00620 0x5e27, 0x75c7, 0x90d1, 0x8bc1, 0x829d, 0x679d, 0x652f, 0x5431,
00621 0x8718, 0x77e5, 0x80a2, 0x8102, 0x6c41, 0x4e4b, 0x7ec7, 0x804c,
00622 0x76f4, 0x69d0, 0x6b96, 0x6267, 0x503c, 0x4f84, 0x5740, 0x6307,
00623 0x6b62, 0x8dbe, 0x53ea, 0x65e8, 0x7eb8, 0x5fd7, 0x631a, 0x63b7,
00624 0x81f3, 0x81fd, 0x7f6e, 0x5e1c, 0x5cd9, 0x5236, 0x667a, 0x79e9,
00625 0x7a1a, 0x8d28, 0x7099, 0x75d4, 0x6ede, 0x6cbb, 0x7a92, 0x4e2d,
00626 0x76c5, 0x5fe0, 0x949f, 0x8877, 0x7ec8, 0x79cd, 0x80bf, 0x91cd,
00627 0x4ef2, 0x4f17, 0x821f, 0x5468, 0x5dde, 0x6d32, 0x8bcc, 0x7ca5,
00628 0x8f74, 0x8098, 0x5e1a, 0x5492, 0x76b1, 0x5b99, 0x663c, 0x9aa4,
00629 0x73e0, 0x682a, 0x86db, 0x6731, 0x732a, 0x8bf8, 0x8bdb, 0x9010,
00630 0x7af9, 0x70db, 0x716e, 0x62c4, 0x77a9, 0x5631, 0x4e3b, 0x8457,
00631 0x67f1, 0x52a9, 0x86c0, 0x8d2e, 0x94f8, 0x7b51,
00632 /* 0x57 */
```



```
00633 0x4f4f, 0x6ce8, 0x795d, 0x9a7b, 0x6293, 0x722a, 0x62fd, 0x4e13,
00634 0x7816, 0x8f6c, 0x64b0, 0x8d5a, 0x7bc6, 0x6869, 0x5e84, 0x88c5,
00635 0x5986, 0x649e, 0x58ee, 0x72b6, 0x690e, 0x9525, 0x8ffd, 0x8d58,
00636 0x5760, 0x7f00, 0x8c06, 0x51c6, 0x6349, 0x62d9, 0x5353, 0x684c,
00637 0x7422, 0x8301, 0x914c, 0x5544, 0x7740, 0x707c, 0x6d4a, 0x5179,
00638 0x54a8, 0x8d44, 0x59ff, 0x6ecb, 0x6dc4, 0x5b5c, 0x7d2b, 0x4ed4,
00639 0x7c7d, 0x6ed3, 0x5b50, 0x81ea, 0x6e0d, 0x5b57, 0x9b03, 0x68d5,
00640 0x8e2a, 0x5b97, 0x7efc, 0x603b, 0x7eb5, 0x90b9, 0x8d70, 0x594f,
00641 0x63cd, 0x79df, 0x8db3, 0x5352, 0x65cf, 0x7956, 0x8bc5, 0x963b,
00642 0x7ec4, 0x94bb, 0x7e82, 0x5634, 0x9189, 0x6700, 0x7f6a, 0x5c0a,
00643 0x9075, 0x6628, 0x5de6, 0x4f50, 0x67de, 0x505a, 0x4f5c, 0x5750,
00644 0x5ea7, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00645 /* 0x58 */
00646 0x4e8d, 0x4e0c, 0x5140, 0x4e10, 0x5eff, 0x5345, 0x4e15, 0x4e98,
00647 0x4e1e, 0x9b32, 0x5b6c, 0x5669, 0x4e28, 0x79ba, 0x4e3f, 0x5315,
00648 0x4e47, 0x592d, 0x723b, 0x536e, 0x6c10, 0x56df, 0x80e4, 0x9997,
00649 0x6bd3, 0x777e, 0x9f17, 0x4e36, 0x4e9f, 0x9f10, 0x4e5c, 0x4e69,
00650 0x4e93, 0x8288, 0x5b5b, 0x556c, 0x560f, 0x4ec4, 0x538d, 0x539d,
00651 0x53a3, 0x53a5, 0x53ae, 0x9765, 0x8d5d, 0x531a, 0x53f5, 0x5326,
00652 0x532e, 0x533e, 0x8d5c, 0x5366, 0x5363, 0x5202, 0x5208, 0x520e,
00653 0x522d, 0x5233, 0x523f, 0x5240, 0x524c, 0x525e, 0x5261, 0x525c,
00654 0x84af, 0x527d, 0x5282, 0x5281, 0x5290, 0x5293, 0x5182, 0x7f54,
00655 0x4ebb, 0x4ec3, 0x4ec9, 0x4ec2, 0x4ee8, 0x4ee1, 0x4eeb, 0x4ede,
00656 0x4f1b, 0x4ef3, 0x4f22, 0x4f64, 0x4ef5, 0x4f25, 0x4f27, 0x4f09,
00657 0x4f2b, 0x4f5e, 0x4f67, 0x6538, 0x4f5a, 0x4f5d,
00658 /* 0x59 */
00659 0x4f5f, 0x4f57, 0x4f32, 0x4f3d, 0x4f76, 0x4f74, 0x4f91, 0x4f89,
00660 0x4f83, 0x4f8f, 0x4f7e, 0x4f7b, 0x4faa, 0x4f7c, 0x4fac, 0x4f94,
00661 0x4fe6, 0x4fe8, 0x4fea, 0x4fc5, 0x4fda, 0x4fe3, 0x4fdc, 0x4fd1,
00662 0x4fdf, 0x4fff, 0x5029, 0x504c, 0x4ff3, 0x502c, 0x500f, 0x502e,
00663 0x502d, 0x4ffe, 0x501c, 0x500c, 0x5025, 0x5028, 0x507e, 0x5043,
00664 0x5055, 0x5048, 0x504e, 0x506c, 0x507b, 0x50a5, 0x50a7, 0x50a9,
00665 0x50ba, 0x50d6, 0x5106, 0x50ed, 0x50ec, 0x50e6, 0x50ee, 0x5107,
00666 0x510b, 0x4edd, 0x6c3d, 0x4f58, 0x4f65, 0x4fce, 0x9fa0, 0x6c46,
00667 0x7c74, 0x516e, 0x5dfd, 0x9ec9, 0x9998, 0x5181, 0x5914, 0x52f9,
00668 0x530d, 0x8a07, 0x5310, 0x51eb, 0x5919, 0x5155, 0x4ea0, 0x5156,
00669 0x4eb3, 0x886e, 0x88a4, 0x4eb5, 0x8114, 0x88d2, 0x7980, 0x5b34,
00670 0x8803, 0x7fb8, 0x51ab, 0x51b1, 0x51bd, 0x51bc,
00671 /* 0x5a */
00672 0x51c7, 0x5196, 0x51a2, 0x51a5, 0x8ba0, 0x8ba6, 0x8ba7, 0x8baa,
00673 0x8bb4, 0x8bb5, 0x8bb7, 0x8bc2, 0x8bc3, 0x8bcb, 0x8bcf, 0x8bce,
00674 0x8bd2, 0x8bd3, 0x8bd4, 0x8bd6, 0x8bd8, 0x8bd9, 0x8bdc, 0x8bdf,
00675 0x8be0, 0x8be4, 0x8be8, 0x8be9, 0x8bee, 0x8bf0, 0x8bf3, 0x8bf6,
00676 0x8bf9, 0x8bfc, 0x8bff, 0x8c00, 0x8c02, 0x8c04, 0x8c07, 0x8c0c,
00677 0x8c0f, 0x8c11, 0x8c12, 0x8c14, 0x8c15, 0x8c16, 0x8c19, 0x8c1b,
00678 0x8c18, 0x8c1d, 0x8c1f, 0x8c20, 0x8c21, 0x8c25, 0x8c27, 0x8c2a,
00679 0x8c2b, 0x8c2e, 0x8c2f, 0x8c32, 0x8c33, 0x8c35, 0x8c36, 0x5369,
00680 0x537a, 0x961d, 0x9622, 0x9621, 0x9631, 0x962a, 0x963d, 0x963c,
00681 0x9642, 0x9649, 0x9654, 0x965f, 0x9667, 0x966c, 0x9672, 0x9674,
00682 0x9688, 0x968d, 0x9697, 0x96b0, 0x9097, 0x909b, 0x909d, 0x9099,
00683 0x90ac, 0x90a1, 0x90b4, 0x90b3, 0x90b6, 0x90ba,
00684 /* 0x5b */
00685 0x90b8, 0x90b0, 0x90cf, 0x90c5, 0x90be, 0x90d0, 0x90c4, 0x90c7,
00686 0x90d3, 0x90e6, 0x90e2, 0x90dc, 0x90d7, 0x90db, 0x90eb, 0x90ef,
00687 0x90fe, 0x9104, 0x9122, 0x911e, 0x9123, 0x9131, 0x912f, 0x9139,
00688 0x9143, 0x9146, 0x520d, 0x5942, 0x52a2, 0x52ac, 0x52ad, 0x52be,
00689 0x54ff, 0x52d0, 0x52d6, 0x52f0, 0x53df, 0x71ee, 0x77cd, 0x5ef4,
00690 0x51f5, 0x51fc, 0x9b2f, 0x53b6, 0x5f01, 0x755a, 0x5def, 0x574c,
00691 0x57a9, 0x57a1, 0x587e, 0x58bc, 0x58c5, 0x58d1, 0x5729, 0x572c,
00692 0x572a, 0x573a, 0x5739, 0x572e, 0x572f, 0x575c, 0x573b, 0x5742,
00693 0x5769, 0x5785, 0x576b, 0x5786, 0x577c, 0x577b, 0x5768, 0x576d,
00694 0x5776, 0x5773, 0x57ad, 0x57a4, 0x578c, 0x57b2, 0x57cf, 0x57a7,
00695 0x57b4, 0x5793, 0x57a0, 0x57d5, 0x57d8, 0x57da, 0x57d9, 0x57d2,
00696 0x57b8, 0x57f4, 0x57ef, 0x57f8, 0x57e4, 0x57dd,
00697 /* 0x5c */
00698 0x580b, 0x580d, 0x57fd, 0x57ed, 0x5800, 0x581e, 0x5819, 0x5844,
00699 0x5820, 0x5865, 0x586c, 0x5881, 0x5889, 0x589a, 0x5880, 0x99a8,
00700 0x9f19, 0x61ff, 0x8279, 0x827d, 0x827f, 0x828f, 0x828a, 0x82a8,
00701 0x8284, 0x828e, 0x8291, 0x8297, 0x8299, 0x82ab, 0x82b8, 0x82be,
00702 0x82b0, 0x82c8, 0x82ca, 0x82e3, 0x8298, 0x82b7, 0x82ae, 0x82cb,
00703 0x82cc, 0x82c1, 0x82a9, 0x82b4, 0x82a1, 0x82aa, 0x829f, 0x82c4,
00704 0x82ce, 0x82a4, 0x82e1, 0x8309, 0x82f7, 0x82e4, 0x830f, 0x8307,
00705 0x82dc, 0x82f4, 0x82d2, 0x82d8, 0x830c, 0x82fb, 0x82d3, 0x8311,
00706 0x831a, 0x8306, 0x8314, 0x8315, 0x82e0, 0x82d5, 0x831c, 0x8351,
00707 0x835b, 0x835c, 0x8308, 0x8392, 0x833c, 0x8334, 0x8331, 0x839b,
00708 0x835e, 0x832f, 0x834f, 0x8347, 0x8343, 0x835f, 0x8340, 0x8317,
00709 0x8360, 0x832d, 0x833a, 0x8333, 0x8366, 0x8365,
00710 /* 0x5d */
00711 0x8368, 0x831b, 0x8369, 0x836c, 0x836a, 0x836d, 0x836e, 0x83b0,
00712 0x8378, 0x83b3, 0x83b4, 0x83a0, 0x83aa, 0x8393, 0x839c, 0x8385,
00713 0x837c, 0x83b6, 0x83a9, 0x837d, 0x83b8, 0x837b, 0x8398, 0x839e,
00714 0x83a8, 0x83ba, 0x83bc, 0x83c1, 0x8401, 0x83e5, 0x83d8, 0x5807,
00715 0x8418, 0x840b, 0x83dd, 0x83fd, 0x83d6, 0x841c, 0x8438, 0x8411,
00716 0x8406, 0x83d4, 0x83df, 0x840f, 0x8403, 0x83f8, 0x83f9, 0x83ea,
00717 0x83c5, 0x83c0, 0x8426, 0x83f0, 0x83e1, 0x845c, 0x8451, 0x845a,
00718 0x8459, 0x8473, 0x8487, 0x8488, 0x847a, 0x8489, 0x8478, 0x843c,
00719 0x8446, 0x8469, 0x8476, 0x848c, 0x848e, 0x8431, 0x846d, 0x84c1,
```

```

00720 0x84cd, 0x84d0, 0x84e6, 0x84bd, 0x84d3, 0x84ca, 0x84bf, 0x84ba,
00721 0x84e0, 0x84a1, 0x84b9, 0x84b4, 0x8497, 0x84e5, 0x84e3, 0x850c,
00722 0x750d, 0x8538, 0x84f0, 0x8539, 0x851f, 0x853a,
00723 /* 0x5e */
00724 0x8556, 0x853b, 0x84ff, 0x84fc, 0x8559, 0x8548, 0x8568, 0x8564,
00725 0x855e, 0x857a, 0x77a2, 0x8543, 0x8572, 0x857b, 0x85a4, 0x85a8,
00726 0x8587, 0x858f, 0x8579, 0x85ae, 0x859c, 0x8585, 0x85b9, 0x85b7,
00727 0x85b0, 0x85d3, 0x85c1, 0x85dc, 0x85ff, 0x8627, 0x8605, 0x8629,
00728 0x8616, 0x863c, 0x5efe, 0x5f08, 0x593c, 0x5941, 0x8037, 0x5955,
00729 0x595a, 0x5958, 0x530f, 0x5c22, 0x5c25, 0x5c2c, 0x5c34, 0x624c,
00730 0x626a, 0x629f, 0x62bb, 0x62ca, 0x62da, 0x62d7, 0x62ee, 0x6322,
00731 0x62f6, 0x6339, 0x634b, 0x6343, 0x63ad, 0x63f6, 0x6371, 0x637a,
00732 0x638e, 0x63b4, 0x636d, 0x63ac, 0x638a, 0x6369, 0x63ae, 0x63bc,
00733 0x63f2, 0x63f8, 0x63e0, 0x63ff, 0x63c4, 0x63de, 0x63ce, 0x6452,
00734 0x63c6, 0x63be, 0x6445, 0x6441, 0x640b, 0x641b, 0x6420, 0x640c,
00735 0x6426, 0x6421, 0x645e, 0x6484, 0x646d, 0x6496,
00736 /* 0x5f */
00737 0x647a, 0x64b7, 0x64b8, 0x6499, 0x64ba, 0x64c0, 0x64d0, 0x64d7,
00738 0x64e4, 0x64e2, 0x6509, 0x6525, 0x652e, 0x5f0b, 0x5fd2, 0x7519,
00739 0x5f11, 0x535f, 0x53f1, 0x53fd, 0x53e9, 0x53e8, 0x53fb, 0x5412,
00740 0x5416, 0x5406, 0x544b, 0x5452, 0x5453, 0x5454, 0x5456, 0x5443,
00741 0x5421, 0x5457, 0x5459, 0x5423, 0x5432, 0x5482, 0x5494, 0x5477,
00742 0x5471, 0x5464, 0x549a, 0x549b, 0x5484, 0x5476, 0x5466, 0x549d,
00743 0x54d0, 0x54ad, 0x54c2, 0x54b4, 0x54d2, 0x54a7, 0x54a6, 0x54d3,
00744 0x54d4, 0x5472, 0x54a3, 0x54d5, 0x54bb, 0x54bf, 0x54cc, 0x54d9,
00745 0x54da, 0x54dc, 0x54a9, 0x54aa, 0x54a4, 0x54dd, 0x54cf, 0x54de,
00746 0x551b, 0x54e7, 0x5520, 0x54fd, 0x5514, 0x54f3, 0x5522, 0x5523,
00747 0x550f, 0x5511, 0x5527, 0x552a, 0x5567, 0x558f, 0x55b5, 0x5549,
00748 0x556d, 0x5541, 0x5555, 0x553f, 0x5550, 0x553c,
00749 /* 0x60 */
00750 0x5537, 0x5556, 0x5575, 0x5576, 0x5577, 0x5533, 0x5530, 0x555c,
00751 0x558b, 0x55d2, 0x5583, 0x55b1, 0x55b9, 0x5588, 0x5581, 0x559f,
00752 0x557e, 0x55d6, 0x5591, 0x557b, 0x55df, 0x55bd, 0x55be, 0x5594,
00753 0x5599, 0x55ea, 0x55f7, 0x55c9, 0x561f, 0x55d1, 0x55eb, 0x55ec,
00754 0x55d4, 0x55e6, 0x55dd, 0x55c4, 0x55ef, 0x55e5, 0x55f2, 0x55f3,
00755 0x55cc, 0x55cd, 0x55e8, 0x55f5, 0x55e4, 0x8f94, 0x561e, 0x5608,
00756 0x560c, 0x5601, 0x5624, 0x5623, 0x55fe, 0x5600, 0x5627, 0x562d,
00757 0x5658, 0x5639, 0x5657, 0x562c, 0x564d, 0x5662, 0x5659, 0x565c,
00758 0x564c, 0x565d, 0x5686, 0x5664, 0x5671, 0x566b, 0x567b, 0x567c,
00759 0x5685, 0x5693, 0x56af, 0x56d4, 0x56d7, 0x56dd, 0x56e1, 0x56f5,
00760 0x56eb, 0x56f9, 0x56ff, 0x5704, 0x570a, 0x5709, 0x571c, 0x5e0f,
00761 0x5e19, 0x5e14, 0x5e11, 0x5e31, 0x5e3b, 0x5e3c,
00762 /* 0x61 */
00763 0x5e37, 0x5e44, 0x5e54, 0x5e5b, 0x5e5e, 0x5e61, 0x5c8c, 0x5c7a,
00764 0x5c8d, 0x5c90, 0x5c96, 0x5c88, 0x5c98, 0x5c99, 0x5c91, 0x5c9a,
00765 0x5c9c, 0x5cb5, 0x5ca2, 0x5cbd, 0x5cac, 0x5cab, 0x5cb1, 0x5ca3,
00766 0x5cc1, 0x5cb7, 0x5cc4, 0x5cd2, 0x5ce4, 0x5ccb, 0x5ce5, 0x5d02,
00767 0x5d03, 0x5d27, 0x5d26, 0x5d2e, 0x5d24, 0x5d1e, 0x5d06, 0x5d1b,
00768 0x5d58, 0x5d3e, 0x5d34, 0x5d3d, 0x5d6c, 0x5d5b, 0x5d6f, 0x5d5d,
00769 0x5d6b, 0x5d4b, 0x5d4a, 0x5d69, 0x5d74, 0x5d82, 0x5d99, 0x5d9d,
00770 0x8c73, 0x5db7, 0x5dc5, 0x5f73, 0x5f77, 0x5f82, 0x5f87, 0x5f89,
00771 0x5f8c, 0x5f95, 0x5f99, 0x5f9c, 0x5fa8, 0x5fad, 0x5fb5, 0x5fbc,
00772 0x8862, 0x5f61, 0x72ad, 0x72b0, 0x72b4, 0x72b7, 0x72b8, 0x72c3,
00773 0x72c1, 0x72ce, 0x72cd, 0x72d2, 0x72e8, 0x72ef, 0x72e9, 0x72f2,
00774 0x72f4, 0x72f7, 0x7301, 0x72f3, 0x7303, 0x72fa,
00775 /* 0x62 */
00776 0x72fb, 0x7317, 0x7313, 0x7321, 0x730a, 0x731e, 0x731d, 0x7315,
00777 0x7322, 0x7339, 0x7325, 0x732c, 0x7338, 0x7331, 0x7350, 0x734d,
00778 0x7357, 0x7360, 0x736c, 0x736f, 0x737e, 0x821b, 0x5925, 0x98e7,
00779 0x5924, 0x5902, 0x9963, 0x9967, 0x9968, 0x9969, 0x996a, 0x996b,
00780 0x996c, 0x9974, 0x9977, 0x997d, 0x9980, 0x9984, 0x9987, 0x998a,
00781 0x998d, 0x9990, 0x9991, 0x9993, 0x9994, 0x9995, 0x5e80, 0x5e91,
00782 0x5e8b, 0x5e96, 0x5ea5, 0x5ea0, 0x5eb9, 0x5eb5, 0x5ebe, 0x5eb3,
00783 0x8d53, 0x5ed2, 0x5ed1, 0x5edb, 0x5ee8, 0x5eea, 0x81ba, 0x5fc4,
00784 0x5fc9, 0x5fd6, 0x5fcf, 0x6003, 0x5fee, 0x6004, 0x5fel, 0x5fe4,
00785 0x5ffe, 0x6005, 0x6006, 0x5fea, 0x5fed, 0x5ff8, 0x6019, 0x6035,
00786 0x6026, 0x601b, 0x600f, 0x600d, 0x6029, 0x602b, 0x600a, 0x603f,
00787 0x6021, 0x6078, 0x6079, 0x607b, 0x607a, 0x6042,
00788 /* 0x63 */
00789 0x606a, 0x607d, 0x6096, 0x609a, 0x60ad, 0x609d, 0x6083, 0x6092,
00790 0x608c, 0x609b, 0x60ec, 0x60bb, 0x60b1, 0x60dd, 0x60d8, 0x60c6,
00791 0x60da, 0x60b4, 0x6120, 0x6126, 0x6115, 0x6123, 0x60f4, 0x6100,
00792 0x610e, 0x612b, 0x614a, 0x6175, 0x61ac, 0x6194, 0x61a7, 0x61b7,
00793 0x61d4, 0x61f5, 0x5fdd, 0x96b3, 0x95e9, 0x95eb, 0x95f1, 0x95f3,
00794 0x95f5, 0x95f6, 0x95fc, 0x95fe, 0x9603, 0x9604, 0x9606, 0x9608,
00795 0x960a, 0x960b, 0x960c, 0x960d, 0x960f, 0x9612, 0x9615, 0x9616,
00796 0x9617, 0x9619, 0x961a, 0x4e2c, 0x723f, 0x6215, 0x6c35, 0x6c54,
00797 0x6c5c, 0x6c4a, 0x6ca3, 0x6c85, 0x6c90, 0x6c94, 0x6c8c, 0x6c68,
00798 0x6c69, 0x6c74, 0x6c76, 0x6c86, 0x6ca9, 0x6cd0, 0x6cd4, 0x6cad,
00799 0x6cf7, 0x6cf8, 0x6cf1, 0x6cd7, 0x6cb2, 0x6ce0, 0x6cd6, 0x6cfa,
00800 0x6ceb, 0x6cee, 0x6cb1, 0x6cd3, 0x6cef, 0x6cfe,
00801 /* 0x64 */
00802 0x6d39, 0x6d27, 0x6d0c, 0x6d43, 0x6d48, 0x6d07, 0x6d04, 0x6d19,
00803 0x6d0e, 0x6d2b, 0x6d4d, 0x6d2e, 0x6d35, 0x6d1a, 0x6d4f, 0x6d52,
00804 0x6d54, 0x6d33, 0x6d91, 0x6d6f, 0x6d9e, 0x6da0, 0x6d5e, 0x6d93,
00805 0x6d94, 0x6d5c, 0x6d60, 0x6d7c, 0x6d63, 0x6e1a, 0x6dc7, 0x6dc5,
00806 0x6dde, 0x6e0e, 0x6dbf, 0x6de0, 0x6e11, 0x6de6, 0x6ddd, 0x6dd9,

```



```
00807 0x6e16, 0x6dab, 0x6e0c, 0x6dae, 0x6e2b, 0x6e6e, 0x6e4e, 0x6e6b,
00808 0x6eb2, 0x6e5f, 0x6e86, 0x6e53, 0x6e54, 0x6e32, 0x6e25, 0x6e44,
00809 0x6edf, 0x6eb1, 0x6e98, 0x6ee0, 0x6f2d, 0x6ee2, 0x6ea5, 0x6ea7,
00810 0x6ebd, 0x6ebb, 0x6eb7, 0x6ed7, 0x6eb4, 0x6ecf, 0x6e8f, 0x6ec2,
00811 0x6e9f, 0x6f62, 0x6f46, 0x6f47, 0x6f24, 0x6f15, 0x6ef9, 0x6f2f,
00812 0x6f36, 0x6f4b, 0x6f74, 0x6f2a, 0x6f09, 0x6f29, 0x6f89, 0x6f8d,
00813 0x6f8c, 0x6f78, 0x6f72, 0x6f7c, 0x6f7a, 0x6fd1,
00814 /* 0x65 */
00815 0x6fc9, 0x6fa7, 0x6fb9, 0x6fb6, 0x6fc2, 0x6fe1, 0xfee, 0x6fde,
00816 0x6fe0, 0x6fef, 0x701a, 0x7023, 0x701b, 0x7039, 0x7035, 0x704f,
00817 0x705e, 0x5b80, 0x5b84, 0x5b95, 0x5b93, 0x5ba5, 0x5bb8, 0x752f,
00818 0x9a9e, 0x6434, 0x5be4, 0x5bee, 0x8930, 0x5bf0, 0x8e47, 0x8b07,
00819 0x8fb6, 0x8fd3, 0x8fd5, 0x8fe5, 0x8fee, 0x8fe4, 0x8fe9, 0x8fe6,
00820 0x8ff3, 0x8fe8, 0x9005, 0x9004, 0x900b, 0x9026, 0x9011, 0x900d,
00821 0x9016, 0x9021, 0x9035, 0x9036, 0x902d, 0x902f, 0x9044, 0x9051,
00822 0x9052, 0x9050, 0x9068, 0x9058, 0x9062, 0x905b, 0x66b9, 0x9074,
00823 0x907d, 0x9082, 0x9088, 0x9083, 0x908b, 0x5f50, 0x5f57, 0x5f56,
00824 0x5f58, 0x5c3b, 0x54ab, 0x5c50, 0x5c59, 0x5b71, 0x5c63, 0x5c66,
00825 0x7fbc, 0x5f2a, 0x5f29, 0x5f2d, 0x8274, 0x5f3c, 0x9b3b, 0x5c6e,
00826 0x5981, 0x5983, 0x598d, 0x59a9, 0x59aa, 0x59a3,
00827 /* 0x66 */
00828 0x5997, 0x59ca, 0x59ab, 0x599e, 0x59a4, 0x59d2, 0x59b2, 0x59af,
00829 0x59d7, 0x59be, 0x5a05, 0x5a06, 0x59dd, 0x5a08, 0x59e3, 0x59d8,
00830 0x59f9, 0x5a0c, 0x5a09, 0x5a32, 0x5a34, 0x5a11, 0x5a23, 0x5a13,
00831 0x5a40, 0x5a67, 0x5a4a, 0x5a55, 0x5a3c, 0x5a62, 0x5a75, 0x80ec,
00832 0x5aaa, 0x5a9b, 0x5a77, 0x5a7a, 0x5abe, 0x5aeb, 0x5ab2, 0x5ad2,
00833 0x5ad4, 0x5ab8, 0x5ae0, 0x5ae3, 0x5af1, 0x5ad6, 0x5ae6, 0x5ad8,
00834 0x5adc, 0x5b09, 0x5b17, 0x5b16, 0x5b32, 0x5b37, 0x5b40, 0x5c15,
00835 0x5c1c, 0x5b5a, 0x5b65, 0x5b73, 0x5b51, 0x5b53, 0x5b62, 0x9a75,
00836 0x9a77, 0x9a78, 0x9a7a, 0x9a7f, 0x9a7d, 0x9a80, 0x9a81, 0x9a85,
00837 0x9a88, 0x9a8a, 0x9a90, 0x9a92, 0x9a93, 0x9a96, 0x9a98, 0x9a9b,
00838 0x9a9c, 0x9a9d, 0x9a9f, 0x9aa0, 0x9aa2, 0x9aa3, 0x9aa5, 0x9aa7,
00839 0x7e9f, 0x7ea1, 0x7ea3, 0x7ea5, 0x7ea8, 0x7ea9,
00840 /* 0x67 */
00841 0x7ead, 0x7eb0, 0x7ebe, 0x7ec0, 0x7ec1, 0x7ec2, 0x7ec9, 0x7ecb,
00842 0x7ecc, 0x7ed0, 0x7ed4, 0x7edb, 0x7ee0, 0x7ee1, 0x7ee8,
00843 0x7eeb, 0x7eee, 0x7eef, 0x7ef1, 0x7ef2, 0x7f0d, 0x7ef6, 0x7efa,
00844 0x7efb, 0x7efe, 0x7f01, 0x7f02, 0x7f03, 0x7f07, 0x7f08, 0x7f0b,
00845 0x7f0c, 0x7f0f, 0x7f11, 0x7f12, 0x7f17, 0x7f19, 0x7f1c, 0x7f1b,
00846 0x7f1f, 0x7f21, 0x7f22, 0x7f23, 0x7f24, 0x7f25, 0x7f26, 0x7f27,
00847 0x7f2a, 0x7f2b, 0x7f2c, 0x7f2d, 0x7f2f, 0x7f30, 0x7f31, 0x7f32,
00848 0x7f33, 0x7f35, 0x5e7a, 0x757f, 0x5ddb, 0x753e, 0x9095, 0x738e,
00849 0x7391, 0x73ae, 0x73a2, 0x739f, 0x73cf, 0x73c2, 0x73d1, 0x73b7,
00850 0x73b3, 0x73c0, 0x73c9, 0x73c8, 0x73e5, 0x73d9, 0x987c, 0x740a,
00851 0x73e9, 0x73e7, 0x73de, 0x73ba, 0x73f2, 0x740f, 0x742a, 0x745b,
00852 0x7426, 0x7425, 0x7428, 0x7430, 0x742e, 0x742c,
00853 /* 0x68 */
00854 0x741b, 0x741a, 0x7441, 0x745c, 0x7457, 0x7455, 0x7459, 0x7477,
00855 0x746d, 0x747e, 0x749c, 0x748e, 0x7480, 0x7481, 0x7487, 0x748b,
00856 0x749e, 0x74a8, 0x74a9, 0x7490, 0x74a7, 0x74d2, 0x74ba, 0x97ea,
00857 0x97eb, 0x97ec, 0x674c, 0x6753, 0x675e, 0x6748, 0x6769, 0x67a5,
00858 0x6787, 0x676a, 0x6773, 0x6798, 0x67a7, 0x6775, 0x67a8, 0x679e,
00859 0x67ad, 0x678b, 0x6777, 0x677c, 0x67f0, 0x6809, 0x67d8, 0x680a,
00860 0x67e9, 0x67b0, 0x680c, 0x67d9, 0x67b5, 0x67da, 0x67b3, 0x67dd,
00861 0x6800, 0x67c3, 0x67b8, 0x67e2, 0x680e, 0x67c1, 0x67fd, 0x6832,
00862 0x6833, 0x6860, 0x6861, 0x684e, 0x6862, 0x6844, 0x6864, 0x6883,
00863 0x681d, 0x6855, 0x6866, 0x6841, 0x6867, 0x6840, 0x683e, 0x684a,
00864 0x6849, 0x6829, 0x68b5, 0x688f, 0x6874, 0x6877, 0x6893, 0x686b,
00865 0x68c2, 0x696e, 0x68fc, 0x691f, 0x6920, 0x68f9,
00866 /* 0x69 */
00867 0x6924, 0x68f0, 0x690b, 0x6901, 0x6957, 0x68e3, 0x6910, 0x6971,
00868 0x6939, 0x6960, 0x6942, 0x695d, 0x6984, 0x696b, 0x6980, 0x6998,
00869 0x6978, 0x6934, 0x69cc, 0x6987, 0x6988, 0x69ce, 0x6989, 0x6966,
00870 0x6963, 0x6979, 0x699b, 0x69a7, 0x69bb, 0x69ab, 0x69ad, 0x69d4,
00871 0x69b1, 0x69c1, 0x69ca, 0x69df, 0x6995, 0x69e0, 0x698d, 0x69ff,
00872 0x6a2f, 0x69ed, 0x6a17, 0x6a18, 0x6a65, 0x69f2, 0x6a44, 0x6a3e,
00873 0x6aa0, 0x6a50, 0x6a5b, 0x6a35, 0x6a8e, 0x6a79, 0x6a3d, 0x6a28,
00874 0x6a58, 0x6a7c, 0x6a91, 0x6a90, 0x6aa9, 0x6a97, 0x6aab, 0x7337,
00875 0x7352, 0x6b81, 0x6b82, 0x6b87, 0x6b84, 0x6b92, 0x6b93, 0x6b8d,
00876 0x6b9a, 0x6b9b, 0x6ba1, 0x6baa, 0x8f6b, 0x8f6d, 0x8f71, 0x8f72,
00877 0x8f73, 0x8f75, 0x8f76, 0x8f78, 0x8f77, 0x8f79, 0x8f7a, 0x8f7c,
00878 0x8f7e, 0x8f81, 0x8f82, 0x8f84, 0x8f87, 0x8f8b,
00879 /* 0x6a */
00880 0x8f8d, 0x8f8e, 0x8f8f, 0x8f98, 0x8f9a, 0x8ece, 0x620b, 0x6217,
00881 0x621b, 0x621f, 0x6222, 0x6221, 0x6225, 0x6224, 0x622c, 0x81e7,
00882 0x74ef, 0x74fa, 0x74ff, 0x750f, 0x7511, 0x7513, 0x6534, 0x65ee,
00883 0x65ef, 0x65f0, 0x660a, 0x6619, 0x6772, 0x6603, 0x6615, 0x6600,
00884 0x7085, 0x66f7, 0x661d, 0x6634, 0x6631, 0x6636, 0x6635, 0x8006,
00885 0x665f, 0x6654, 0x6641, 0x664f, 0x6656, 0x6661, 0x6657, 0x6677,
00886 0x6684, 0x668c, 0x66a7, 0x669d, 0x66be, 0x66db, 0x66dc, 0x66e6,
00887 0x66e9, 0x8d32, 0x8d33, 0x8d36, 0x8d3b, 0x8d3d, 0x8d40, 0x8d45,
00888 0x8d46, 0x8d48, 0x8d49, 0x8d47, 0x8d4d, 0x8d55, 0x8d59, 0x89c7,
00889 0x89ca, 0x89cb, 0x89cc, 0x89ce, 0x89cf, 0x89d0, 0x89d1, 0x726e,
00890 0x729f, 0x725d, 0x7266, 0x726f, 0x727e, 0x727f, 0x7284, 0x728b,
00891 0x728d, 0x728f, 0x7292, 0x6308, 0x6332, 0x63b0,
00892 /* 0x6b */
00893 0x643f, 0x64d8, 0x8004, 0x6bea, 0x6bf3, 0x6bfd, 0x6bf5, 0x6bf9,
```

```
00894 0x6c05, 0x6c07, 0x6c06, 0x6c0d, 0x6c15, 0x6c18, 0x6c19, 0x6c1a,
00895 0x6c21, 0x6c29, 0x6c24, 0x6c2a, 0x6c32, 0x6535, 0x6555, 0x656b,
00896 0x724d, 0x7252, 0x7256, 0x7230, 0x8662, 0x5216, 0x809f, 0x809c,
00897 0x8093, 0x80bc, 0x670a, 0x80bd, 0x80b1, 0x80ab, 0x80ad, 0x80b4,
00898 0x80b7, 0x80e7, 0x80e8, 0x80e9, 0x80ea, 0x80db, 0x80c2, 0x80c4,
00899 0x80d9, 0x80cd, 0x80cd, 0x80d7, 0x6710, 0x80dd, 0x80eb, 0x80f1, 0x80f4,
00900 0x80ed, 0x810d, 0x810e, 0x80f2, 0x80fc, 0x6715, 0x8112, 0x8c5a,
00901 0x8136, 0x811e, 0x812c, 0x8118, 0x8132, 0x8148, 0x814c, 0x8153,
00902 0x8174, 0x8159, 0x815a, 0x8171, 0x8160, 0x8169, 0x817c, 0x817d,
00903 0x816d, 0x8167, 0x584d, 0x5ab5, 0x8188, 0x8182, 0x8191, 0x6ed5,
00904 0x81a3, 0x81aa, 0x81cc, 0x6726, 0x81ca, 0x81bb,
00905 /* 0x6c */
00906 0x81c1, 0x81a6, 0x6b24, 0x6b37, 0x6b39, 0x6b43, 0x6b46, 0x6b59,
00907 0x98d1, 0x98d2, 0x98d3, 0x98d5, 0x98d9, 0x98da, 0x6bb3, 0x5f40,
00908 0x6bc2, 0x89f3, 0x6590, 0x9f51, 0x6593, 0x65bc, 0x65c6, 0x65c4,
00909 0x65c3, 0x65cc, 0x65ce, 0x65d2, 0x65d6, 0x7080, 0x709c, 0x7096,
00910 0x709d, 0x70bb, 0x70c0, 0x70b7, 0x70ab, 0x70b1, 0x70e8, 0x70ca,
00911 0x7110, 0x7113, 0x7116, 0x712f, 0x7131, 0x7173, 0x715c, 0x7168,
00912 0x7145, 0x7172, 0x714a, 0x7178, 0x717a, 0x7198, 0x71b3, 0x71b5,
00913 0x71a8, 0x71a0, 0x71e0, 0x71d4, 0x71e7, 0x71f9, 0x721d, 0x7228,
00914 0x706c, 0x7118, 0x7166, 0x71b9, 0x623e, 0x623d, 0x6243, 0x6248,
00915 0x6249, 0x793b, 0x7940, 0x7946, 0x7949, 0x795b, 0x795c, 0x7953,
00916 0x795a, 0x7962, 0x7957, 0x7960, 0x796f, 0x7967, 0x797a, 0x7985,
00917 0x798a, 0x799a, 0x79a7, 0x79b3, 0x5fd1, 0x5fd0,
00918 /* 0x6d */
00919 0x603c, 0x605d, 0x605a, 0x6067, 0x6041, 0x6059, 0x6063, 0x60ab,
00920 0x6106, 0x610d, 0x615d, 0x61a9, 0x619d, 0x61cb, 0x61d1, 0x6206,
00921 0x8080, 0x807f, 0x6c93, 0x6cf6, 0x6dfc, 0x77f6, 0x77f8, 0x7800,
00922 0x7809, 0x7817, 0x7818, 0x7811, 0x65ab, 0x782d, 0x781c, 0x781d,
00923 0x7839, 0x783a, 0x783b, 0x781f, 0x783c, 0x7825, 0x782c, 0x7823,
00924 0x7829, 0x784e, 0x786d, 0x7856, 0x7857, 0x7826, 0x7850, 0x7847,
00925 0x784c, 0x786a, 0x789b, 0x7893, 0x789a, 0x7887, 0x789c, 0x78a1,
00926 0x78a3, 0x78b2, 0x78b9, 0x78a5, 0x78d4, 0x78d9, 0x78c9, 0x78ec,
00927 0x78f2, 0x7905, 0x78f4, 0x7913, 0x7924, 0x791e, 0x7934, 0x9f9b,
00928 0x9ef9, 0x9efb, 0x9efc, 0x76f1, 0x7704, 0x770d, 0x76f9, 0x7707,
00929 0x7708, 0x771a, 0x7722, 0x7719, 0x772d, 0x7726, 0x7735, 0x7738,
00930 0x7750, 0x7751, 0x7747, 0x7743, 0x775a, 0x7768,
00931 /* 0x6e */
00932 0x7762, 0x7765, 0x777f, 0x778d, 0x777d, 0x7780, 0x778c, 0x7791,
00933 0x779f, 0x77a0, 0x77b0, 0x77b5, 0x77bd, 0x753a, 0x7540, 0x754e,
00934 0x754b, 0x7548, 0x755b, 0x7572, 0x7579, 0x7583, 0x7f58, 0x7f61,
00935 0x7f5f, 0x8a48, 0x7f68, 0x7f74, 0x7f71, 0x7f79, 0x7f81, 0x7f7e,
00936 0x76cd, 0x76e5, 0x8832, 0x9485, 0x9486, 0x9487, 0x948b, 0x948a,
00937 0x948c, 0x948d, 0x948f, 0x9490, 0x9494, 0x9497, 0x9495, 0x949a,
00938 0x949b, 0x949c, 0x949a, 0x94a3, 0x94a4, 0x94ab, 0x94aa, 0x94ad, 0x94ac,
00939 0x94af, 0x94b0, 0x94b2, 0x94b4, 0x94b6, 0x94b7, 0x94b8, 0x94b9,
00940 0x94ba, 0x94bc, 0x94bd, 0x94bf, 0x94c4, 0x94c8, 0x94c9, 0x94ca,
00941 0x94cb, 0x94cc, 0x94cd, 0x94ce, 0x94d0, 0x94d1, 0x94d2, 0x94d5,
00942 0x94d6, 0x94d7, 0x94d9, 0x94d8, 0x94db, 0x94de, 0x94df, 0x94e0,
00943 0x94e2, 0x94e4, 0x94e5, 0x94e7, 0x94e8, 0x94ea,
00944 /* 0x6f */
00945 0x94e9, 0x94eb, 0x94ee, 0x94ef, 0x94f3, 0x94f4, 0x94f5, 0x94f7,
00946 0x94f9, 0x94fc, 0x94fd, 0x94ff, 0x9503, 0x9502, 0x9506, 0x9507,
00947 0x9509, 0x950a, 0x950d, 0x950e, 0x950f, 0x9512, 0x9513, 0x9514,
00948 0x9515, 0x9516, 0x9518, 0x951b, 0x951d, 0x951e, 0x951f, 0x9522,
00949 0x952a, 0x952b, 0x9529, 0x952c, 0x9531, 0x9532, 0x9534, 0x9536,
00950 0x9537, 0x9538, 0x953c, 0x953e, 0x953f, 0x9542, 0x9535, 0x9544,
00951 0x9545, 0x9546, 0x9549, 0x954c, 0x954e, 0x954f, 0x9552, 0x9553,
00952 0x9554, 0x9556, 0x9557, 0x9558, 0x9559, 0x955b, 0x955e, 0x955f,
00953 0x955d, 0x9561, 0x9562, 0x9564, 0x9565, 0x9566, 0x9567, 0x9568,
00954 0x9569, 0x956a, 0x956b, 0x956c, 0x956f, 0x9571, 0x9572, 0x9573,
00955 0x953a, 0x77e7, 0x77ec, 0x96c9, 0x79d5, 0x79ed, 0x79e3, 0x79eb,
00956 0x7a06, 0x5d47, 0x7a03, 0x7a02, 0x7a1e, 0x7a14,
00957 /* 0x70 */
00958 0x7a39, 0x7a37, 0x7a51, 0x9ecf, 0x99a5, 0x7a70, 0x7688, 0x768e,
00959 0x7693, 0x7699, 0x76a4, 0x74de, 0x74e0, 0x752c, 0x9e20, 0x9e22,
00960 0x9e28, 0x9e29, 0x9e2a, 0x9e2b, 0x9e2c, 0x9e32, 0x9e31, 0x9e36,
00961 0x9e38, 0x9e37, 0x9e39, 0x9e3a, 0x9e3e, 0x9e41, 0x9e42, 0x9e44,
00962 0x9e46, 0x9e47, 0x9e48, 0x9e49, 0x9e4b, 0x9e4c, 0x9e4e, 0x9e51,
00963 0x9e55, 0x9e57, 0x9e5a, 0x9e5b, 0x9e5c, 0x9e5e, 0x9e63, 0x9e66,
00964 0x9e67, 0x9e68, 0x9e69, 0x9e6a, 0x9e6b, 0x9e6c, 0x9e71, 0x9e6d,
00965 0x9e73, 0x7592, 0x7594, 0x7596, 0x75a0, 0x759d, 0x75ac, 0x75a3,
00966 0x75b3, 0x75b4, 0x75b8, 0x75c4, 0x75b1, 0x75b0, 0x75c3, 0x75c2,
00967 0x75d6, 0x75cd, 0x75e3, 0x75e8, 0x75e6, 0x75e4, 0x75eb, 0x75e7,
00968 0x7603, 0x75f1, 0x75fc, 0x75ff, 0x7610, 0x7600, 0x7605, 0x760c,
00969 0x7617, 0x760a, 0x7625, 0x7618, 0x7615, 0x7619,
00970 /* 0x71 */
00971 0x761b, 0x763c, 0x7622, 0x7620, 0x7640, 0x762d, 0x7630, 0x763f,
00972 0x7635, 0x7643, 0x763e, 0x7633, 0x764d, 0x765e, 0x7654, 0x765c,
00973 0x7656, 0x766b, 0x766f, 0x766f, 0x76ae, 0x7a78, 0x7a79, 0x7a80,
00974 0x7a86, 0x7a88, 0x7a95, 0x7aa6, 0x7aa0, 0x7aac, 0x7aa8, 0x7aad,
00975 0x7ab3, 0x8864, 0x8869, 0x8872, 0x887d, 0x887f, 0x8882, 0x88a2,
00976 0x88c6, 0x88b7, 0x88bc, 0x88c9, 0x88e2, 0x88ce, 0x88e3, 0x88e5,
00977 0x88f1, 0x891a, 0x88fc, 0x88e8, 0x88fe, 0x88f0, 0x8921, 0x8919,
00978 0x8913, 0x891b, 0x890a, 0x8934, 0x892b, 0x8936, 0x8941, 0x8966,
00979 0x897b, 0x758b, 0x80e5, 0x76b2, 0x76b4, 0x77dc, 0x8012, 0x8014,
00980 0x8016, 0x801c, 0x8020, 0x8022, 0x8025, 0x8026, 0x8027, 0x8029,
```

```
00981 0x8028, 0x8031, 0x800b, 0x8035, 0x8043, 0x8046, 0x804d, 0x8052,
00982 0x8069, 0x8071, 0x8983, 0x9878, 0x9880, 0x9883,
00983 /* 0x72 */
00984 0x9889, 0x988c, 0x988d, 0x988f, 0x9894, 0x989a, 0x989b, 0x989e,
00985 0x989f, 0x98a1, 0x98a2, 0x98a5, 0x98a6, 0x864d, 0x8654, 0x866c,
00986 0x866e, 0x867f, 0x867a, 0x867c, 0x867b, 0x86a8, 0x868d, 0x868b,
00987 0x86ac, 0x869d, 0x86a7, 0x86a3, 0x86aa, 0x8693, 0x86a9, 0x86b6,
00988 0x86c4, 0x86b5, 0x86ce, 0x86b0, 0x86ba, 0x86b1, 0x86af, 0x86c9,
00989 0x86cf, 0x86b4, 0x86e9, 0x86f1, 0x86f2, 0x86ed, 0x86f3, 0x86d0,
00990 0x8713, 0x86de, 0x86f4, 0x86df, 0x86d8, 0x86d1, 0x8703, 0x8707,
00991 0x86f8, 0x8708, 0x870a, 0x870d, 0x8709, 0x8723, 0x873b, 0x871e,
00992 0x8725, 0x872e, 0x871a, 0x873e, 0x8748, 0x8734, 0x8731, 0x8729,
00993 0x8737, 0x873f, 0x8782, 0x8722, 0x877d, 0x877e, 0x877b, 0x8760,
00994 0x8770, 0x874c, 0x876e, 0x878b, 0x8753, 0x8763, 0x877c, 0x8764,
00995 0x8759, 0x8765, 0x8793, 0x87af, 0x87a8, 0x87d2,
00996 /* 0x73 */
00997 0x87c6, 0x8788, 0x8785, 0x87ad, 0x8797, 0x8783, 0x87ab, 0x87e5,
00998 0x87ac, 0x87b5, 0x87b3, 0x87cb, 0x87d3, 0x87bd, 0x87d1, 0x87c0,
00999 0x87ca, 0x87db, 0x87ea, 0x87e0, 0x87ee, 0x8816, 0x8813, 0x87fe,
01000 0x880a, 0x881b, 0x8821, 0x8839, 0x883c, 0x7f36, 0x7f42, 0x7f44,
01001 0x7f45, 0x8210, 0x7afa, 0x7afd, 0x7b08, 0x7b03, 0x7b04, 0x7b15,
01002 0x7b0a, 0x7b2b, 0x7b0f, 0x7b47, 0x7b38, 0x7b2a, 0x7b19, 0x7b2e,
01003 0x7b31, 0x7b20, 0x7b25, 0x7b24, 0x7b33, 0x7b3e, 0x7b1e, 0x7b58,
01004 0x7b5a, 0x7b45, 0x7b75, 0x7b4c, 0x7b5d, 0x7b60, 0x7b6e, 0x7b7b,
01005 0x7b62, 0x7b72, 0x7b71, 0x7b90, 0x7ba6, 0x7ba7, 0x7bb8, 0x7bac,
01006 0x7b9d, 0x7ba8, 0x7b85, 0x7baa, 0x7b9c, 0x7ba2, 0x7bab, 0x7bb4,
01007 0x7bd1, 0x7bcl, 0x7bcc, 0x7bdd, 0x7bda, 0x7be5, 0x7be6, 0x7bea,
01008 0x7c0c, 0x7bfe, 0x7bfc, 0x7c0f, 0x7c16, 0x7c0b,
01009 /* 0x74 */
01010 0x7c1f, 0x7c2a, 0x7c26, 0x7c38, 0x7c41, 0x7c40, 0x81fe, 0x8201,
01011 0x8202, 0x8204, 0x81ec, 0x8844, 0x8221, 0x8222, 0x8223, 0x822d,
01012 0x822f, 0x8228, 0x822b, 0x8238, 0x823b, 0x8233, 0x8234, 0x823e,
01013 0x8244, 0x8249, 0x824b, 0x824f, 0x825a, 0x825f, 0x8268, 0x887e,
01014 0x8885, 0x8888, 0x88d8, 0x88df, 0x895e, 0x7f9d, 0x7f9f, 0x7fa7,
01015 0x7faf, 0x7fb0, 0x7fb2, 0x7c7c, 0x6549, 0x7c91, 0x7c9d, 0x7c9c,
01016 0x7c9e, 0x7ca2, 0x7cb2, 0x7cbc, 0x7cbd, 0x7cc1, 0x7cc7, 0x7ccc,
01017 0x7ccd, 0x7cc8, 0x7cc5, 0x7cd7, 0x7ce8, 0x826e, 0x66a8, 0x7fbf,
01018 0x7fce, 0x7fd5, 0x7fe5, 0x7fel, 0x7fe6, 0x7fe9, 0x7fee, 0x7ff3,
01019 0x7cf8, 0x7d77, 0x7da6, 0x7dae, 0x7e47, 0x7e9b, 0x9eb8, 0x9eb4,
01020 0x8d73, 0x8d84, 0x8d94, 0x8d91, 0x8db1, 0x8d67, 0x8d6d, 0x8c47,
01021 0x8c49, 0x914a, 0x9150, 0x914e, 0x914f, 0x9164,
01022 /* 0x75 */
01023 0x9162, 0x9161, 0x9170, 0x9169, 0x916f, 0x917d, 0x917e, 0x9172,
01024 0x9174, 0x9179, 0x918c, 0x9185, 0x9190, 0x918d, 0x9191, 0x91a2,
01025 0x91a3, 0x91aa, 0x91ad, 0x91ae, 0x91af, 0x91b5, 0x91b4, 0x91ba,
01026 0x8c55, 0x9e7e, 0x8db8, 0x8deb, 0x8e05, 0x8e59, 0x8e69, 0x8db5,
01027 0x8dbf, 0x8dbc, 0x8dba, 0x8dc4, 0x8dd6, 0x8dd7, 0x8dda, 0x8dde,
01028 0x8dce, 0x8dcf, 0x8ddb, 0x8dc6, 0x8dec, 0x8df7, 0x8df8, 0x8de3,
01029 0x8df9, 0x8dfb, 0x8de4, 0x8e09, 0x8dfd, 0x8e14, 0x8e1d, 0x8e1f,
01030 0x8e2c, 0x8e2e, 0x8e23, 0x8e2f, 0x8e3a, 0x8e40, 0x8e39, 0x8e35,
01031 0x8e3d, 0x8e31, 0x8e49, 0x8e41, 0x8e42, 0x8e51, 0x8e52, 0x8e4a,
01032 0x8e70, 0x8e76, 0x8e7c, 0x8e6f, 0x8e74, 0x8e85, 0x8e8f, 0x8e94,
01033 0x8e90, 0x8e9c, 0x8e9e, 0x8c78, 0x8c82, 0x8c8a, 0x8c85, 0x8c98,
01034 0x8c94, 0x659b, 0x89d6, 0x89de, 0x89da, 0x89dc,
01035 /* 0x76 */
01036 0x89e5, 0x89eb, 0x89ef, 0x8a3e, 0x8b26, 0x9753, 0x96e9, 0x96f3,
01037 0x96ef, 0x9706, 0x9701, 0x9708, 0x970f, 0x970e, 0x972a, 0x972d,
01038 0x9730, 0x973e, 0x9f80, 0x9f83, 0x9f85, 0x9f86, 0x9f87, 0x9f88,
01039 0x9f89, 0x9f8a, 0x9f8c, 0x9efe, 0x9f0b, 0x9f0d, 0x96b9, 0x96bc,
01040 0x96bd, 0x96ce, 0x96d2, 0x77bf, 0x96e0, 0x928e, 0x92ae, 0x92c8,
01041 0x933e, 0x936a, 0x93ca, 0x938f, 0x943e, 0x946b, 0x9c7f, 0x9c82,
01042 0x9c85, 0x9c86, 0x9c87, 0x9c88, 0x7a23, 0x9c8b, 0x9c8e, 0x9c90,
01043 0x9c91, 0x9c92, 0x9c94, 0x9c95, 0x9c9a, 0x9c9b, 0x9c9e, 0x9c9f,
01044 0x9ca0, 0x9ca1, 0x9ca2, 0x9ca3, 0x9ca5, 0x9ca6, 0x9ca7, 0x9ca8,
01045 0x9ca9, 0x9cab, 0x9cad, 0x9cae, 0x9cb0, 0x9cb1, 0x9cb2, 0x9cb3,
01046 0x9cb4, 0x9cb5, 0x9cb6, 0x9cb7, 0x9cba, 0x9cbb, 0x9cbb, 0x9cbd,
01047 0x9cc4, 0x9cc5, 0x9cc6, 0x9cc7, 0x9cca, 0x9ccb,
01048 /* 0x77 */
01049 0x9ccc, 0x9ccd, 0x9cce, 0x9ccf, 0x9cd0, 0x9cd3, 0x9cd4, 0x9cd5,
01050 0x9cd7, 0x9cd8, 0x9cd9, 0x9cdc, 0x9cdd, 0x9cdf, 0x9ce2, 0x977c,
01051 0x9785, 0x9791, 0x9792, 0x9794, 0x97af, 0x97ab, 0x97a3, 0x97b2,
01052 0x97b4, 0x9ab1, 0x9ab0, 0x9ab7, 0x9e58, 0x9ab6, 0x9aba, 0x9abc,
01053 0x9ac1, 0x9ac0, 0x9ac5, 0x9ac2, 0x9acb, 0x9acc, 0x9ad1, 0x9b45,
01054 0x9b43, 0x9b47, 0x9b49, 0x9b48, 0x9b4d, 0x9b51, 0x98e8, 0x99d0,
01055 0x992e, 0x9955, 0x9954, 0x9adf, 0x9ae1, 0x9ae6, 0x9aef, 0x9aeb,
01056 0x9afb, 0x9aed, 0x9af9, 0x9b08, 0x9b0f, 0x9b13, 0x9b1f, 0x9b23,
01057 0x9ebd, 0x9ebe, 0x7e3b, 0x9e82, 0x9e87, 0x9e88, 0x9e8b, 0x9e92,
01058 0x93d6, 0x9e9d, 0x9e9f, 0x9edb, 0x9edc, 0x9edd, 0x9eed, 0x9edf,
01059 0x9ee2, 0x9ee9, 0x9ee7, 0x9ee5, 0x9eea, 0x9eef, 0x9f22, 0x9f2c,
01060 0x9f2f, 0x9f39, 0x9f37, 0x9f3d, 0x9f3e, 0x9f44,
01061 };
01062
01063 static int
01064 gb2312_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
01065 {
01066     unsigned char c1 = (s[0] & 0x7f);
01067     if ((c1 >= 0x21 && c1 <= 0x29) || (c1 >= 0x30 && c1 <= 0x77)) {
```

```
01068     if (n >= 2) {
01069         unsigned char c2 = (s[1] & 0x7F);
01070         if (c2 >= 0x21 && c2 < 0x7f) {
01071             unsigned int i = 94 * (c1 - 0x21) + (c2 - 0x21);
01072             unsigned short wc = 0xffffd;
01073             if (i < 1410) {
01074                 if (i < 831)
01075                     wc = gb2312_2uni_page21[i];
01076             } else {
01077                 if (i < 8178)
01078                     wc = gb2312_2uni_page30[i-1410];
01079             }
01080             if (wc != 0xffffd) {
01081                 *pwc = (ucs4_t) wc;
01082                 return 2;
01083             }
01084         }
01085         return RET_ILSEQ;
01086     }
01087     return RET_TOOFEW(0);
01088 }
01089 return RET_ILSEQ;
01090 }
01091 #endif /* NEED_TOWC */
01092
01093 #ifndef NEED_TOMB
01094 static const unsigned short gb2312_2charset[7445] = {
01095     0x2168, 0x216c, 0x2127, 0x2163, 0x2140, 0x2141, 0x2824, 0x2822,
01096     0x2828, 0x2826, 0x283a, 0x282c, 0x282a, 0x2830, 0x282e, 0x2142,
01097     0x2834, 0x2832, 0x2839, 0x2821, 0x2825, 0x2827, 0x2829, 0x282d,
01098     0x2831, 0x2823, 0x282b, 0x282f, 0x2833, 0x2835, 0x2836, 0x2837,
01099     0x2838, 0x2126, 0x2125, 0x2621, 0x2622, 0x2623, 0x2624, 0x2625,
01100     0x2626, 0x2627, 0x2628, 0x2629, 0x262a, 0x262b, 0x262c, 0x262d,
01101     0x262e, 0x262f, 0x2630, 0x2631, 0x2632, 0x2633, 0x2634, 0x2635,
01102     0x2636, 0x2637, 0x2638, 0x2641, 0x2642, 0x2643, 0x2644, 0x2645,
01103     0x2646, 0x2647, 0x2648, 0x2649, 0x264a, 0x264b, 0x264c, 0x264d,
01104     0x264e, 0x264f, 0x2650, 0x2651, 0x2652, 0x2653, 0x2654, 0x2655,
01105     0x2656, 0x2657, 0x2658, 0x2727, 0x2721, 0x2722, 0x2723, 0x2724,
01106     0x2725, 0x2726, 0x2728, 0x2729, 0x272a, 0x272b, 0x272c, 0x272d,
01107     0x272e, 0x272f, 0x2730, 0x2731, 0x2732, 0x2733, 0x2734, 0x2735,
01108     0x2736, 0x2737, 0x2738, 0x2739, 0x273a, 0x273b, 0x273c, 0x273d,
01109     0x273e, 0x273f, 0x2740, 0x2741, 0x2751, 0x2752, 0x2753, 0x2754,
01110     0x2755, 0x2756, 0x2758, 0x2759, 0x275a, 0x275b, 0x275c, 0x275d,
01111     0x275e, 0x275f, 0x2760, 0x2761, 0x2762, 0x2763, 0x2764, 0x2765,
01112     0x2766, 0x2767, 0x2768, 0x2769, 0x276a, 0x276b, 0x276c, 0x276d,
01113     0x276e, 0x276f, 0x2770, 0x2771, 0x2757, 0x212a, 0x212c, 0x212e,
01114     0x212f, 0x2130, 0x2131, 0x212d, 0x216b, 0x2164, 0x2165, 0x2179,
01115     0x2166, 0x216d, 0x2271, 0x2272, 0x2273, 0x2274, 0x2275, 0x2276,
01116     0x2277, 0x2278, 0x2279, 0x227a, 0x227b, 0x227c, 0x217b, 0x217c,
01117     0x217a, 0x217d, 0x214a, 0x2147, 0x2146, 0x214c, 0x2158, 0x215e,
01118     0x214f, 0x214e, 0x2144, 0x2145, 0x2149, 0x2148, 0x2152, 0x2153,
01119     0x2160, 0x215f, 0x2143, 0x214b, 0x2157, 0x2156, 0x2155, 0x2159,
01120     0x2154, 0x215c, 0x215d, 0x215a, 0x215b, 0x2151, 0x214d, 0x2150,
01121     0x2259, 0x225a, 0x225b, 0x225c, 0x225d, 0x225e, 0x225f, 0x2260,
01122     0x2261, 0x2262, 0x2245, 0x2246, 0x2247, 0x2248, 0x2249, 0x224a,
01123     0x224b, 0x224c, 0x224d, 0x224e, 0x224f, 0x2250, 0x2251, 0x2252,
01124     0x2253, 0x2254, 0x2255, 0x2256, 0x2257, 0x2258, 0x2231, 0x2232,
01125     0x2233, 0x2234, 0x2235, 0x2236, 0x2237, 0x2238, 0x2239, 0x223a,
01126     0x223b, 0x223c, 0x223d, 0x223e, 0x223f, 0x2240, 0x2241, 0x2242,
01127     0x2243, 0x2244, 0x2924, 0x2925, 0x2926, 0x2927, 0x2928, 0x2929,
01128     0x292a, 0x292b, 0x292c, 0x292d, 0x292e, 0x292f, 0x2930, 0x2931,
01129     0x2932, 0x2933, 0x2934, 0x2935, 0x2936, 0x2937, 0x2938, 0x2939,
01130     0x293a, 0x293b, 0x293c, 0x293d, 0x293e, 0x293f, 0x2940, 0x2941,
01131     0x2942, 0x2943, 0x2944, 0x2945, 0x2946, 0x2947, 0x2948, 0x2949,
01132     0x294a, 0x294b, 0x294c, 0x294d, 0x294e, 0x294f, 0x2950, 0x2951,
01133     0x2952, 0x2953, 0x2954, 0x2955, 0x2956, 0x2957, 0x2958, 0x2959,
01134     0x295a, 0x295b, 0x295c, 0x295d, 0x295e, 0x295f, 0x2960, 0x2961,
01135     0x2962, 0x2963, 0x2964, 0x2965, 0x2966, 0x2967, 0x2968, 0x2969,
01136     0x296a, 0x296b, 0x296c, 0x296d, 0x296e, 0x296f, 0x2176, 0x2175,
01137     0x2178, 0x2177, 0x2174, 0x2173, 0x2170, 0x2172, 0x2171, 0x216f,
01138     0x216e, 0x2162, 0x2161, 0x2121, 0x2122, 0x2123, 0x2128, 0x2129,
01139     0x2134, 0x2135, 0x2136, 0x2137, 0x2138, 0x2139, 0x213a, 0x213b,
01140     0x213e, 0x213f, 0x217e, 0x2132, 0x2133, 0x213c, 0x213d, 0x2421,
01141     0x2422, 0x2423, 0x2424, 0x2425, 0x2426, 0x2427, 0x2428, 0x2429,
01142     0x242a, 0x242b, 0x242c, 0x242d, 0x242e, 0x242f, 0x2430, 0x2431,
01143     0x2432, 0x2433, 0x2434, 0x2435, 0x2436, 0x2437, 0x2438, 0x2439,
01144     0x243a, 0x243b, 0x243c, 0x243d, 0x243e, 0x243f, 0x2440, 0x2441,
01145     0x2442, 0x2443, 0x2444, 0x2445, 0x2446, 0x2447, 0x2448, 0x2449,
01146     0x244a, 0x244b, 0x244c, 0x244d, 0x244e, 0x244f, 0x2450, 0x2451,
01147     0x2452, 0x2453, 0x2454, 0x2455, 0x2456, 0x2457, 0x2458, 0x2459,
01148     0x245a, 0x245b, 0x245c, 0x245d, 0x245e, 0x245f, 0x2460, 0x2461,
01149     0x2462, 0x2463, 0x2464, 0x2465, 0x2466, 0x2467, 0x2468, 0x2469,
01150     0x246a, 0x246b, 0x246c, 0x246d, 0x246e, 0x246f, 0x2470, 0x2471,
01151     0x2472, 0x2473, 0x2521, 0x2522, 0x2523, 0x2524, 0x2525, 0x2526,
01152     0x2527, 0x2528, 0x2529, 0x252a, 0x252b, 0x252c, 0x252d, 0x252e,
01153     0x252f, 0x2530, 0x2531, 0x2532, 0x2533, 0x2534, 0x2535, 0x2536,
01154     0x2537, 0x2538, 0x2539, 0x253a, 0x253b, 0x253c, 0x253d, 0x253e,
```

01155 0x253f, 0x2540, 0x2541, 0x2542, 0x2543, 0x2544, 0x2545, 0x2546,
01156 0x2547, 0x2548, 0x2549, 0x254a, 0x254b, 0x254c, 0x254d, 0x254e,
01157 0x254f, 0x2550, 0x2551, 0x2552, 0x2553, 0x2554, 0x2555, 0x2556,
01158 0x2557, 0x2558, 0x2559, 0x255a, 0x255b, 0x255c, 0x255d, 0x255e,
01159 0x255f, 0x2560, 0x2561, 0x2562, 0x2563, 0x2564, 0x2565, 0x2566,
01160 0x2567, 0x2568, 0x2569, 0x256a, 0x256b, 0x256c, 0x256d, 0x256e,
01161 0x256f, 0x2570, 0x2571, 0x2572, 0x2573, 0x2574, 0x2575, 0x2576,
01162 0x2124, 0x2845, 0x2846, 0x2847, 0x2848, 0x2849, 0x284a, 0x284b,
01163 0x284c, 0x284d, 0x284e, 0x284f, 0x2850, 0x2851, 0x2852, 0x2853,
01164 0x2854, 0x2855, 0x2856, 0x2857, 0x2858, 0x2859, 0x285a, 0x285b,
01165 0x285c, 0x285d, 0x285e, 0x285f, 0x2860, 0x2861, 0x2862, 0x2863,
01166 0x2864, 0x2865, 0x2866, 0x2867, 0x2868, 0x2869, 0x2265, 0x2266,
01167 0x2267, 0x2268, 0x2269, 0x226a, 0x226b, 0x226c, 0x226d, 0x226e,
01168 0x523b, 0x3621, 0x465f, 0x4d72, 0x5549, 0x487d, 0x494f, 0x4f42,
01169 0x5822, 0x323b, 0x536b, 0x5824, 0x3373, 0x5728, 0x4752, 0x5827,
01170 0x4a40, 0x4770, 0x317b, 0x5235, 0x3454, 0x362b, 0x4b3f, 0x5829,
01171 0x362a, 0x413d, 0x514f, 0x4925, 0x582d, 0x3876, 0x513e, 0x635c,
01172 0x5650, 0x3761, 0x342e, 0x4159, 0x583c, 0x4d68, 0x3524, 0x4e2a,
01173 0x5677, 0x4076, 0x3e59, 0x582f, 0x444b, 0x3e43, 0x5831, 0x4334,
01174 0x5265, 0x562e, 0x4e5a, 0x5527, 0x3a75, 0x3726, 0x4056, 0x4639,
01175 0x4552, 0x4747, 0x3954, 0x334b, 0x5252, 0x583f, 0x3e45, 0x4672,
01176 0x5232, 0x4f30, 0x4f67, 0x4a69, 0x5840, 0x4272, 0x4252, 0x4869,
01177 0x472c, 0x414b, 0x5368, 0x5579, 0x4a42, 0x367e, 0x5821, 0x535a,
01178 0x3f77, 0x5446, 0x3b25, 0x5841, 0x4e65, 0x3e2e, 0x5828, 0x5147,
01179 0x5029, 0x583d, 0x596f, 0x4d76, 0x3f3a, 0x3d3b, 0x3a25, 0x5260,
01180 0x327a, 0x3a60, 0x4436, 0x4f6d, 0x3e29, 0x4d24, 0x4141, 0x4757,
01181 0x5971, 0x5974, 0x484b, 0x5869, 0x525a, 0x4a32, 0x484a, 0x586c,
01182 0x586a, 0x5846, 0x3d76, 0x464d, 0x3370, 0x586b, 0x3d71, 0x3d69,
01183 0x4854, 0x3453, 0x4258, 0x3256, 0x5750, 0x4a4b, 0x4b7b, 0x554c,
01184 0x3836, 0x4f49, 0x595a, 0x5870, 0x472a, 0x586e, 0x347a, 0x416e,
01185 0x5254, 0x586d, 0x5247, 0x586f, 0x4347, 0x5176, 0x5659, 0x5872,
01186 0x5875, 0x3c7e, 0x3c5b, 0x484e, 0x375d, 0x3742, 0x4673, 0x5878,
01187 0x5241, 0x4e69, 0x3c3f, 0x377c, 0x3725, 0x505d, 0x565a, 0x5345,
01188 0x3b6f, 0x3b61, 0x5871, 0x4921, 0x4e30, 0x342b, 0x5873, 0x494b,
01189 0x5876, 0x4257, 0x5877, 0x4e31, 0x5879, 0x322e, 0x3940, 0x5923,
01190 0x3069, 0x4166, 0x496c, 0x4b45, 0x4b46, 0x5924, 0x3568, 0x352b,
01191 0x4e3b, 0x354d, 0x5721, 0x5774, 0x5353, 0x4c65, 0x3a4e, 0x5922,
01192 0x595c, 0x5360, 0x587d, 0x3770, 0x5777, 0x587e, 0x587a, 0x5921,
01193 0x4463, 0x5336, 0x5874, 0x595d, 0x587b, 0x4565, 0x4050, 0x5170,
01194 0x305b, 0x3c51, 0x5926, 0x5925, 0x592c, 0x592e, 0x592b, 0x4a39,
01195 0x5929, 0x5636, 0x335e, 0x5928, 0x407d, 0x4a4c, 0x592a, 0x5927,
01196 0x5930, 0x3631, 0x3929, 0x5240, 0x4f40, 0x4242, 0x3d44, 0x556c,
01197 0x3260, 0x4748, 0x3f6b, 0x592d, 0x592f, 0x4e6a, 0x3a6e, 0x4756,
01198 0x3163, 0x3459, 0x366d, 0x5934, 0x3f21, 0x595e, 0x474e, 0x407e,
01199 0x5938, 0x4b57, 0x377d, 0x5935, 0x5937, 0x3123, 0x5361, 0x5939,
01200 0x5045, 0x5936, 0x5931, 0x5932, 0x4129, 0x5933, 0x3c73, 0x505e,
01201 0x3829, 0x3e63, 0x593d, 0x593a, 0x3033, 0x5942, 0x5944, 0x3136,
01202 0x593f, 0x3539, 0x3e73, 0x4c48, 0x3a72, 0x5250, 0x5943, 0x3d68,
01203 0x332b, 0x5945, 0x3e6b, 0x5946, 0x593b, 0x445f, 0x593e, 0x5941,
01204 0x5940, 0x552e, 0x5635, 0x4763, 0x5948, 0x3c59, 0x594a, 0x593c,
01205 0x594b, 0x462b, 0x5776, 0x5949, 0x5776, 0x4d23, 0x3d21, 0x594c, 0x453c,
01206 0x4d35, 0x594d, 0x5947, 0x3325, 0x3f7e, 0x3835, 0x407c, 0x3078,
01207 0x3476, 0x594e, 0x594f, 0x3422, 0x5950, 0x345f, 0x3041, 0x5951,
01208 0x4935, 0x4f71, 0x5952, 0x4145, 0x5956, 0x492e, 0x5955, 0x5954,
01209 0x5957, 0x4b5b, 0x3d29, 0x4627, 0x5953, 0x5958, 0x5959, 0x4865,
01210 0x405c, 0x3679, 0x5823, 0x544a, 0x542a, 0x5056, 0x3364, 0x5557,
01211 0x4f48, 0x3962, 0x3f4b, 0x4362, 0x3652, 0x4d43, 0x596e, 0x5970,
01212 0x3533, 0x3635, 0x3e24, 0x486b, 0x482b, 0x304b, 0x392b, 0x4179,
01213 0x5962, 0x403c, 0x3932, 0x3958, 0x504b, 0x3178, 0x4664, 0x3e5f,
01214 0x3564, 0x5748, 0x5178, 0x3c66, 0x4a5e, 0x3c3d, 0x5966, 0x5867,
01215 0x445a, 0x3854, 0x483d, 0x3261, 0x5459, 0x4330, 0x4361, 0x5a22,
01216 0x485f, 0x5034, 0x3e7c, 0x4529, 0x395a, 0x5a23, 0x5429, 0x5a24,
01217 0x597b, 0x362c, 0x376b, 0x3179, 0x597c, 0x3365, 0x3e76, 0x3f76,
01218 0x5231, 0x4064, 0x3633, 0x597e, 0x597d, 0x3e3b, 0x4660, 0x573c,
01219 0x5a21, 0x4139, 0x3572, 0x4168, 0x3c75, 0x3455, 0x415d, 0x447d,
01220 0x3c38, 0x3732, 0x376f, 0x596c, 0x463e, 0x3f2d, 0x3b4b, 0x354a,
01221 0x5b49, 0x5057, 0x4d39, 0x303c, 0x3376, 0x3b77, 0x5b4a, 0x3a2f,
01222 0x5464, 0x3536, 0x3573, 0x5856, 0x4850, 0x3756, 0x4750, 0x5857,
01223 0x3f2f, 0x5b3b, 0x5858, 0x504c, 0x3b2e, 0x6b3e, 0x4150, 0x4175,
01224 0x5472, 0x3855, 0x3434, 0x3375, 0x493e, 0x4550, 0x4559, 0x407b,
01225 0x3170, 0x5859, 0x394e, 0x353d, 0x585a, 0x5646, 0x4b22, 0x482f,
01226 0x4932, 0x344c, 0x3f4c, 0x3974, 0x585b, 0x585c, 0x3667, 0x3c41,
01227 0x4c6a, 0x4f77, 0x585d, 0x4730, 0x3950, 0x3d23, 0x4c5e, 0x464a,
01228 0x5860, 0x585e, 0x585f, 0x307e, 0x3e67, 0x4a23, 0x3c74, 0x3831,
01229 0x386e, 0x5862, 0x3d4b, 0x5864, 0x5863, 0x457c, 0x5865, 0x5866,
01230 0x4126, 0x4830, 0x306c, 0x3926, 0x3c53, 0x4e71, 0x5b3d, 0x4153,
01231 0x362f, 0x567a, 0x452c, 0x3d59, 0x5b3e, 0x5b3f, 0x4078, 0x3e22,
01232 0x404d, 0x5b40, 0x4a46, 0x322a, 0x5342, 0x4363, 0x512b, 0x5b42,
01233 0x4055, 0x5b43, 0x3f31, 0x443c, 0x475a, 0x5b44, 0x5968, 0x4957,
01234 0x3934, 0x4e70, 0x5448, 0x307c, 0x3452, 0x5059, 0x5969, 0x5e4b,
01235 0x596b, 0x5830, 0x3b2f, 0x3131, 0x3357, 0x584e, 0x5451, 0x3d33,
01236 0x3f6f, 0x4f3b, 0x5850, 0x374b, 0x5851, 0x4625, 0x4778, 0x523d,
01237 0x5852, 0x4464, 0x4a2e, 0x4727, 0x5826, 0x497d, 0x4e67, 0x3b5c,
01238 0x306b, 0x3b2a, 0x502d, 0x3130, 0x5764, 0x573f, 0x3525, 0x4274,
01239 0x444f, 0x3229, 0x3237, 0x3165, 0x5f32, 0x553c, 0x3f28, 0x422c,
01240 0x5855, 0x4231, 0x5854, 0x4e54, 0x5a60, 0x4e40, 0x5834, 0x432e,
01241 0x5321, 0x4e23, 0x3c34, 0x4834, 0x4251, 0x3e6d, 0x5036, 0x5a61,

```
01242 0x4764, 0x3327, 0x3672, 0x4c7c, 0x407a, 0x4077, 0x5139, 0x5161,
01243 0x5847, 0x325e, 0x4065, 0x3a71, 0x5848, 0x542d, 0x4f61, 0x5849,
01244 0x584a, 0x4f43, 0x3378, 0x3e47, 0x584b, 0x5b4c, 0x4825, 0x4f58,
01245 0x487e, 0x324e, 0x5356, 0x3266, 0x3c30, 0x5351, 0x4b2b, 0x3734,
01246 0x3722, 0x4a65, 0x4821, 0x4a5c, 0x3164, 0x5070, 0x4551, 0x5b45,
01247 0x357e, 0x3f5a, 0x3945, 0x3e64, 0x416d, 0x5f36, 0x5f35, 0x563b,
01248 0x3d50, 0x5559, 0x3048, 0x3623, 0x3f49, 0x4c28, 0x5f33, 0x4a37,
01249 0x5352, 0x584f, 0x5236, 0x3a45, 0x4b3e, 0x4c3e, 0x5f37, 0x3570,
01250 0x5f34, 0x5375, 0x3354, 0x3877, 0x5f3a, 0x3a4f, 0x3c2a, 0x3575,
01251 0x4d2c, 0x437b, 0x3a73, 0x4074, 0x4d42, 0x4f72, 0x5f38, 0x4f45,
01252 0x4240, 0x5f39, 0x4270, 0x3e7d, 0x415f, 0x4d4c, 0x5277, 0x374d,
01253 0x5f41, 0x5f44, 0x3771, 0x3049, 0x3656, 0x3754, 0x3a2c, 0x4c7d,
01254 0x3f54, 0x4b31, 0x4674, 0x5628, 0x5f45, 0x4e62, 0x3333, 0x4e7c,
01255 0x3435, 0x4e47, 0x3a70, 0x4e61, 0x513d, 0x5f40, 0x3474, 0x334a,
01256 0x3866, 0x5f3b, 0x4445, 0x5f3c, 0x5f3d, 0x5f3e, 0x453b, 0x5f3f,
01257 0x5f42, 0x5431, 0x5f43, 0x473a, 0x4e58, 0x4458, 0x5f4a, 0x5f4f,
01258 0x565c, 0x5f49, 0x5f5a, 0x4e36, 0x3a47, 0x5f4e, 0x5f48, 0x455e,
01259 0x496b, 0x3a74, 0x437c, 0x3e57, 0x5f46, 0x5f4d, 0x4558, 0x5526,
01260 0x3a4d, 0x3e4c, 0x533d, 0x3840, 0x5664, 0x5f47, 0x393e, 0x3f27,
01261 0x417c, 0x5f4b, 0x5f4c, 0x5f50, 0x5f5b, 0x5f65, 0x5f57, 0x5f56,
01262 0x5749, 0x5f63, 0x5f64, 0x5f64, 0x656b, 0x5227, 0x5f52, 0x3f29, 0x545b,
01263 0x3f48, 0x5f54, 0x4f4c, 0x5f5d, 0x514a, 0x5f5e, 0x3027, 0x4637,
01264 0x5f53, 0x3a65, 0x365f, 0x4d5b, 0x397e, 0x5455, 0x5f5f, 0x4f6c,
01265 0x3025, 0x5f67, 0x5f51, 0x5146, 0x5f55, 0x5f58, 0x5f59, 0x5f5c,
01266 0x3b29, 0x5f60, 0x5f61, 0x5f62, 0x5f66, 0x5f68, 0x5334, 0x3867,
01267 0x4536, 0x5f6a, 0x495a, 0x4128, 0x4444, 0x3f5e, 0x4f78, 0x555c,
01268 0x5f6e, 0x3238, 0x3a5f, 0x5f6c, 0x5b41, 0x5164, 0x4b74, 0x343d,
01269 0x3026, 0x5f71, 0x4c46, 0x5f72, 0x5f6d, 0x5f69, 0x5f6b, 0x5f6f,
01270 0x5f70, 0x3b3d, 0x5f73, 0x5f74, 0x3b23, 0x4a5b, 0x4e28, 0x6027,
01271 0x332a, 0x6026, 0x6021, 0x5f7e, 0x4d59, 0x5f7c, 0x5f7a, 0x3f50,
01272 0x5744, 0x494c, 0x5f78, 0x3021, 0x5f7d, 0x5f7b, 0x6022, 0x6028,
01273 0x3748, 0x4621, 0x4936, 0x4032, 0x5f75, 0x453e, 0x5844, 0x5f79,
01274 0x4476, 0x6023, 0x6024, 0x6025, 0x5025, 0x6034, 0x4c64, 0x6031,
01275 0x3f26, 0x602f, 0x4e39, 0x602b, 0x4946, 0x402e, 0x602e, 0x3a6d,
01276 0x3a30, 0x6029, 0x5f76, 0x6033, 0x6038, 0x342d, 0x6039, 0x4f32,
01277 0x3a48, 0x6030, 0x6030, 0x507a, 0x602c, 0x547b, 0x5f77, 0x4567, 0x602d,
01278 0x5377, 0x6036, 0x6037, 0x6044, 0x5061, 0x603c, 0x6049, 0x604a,
01279 0x603e, 0x602a, 0x4924, 0x6041, 0x6032, 0x4a48, 0x6043, 0x6035,
01280 0x4e4b, 0x4b43, 0x604d, 0x6046, 0x6042, 0x604b, 0x603a, 0x603f,
01281 0x6040, 0x6045, 0x6047, 0x6048, 0x604c, 0x603b, 0x4b54, 0x6055,
01282 0x6056, 0x6052, 0x6050, 0x3c4e, 0x6051, 0x3842, 0x5845, 0x506a,
01283 0x426f, 0x604f, 0x603d, 0x6054, 0x6053, 0x6057, 0x605c, 0x6058,
01284 0x5676, 0x3330, 0x576c, 0x4b3b, 0x605a, 0x4e7b, 0x3a59, 0x6061,
01285 0x605d, 0x522d, 0x6062, 0x605b, 0x6059, 0x605f, 0x6060, 0x605e,
01286 0x6064, 0x4677, 0x582c, 0x546b, 0x6066, 0x4a49, 0x6065, 0x3841,
01287 0x6067, 0x6068, 0x6069, 0x6063, 0x3a3f, 0x4c67, 0x606a, 0x4f79,
01288 0x606b, 0x4842, 0x3d40, 0x4452, 0x606c, 0x606d, 0x4774, 0x4b44,
01289 0x606e, 0x3b58, 0x5836, 0x5272, 0x606f, 0x4d45, 0x365a, 0x6071,
01290 0x5430, 0x4027, 0x3451, 0x4e27, 0x6070, 0x6072, 0x394c, 0x397a,
01291 0x4d3c, 0x6073, 0x4654, 0x6074, 0x5432, 0x4826, 0x6076, 0x6075,
01292 0x6077, 0x4d41, 0x4a25, 0x545a, 0x5b57, 0x5b59, 0x5b58, 0x3967,
01293 0x5b5c, 0x5b5d, 0x3558, 0x5b5a, 0x5b5b, 0x3321, 0x5b5f, 0x3b78,
01294 0x5637, 0x5b60, 0x3e79, 0x373b, 0x5b50, 0x4c2e, 0x3f32, 0x3b35,
01295 0x5778, 0x3f53, 0x3f69, 0x3c61, 0x4c33, 0x5b5e, 0x3053, 0x4e6b,
01296 0x3758, 0x5739, 0x4642, 0x4024, 0x4c39, 0x5b67, 0x5b61, 0x463a,
01297 0x5b63, 0x5b68, 0x4577, 0x5b6a, 0x5b69, 0x3f40, 0x5b66, 0x5b65,
01298 0x3439, 0x402c, 0x4222, 0x5b62, 0x5b64, 0x504d, 0x5b6d, 0x405d,
01299 0x5b72, 0x3662, 0x5b73, 0x5b52, 0x3938, 0x542b, 0x5b6c, 0x3f51,
01300 0x5b70, 0x5b51, 0x3566, 0x5b6b, 0x3f65, 0x5b6e, 0x5b71, 0x5b79,
01301 0x3921, 0x3023, 0x4271, 0x3347, 0x5b6f, 0x5b78, 0x4652, 0x5b74,
01302 0x5b75, 0x5b77, 0x5b76, 0x5b7e, 0x5372, 0x323a, 0x5b7d, 0x5c24,
01303 0x5b7b, 0x5b7a, 0x5b7c, 0x4560, 0x3b79, 0x5c23, 0x5c25, 0x4c43,
01304 0x3651, 0x5d40, 0x5c21, 0x5c22, 0x4735, 0x3669, 0x5c27, 0x5c26,
01305 0x5c29, 0x3124, 0x354c, 0x3f30, 0x515f, 0x3642, 0x5c28, 0x4b7a,
01306 0x6b73, 0x4b7c, 0x4b7e, 0x4c41, 0x487b, 0x5c2a, 0x4c6e, 0x5c2b,
01307 0x5b53, 0x5c2f, 0x5c2c, 0x3e33, 0x4a7b, 0x5c2d, 0x494a, 0x4439,
01308 0x473d, 0x5c2e, 0x5476, 0x5066, 0x442b, 0x3655, 0x5b54, 0x315a,
01309 0x5b55, 0x5b56, 0x3a3e, 0x4840, 0x4a3f, 0x4849, 0x5733, 0x4979,
01310 0x3f47, 0x3a78, 0x523c, 0x623a, 0x3426, 0x3138, 0x3834, 0x4f44,
01311 0x5967, 0x4f26, 0x4d62, 0x596d, 0x3660, 0x5239, 0x393b, 0x6239,
01312 0x6237, 0x3473, 0x4c6c, 0x4c2b, 0x3772, 0x5832, 0x516b, 0x3a3b,
01313 0x4a27, 0x4d37, 0x5244, 0x3f64, 0x3c50, 0x3661, 0x5e45, 0x5e46,
01314 0x5b3c, 0x5159, 0x4666, 0x444e, 0x376e, 0x375c, 0x3f7c, 0x5760,
01315 0x4675, 0x313c, 0x5e48, 0x3d31, 0x4c57, 0x5e4a, 0x5e49, 0x356c,
01316 0x495d, 0x3042, 0x452e, 0x452b, 0x444c, 0x3c69, 0x4b7d, 0x3a43,
01317 0x6579, 0x4867, 0x657a, 0x4d7d, 0x5731, 0x383e, 0x4268, 0x4851,
01318 0x657b, 0x3644, 0x3c4b, 0x517d, 0x6621, 0x436e, 0x6624, 0x657e,
01319 0x6625, 0x4d57, 0x3741, 0x657c, 0x657d, 0x6623, 0x445d, 0x6628,
01320 0x6627, 0x4343, 0x465e, 0x662a, 0x4437, 0x6622, 0x4a3c, 0x3d63,
01321 0x3943, 0x6626, 0x5055, 0x4e2f, 0x6629, 0x6630, 0x5226, 0x3d2a,
01322 0x662d, 0x662f, 0x4051, 0x524c, 0x3c27, 0x6631, 0x5276, 0x574b,
01323 0x4d7e, 0x4d5e, 0x4226, 0x662b, 0x662c, 0x3d3f, 0x662e, 0x6633,
01324 0x6632, 0x6636, 0x6638, 0x446f, 0x4448, 0x3e6a, 0x496f, 0x6637,
01325 0x3670, 0x4364, 0x5369, 0x6634, 0x6635, 0x4822, 0x663d, 0x6639,
01326 0x4645, 0x4d71, 0x663b, 0x663c, 0x3b69, 0x663e, 0x663a, 0x4037,
01327 0x5324, 0x6633, 0x4974, 0x6643, 0x6644, 0x5076, 0x433d, 0x4344,
01328 0x6642, 0x6641, 0x6647, 0x4f31, 0x6b74, 0x664a, 0x6645, 0x3c5e,
```

01329 0x4929, 0x3c35, 0x4f53, 0x6648, 0x6649, 0x664e, 0x6650, 0x6651,
01330 0x664b, 0x3555, 0x664c, 0x664f, 0x445b, 0x6646, 0x664d, 0x6652,
01331 0x6654, 0x6653, 0x6655, 0x5978, 0x6656, 0x6657, 0x5753, 0x665d,
01332 0x665e, 0x3f57, 0x5450, 0x5756, 0x3466, 0x4b6f, 0x665a, 0x5843,
01333 0x574e, 0x5022, 0x434f, 0x665f, 0x3c3e, 0x3942, 0x665b, 0x5127,
01334 0x3a22, 0x424f, 0x582b, 0x4a6b, 0x656e, 0x665c, 0x3775, 0x4866,
01335 0x4475, 0x6532, 0x447e, 0x4b7c, 0x6533, 0x552c, 0x536e, 0x4a58,
01336 0x3032, 0x4b4e, 0x4d6a, 0x3a6a, 0x6535, 0x6534, 0x575a, 0x3959,
01337 0x5666, 0x3628, 0x4d70, 0x524b, 0x3126, 0x4a35, 0x3368, 0x4973,
01338 0x3f4d, 0x507b, 0x4a52, 0x6536, 0x3b42, 0x4f5c, 0x392c, 0x5457,
01339 0x3a26, 0x5167, 0x4f7c, 0x3c52, 0x6537, 0x485d, 0x3f6d, 0x3176,
01340 0x4b5e, 0x3c45, 0x3c44, 0x527a, 0x435c, 0x3f5c, 0x383b, 0x4342,
01341 0x3a2e, 0x5422, 0x475e, 0x442f, 0x326c, 0x3951, 0x653b, 0x4148,
01342 0x552f, 0x653c, 0x653e, 0x3467, 0x3654, 0x4b42, 0x5130, 0x353c,
01343 0x4a59, 0x3762, 0x4964, 0x3d2b, 0x4e3e, 0x5770, 0x5021, 0x4959,
01344 0x367b, 0x6658, 0x3c62, 0x333e, 0x4950, 0x6659, 0x3322, 0x5e4c,
01345 0x5348, 0x5e4d, 0x5222, 0x5e4e, 0x3e4d, 0x5e4f, 0x4a2c, 0x527c,
01346 0x335f, 0x656a, 0x4461, 0x3e21, 0x4e32, 0x4472, 0x3e56, 0x4628,
01347 0x3263, 0x3e53, 0x477c, 0x4c6b, 0x3d6c, 0x4e5d, 0x4a3a, 0x4641,
01348 0x656c, 0x503c, 0x5539, 0x656d, 0x4a74, 0x4d40, 0x4245, 0x656f,
01349 0x4244, 0x6570, 0x6578, 0x4d4d, 0x493d, 0x5259, 0x6128, 0x536c,
01350 0x4b6a, 0x4671, 0x612c, 0x6127, 0x6129, 0x612a, 0x612f, 0x326d,
01351 0x612b, 0x385a, 0x612d, 0x612e, 0x6130, 0x353a, 0x6131, 0x6133,
01352 0x6138, 0x5152, 0x6136, 0x6135, 0x416b, 0x6137, 0x5440, 0x6132,
01353 0x613a, 0x3036, 0x6134, 0x3f79, 0x6139, 0x613b, 0x613e, 0x613c,
01354 0x5645, 0x4f3f, 0x613d, 0x613f, 0x424d, 0x366b, 0x5378, 0x474d,
01355 0x3765, 0x3e7e, 0x6140, 0x6141, 0x6147, 0x3367, 0x4669, 0x345e,
01356 0x5142, 0x6148, 0x6146, 0x6145, 0x6143, 0x6142, 0x3140, 0x5538,
01357 0x6144, 0x614b, 0x614c, 0x614a, 0x6f7a, 0x6153, 0x6152, 0x4736,
01358 0x6149, 0x614e, 0x6150, 0x6154, 0x6151, 0x614d, 0x614f, 0x6155,
01359 0x6156, 0x6157, 0x6158, 0x615a, 0x615b, 0x4e21, 0x675d, 0x3428,
01360 0x565d, 0x5132, 0x3332, 0x3924, 0x5773, 0x4749, 0x3e5e, 0x392e,
01361 0x4e57, 0x326e, 0x5b4f, 0x3c3a, 0x5251, 0x4b48, 0x304d, 0x4f6f,
01362 0x5963, 0x3d6d, 0x3152, 0x4a50, 0x323c, 0x4b27, 0x372b, 0x4a26,
01363 0x4f23, 0x6078, 0x554a, 0x607b, 0x607a, 0x4541, 0x4c7b, 0x4131,
01364 0x6079, 0x5663, 0x322f, 0x5644, 0x355b, 0x3478, 0x5621, 0x4f2f,
01365 0x306f, 0x607c, 0x6121, 0x3323, 0x607d, 0x607e, 0x4331, 0x435d,
01366 0x6122, 0x3779, 0x3b4f, 0x6123, 0x443b, 0x6124, 0x6125, 0x6126,
01367 0x3431, 0x3849, 0x463d, 0x446a, 0x3222, 0x5052, 0x675b, 0x3b43,
01368 0x5357, 0x5344, 0x3963, 0x624f, 0x572f, 0x476c, 0x3153, 0x3432,
01369 0x6251, 0x5072, 0x422e, 0x6250, 0x3f62, 0x5326, 0x3557, 0x6252,
01370 0x356a, 0x436d, 0x387d, 0x382e, 0x4553, 0x374f, 0x6254, 0x6253,
01371 0x3648, 0x5779, 0x4d25, 0x6258, 0x6256, 0x4a7c, 0x3f35, 0x5339,
01372 0x6255, 0x6257, 0x412e, 0x4048, 0x625b, 0x625a, 0x402a, 0x414e,
01373 0x625c, 0x625d, 0x625e, 0x5b48, 0x5153, 0x4d22, 0x3d28, 0x5e43,
01374 0x5825, 0x3f2a, 0x5b4d, 0x526c, 0x467a, 0x452a, 0x5e44, 0x3157,
01375 0x5f2e, 0x4a3d, 0x5f31, 0x392d, 0x527d, 0x3825, 0x3a6b, 0x335a,
01376 0x355c, 0x5545, 0x4356, 0x4f52, 0x3b21, 0x6573, 0x6572, 0x6574,
01377 0x4d64, 0x4875, 0x352f, 0x473f, 0x6576, 0x6c30, 0x6566, 0x3969,
01378 0x3531, 0x423c, 0x6568, 0x6567, 0x6569, 0x524d, 0x616a, 0x504e,
01379 0x4d2e, 0x5165, 0x324a, 0x316b, 0x3172, 0x456d, 0x5543, 0x5330,
01380 0x615c, 0x615d, 0x525b, 0x3339, 0x314b, 0x4d79, 0x5577, 0x615e,
01381 0x3e36, 0x347d, 0x615f, 0x3a5c, 0x6160, 0x3b32, 0x4249, 0x6161,
01382 0x506c, 0x4d3d, 0x6162, 0x3543, 0x4547, 0x6163, 0x6164, 0x5379,
01383 0x6165, 0x512d, 0x6166, 0x4e22, 0x6167, 0x3542, 0x6168, 0x3b55,
01384 0x5044, 0x6260, 0x3158, 0x5264, 0x6261, 0x3c49, 0x484c, 0x6263,
01385 0x6c7e, 0x6c7d, 0x5f2f, 0x6262, 0x563e, 0x4d7c, 0x4326, 0x6343,
01386 0x5652, 0x6267, 0x6268, 0x5347, 0x626c, 0x3f6c, 0x626d, 0x6265,
01387 0x3340, 0x446e, 0x626e, 0x5043, 0x3a76, 0x6269, 0x375e, 0x3b33,
01388 0x4c2c, 0x4b4b, 0x6264, 0x6266, 0x626a, 0x626b, 0x6277, 0x6274,
01389 0x5475, 0x6273, 0x452d, 0x557a, 0x4542, 0x3240, 0x626f, 0x6272,
01390 0x412f, 0x4b3c, 0x3521, 0x6279, 0x3c31, 0x6271, 0x5054, 0x5439,
01391 0x6275, 0x3956, 0x6276, 0x4753, 0x6270, 0x575c, 0x6d21, 0x6278,
01392 0x6d25, 0x627e, 0x4a51, 0x4135, 0x3b50, 0x3f56, 0x3a63, 0x4b21,
01393 0x6d26, 0x6d23, 0x6d22, 0x3b56, 0x6d27, 0x5074, 0x6d24, 0x3a5e,
01394 0x3677, 0x6321, 0x6322, 0x4c71, 0x3927, 0x4f22, 0x4721, 0x3f52,
01395 0x3671, 0x627a, 0x627b, 0x627d, 0x627c, 0x4455, 0x6322, 0x5341,
01396 0x6327, 0x4744, 0x4f24, 0x6329, 0x3a37, 0x6328, 0x3b5a, 0x6323,
01397 0x6324, 0x632a, 0x6326, 0x4e72, 0x5346, 0x3b3c, 0x5443, 0x447a,
01398 0x6d28, 0x507c, 0x6325, 0x4375, 0x632d, 0x312f, 0x6332, 0x3c42,
01399 0x632c, 0x353f, 0x4769, 0x6330, 0x3e2a, 0x4d6f, 0x3b73, 0x4c68,
01400 0x632f, 0x6331, 0x4f27, 0x632e, 0x4e29, 0x3b5d, 0x356b, 0x3e65,
01401 0x3252, 0x334d, 0x3139, 0x632b, 0x3251, 0x352c, 0x395f, 0x3668,
01402 0x4f6b, 0x6337, 0x3b4c, 0x4847, 0x504a, 0x6338, 0x336e, 0x6d29,
01403 0x537a, 0x5364, 0x6d2a, 0x6339, 0x5262, 0x6335, 0x535e, 0x3850,
01404 0x6333, 0x6336, 0x375f, 0x6334, 0x4022, 0x633a, 0x5438, 0x3448,
01405 0x633b, 0x3b45, 0x4977, 0x4965, 0x443d, 0x6d2b, 0x427d, 0x3b5b,
01406 0x3f2e, 0x4e3f, 0x633c, 0x3f36, 0x316f, 0x5477, 0x633e, 0x6d2d,
01407 0x633f, 0x3a29, 0x6d2c, 0x633d, 0x6340, 0x3a36, 0x362e, 0x5038,
01408 0x3043, 0x6d2e, 0x6d2f, 0x4041, 0x6341, 0x4533, 0x6342, 0x5c32,
01409 0x6d30, 0x386a, 0x4e6c, 0x6a27, 0x5067, 0x4a79, 0x4856, 0x4f37,
01410 0x3349, 0x4e52, 0x3d64, 0x635e, 0x3b72, 0x6a28, 0x553d, 0x465d,
01411 0x6a29, 0x6a2a, 0x6a2c, 0x6a2b, 0x6a2e, 0x6a2d, 0x3d58, 0x6a2f,
01412 0x423e, 0x3441, 0x3477, 0x3b27, 0x6c66, 0x6c65, 0x373f, 0x4b79,
01413 0x3162, 0x6c67, 0x4948, 0x6c68, 0x6c69, 0x4a56, 0x5e50, 0x3245,
01414 0x547a, 0x464b, 0x3047, 0x3472, 0x4853, 0x4d50, 0x3f38, 0x3f5b,
01415 0x4724, 0x5634, 0x4029, 0x5e51, 0x4928, 0x516f, 0x4524, 0x3067,

```
01416 0x3336, 0x4845, 0x3062, 0x3776, 0x457a, 0x3673, 0x5552, 0x3350,
01417 0x3c3c, 0x332d, 0x3e71, 0x3051, 0x5256, 0x4a63, 0x5725, 0x4d36,
01418 0x3636, 0x3f39, 0x555b, 0x3827, 0x4557, 0x5e52, 0x3f59, 0x4255,
01419 0x4740, 0x3b24, 0x3128, 0x456a, 0x457b, 0x4c27, 0x3127, 0x3556,
01420 0x4428, 0x5e53, 0x513a, 0x3369, 0x4372, 0x3777, 0x5674, 0x3523,
01421 0x3270, 0x4434, 0x4469, 0x402d, 0x5e54, 0x3068, 0x4544, 0x4160,
01422 0x3955, 0x3e5c, 0x4d58, 0x304e, 0x4d4f, 0x5e56, 0x3e50, 0x573e,
01423 0x5e55, 0x5550, 0x305d, 0x4462, 0x4223, 0x3c70, 0x5335, 0x4039,
01424 0x4521, 0x3226, 0x5471, 0x4028, 0x4a43, 0x5e57, 0x557c, 0x3930,
01425 0x482d, 0x4b29, 0x5e59, 0x3f3d, 0x4634, 0x5727, 0x4a30, 0x4443,
01426 0x3356, 0x3952, 0x5638, 0x6a7c, 0x3034, 0x3f66, 0x4c74, 0x4d5a,
01427 0x563f, 0x424e, 0x4e4e, 0x4c22, 0x502e, 0x4453, 0x3532, 0x5e58,
01428 0x5575, 0x3c37, 0x3b53, 0x3024, 0x4532, 0x346c, 0x5571, 0x6a7d,
01429 0x5e5a, 0x4d26, 0x4d6c, 0x4e66, 0x5e5c, 0x4d31, 0x4026, 0x573d,
01430 0x5e5b, 0x3046, 0x3a34, 0x4953, 0x4473, 0x3e68, 0x3236, 0x404c,
01431 0x4b70, 0x3c71, 0x3b3b, 0x3537, 0x4575, 0x5e66, 0x5e63, 0x3e5d,
01432 0x5e5f, 0x3437, 0x3d5d, 0x5e60, 0x446d, 0x4f46, 0x3560, 0x365e,
01433 0x4a5a, 0x3574, 0x5e65, 0x5546, 0x5e61, 0x4c4d, 0x467e, 0x4545,
01434 0x5234, 0x3e72, 0x4253, 0x4c3d, 0x3338, 0x3d53, 0x3f58, 0x4d46,
01435 0x515a, 0x346b, 0x5e64, 0x5e5d, 0x5e67, 0x6a7e, 0x4230, 0x5e62,
01436 0x5640, 0x3527, 0x3274, 0x5e68, 0x5e72, 0x5e6d, 0x5e71, 0x4860,
01437 0x5761, 0x5e6f, 0x4368, 0x4c61, 0x3265, 0x523e, 0x5e6e, 0x5e6b,
01438 0x4e55, 0x3427, 0x3f2b, 0x3e3e, 0x3d52, 0x5e69, 0x542e, 0x5e5e,
01439 0x5e6a, 0x403f, 0x5e6c, 0x3273, 0x3869, 0x4227, 0x3d41, 0x5e75,
01440 0x5e78, 0x322b, 0x3424, 0x346a, 0x4926, 0x5e76, 0x4b51, 0x3863,
01441 0x5e77, 0x5e7a, 0x5e79, 0x4c42, 0x3061, 0x346e, 0x653a, 0x502f,
01442 0x326b, 0x6b21, 0x5e74, 0x4963, 0x5e73, 0x305a, 0x5221, 0x3177,
01443 0x4c2f, 0x5e70, 0x4b24, 0x552a, 0x5e7b, 0x345d, 0x4426, 0x5e7d,
01444 0x437e, 0x4421, 0x5f21, 0x414c, 0x5e7c, 0x3e6f, 0x4632, 0x3345,
01445 0x4876, 0x4b3a, 0x5e7e, 0x5f24, 0x5732, 0x3337, 0x4143, 0x474b,
01446 0x3225, 0x3469, 0x572b, 0x446c, 0x5f22, 0x5f23, 0x5f25, 0x3a33,
01447 0x5f26, 0x405e, 0x4943, 0x3259, 0x4766, 0x5f27, 0x475c, 0x5f28,
01448 0x6b22, 0x4b53, 0x5f2a, 0x5f29, 0x3241, 0x454a, 0x5f2b, 0x545c,
01449 0x4841, 0x5f2c, 0x3e70, 0x5f2d, 0x5627, 0x6a37, 0x6b36, 0x4a55,
01450 0x587c, 0x3844, 0x3925, 0x3745, 0x557e, 0x394a, 0x5027, 0x744d,
01451 0x3550, 0x4374, 0x3e48, 0x6b37, 0x303d, 0x3d4c, 0x4132, 0x3156,
01452 0x3328, 0x3852, 0x4922, 0x3658, 0x6b38, 0x3e34, 0x4a7d, 0x4743,
01453 0x557b, 0x3773, 0x4e44, 0x552b, 0x3173, 0x6c33, 0x305f, 0x6c35,
01454 0x3637, 0x414f, 0x757a, 0x5031, 0x5565, 0x4e53, 0x3d6f, 0x3362,
01455 0x382b, 0x5536, 0x6d3d, 0x364f, 0x4b39, 0x5042, 0x373d, 0x6c36,
01456 0x4a29, 0x4554, 0x6c39, 0x6c38, 0x4243, 0x6c37, 0x507d, 0x6c3a,
01457 0x6c3b, 0x5765, 0x6c3c, 0x6c3d, 0x466c, 0x4e5e, 0x3c48, 0x4855,
01458 0x3529, 0x3e49, 0x563c, 0x5467, 0x512e, 0x5071, 0x6a38, 0x6a39,
01459 0x6a3a, 0x3a35, 0x4a31, 0x3f75, 0x4d7a, 0x6a40, 0x303a, 0x6a3e,
01460 0x4025, 0x6a3b, 0x327d, 0x4377, 0x3b68, 0x5257, 0x4e74, 0x6a3f,
01461 0x6a3c, 0x6a43, 0x5047, 0x5333, 0x343a, 0x4341, 0x5772, 0x5551,
01462 0x4a47, 0x6a45, 0x6a44, 0x6a47, 0x6a46, 0x5667, 0x4f54, 0x6a4b,
01463 0x3b4e, 0x3d7a, 0x494e, 0x6a4c, 0x4939, 0x4f7e, 0x6a4a, 0x544e,
01464 0x6a4d, 0x6a4f, 0x4d6d, 0x6a49, 0x6a4e, 0x4e6e, 0x3b5e, 0x333f,
01465 0x4655, 0x3e30, 0x4e7a, 0x4767, 0x3e27, 0x6a50, 0x5647, 0x4140,
01466 0x545d, 0x6a51, 0x4f3e, 0x6a52, 0x4a6e, 0x452f, 0x3035, 0x6a54,
01467 0x6a53, 0x745f, 0x443a, 0x3129, 0x655f, 0x6a55, 0x4a6f, 0x6a56,
01468 0x6a57, 0x4658, 0x6a58, 0x6a59, 0x543b, 0x477a, 0x5237, 0x387c,
01469 0x6a42, 0x325c, 0x427c, 0x5478, 0x4c66, 0x576e, 0x5442, 0x5350,
01470 0x6b43, 0x4573, 0x377e, 0x6b54, 0x4b37, 0x6b5e, 0x404a, 0x4d7b,
01471 0x332f, 0x465a, 0x6b7c, 0x443e, 0x4e34, 0x4429, 0x313e, 0x547d,
01472 0x4a75, 0x566c, 0x4653, 0x3664, 0x3b7a, 0x5060, 0x4931, 0x5453,
01473 0x4828, 0x384b, 0x683e, 0x493c, 0x683b, 0x406e, 0x5053, 0x3244,
01474 0x3465, 0x683c, 0x5548, 0x3645, 0x683d, 0x4a78, 0x385c, 0x4c75,
01475 0x4034, 0x516e, 0x683f, 0x6842, 0x3a3c, 0x312d, 0x3d5c, 0x6a3d,
01476 0x6843, 0x6846, 0x684b, 0x684c, 0x4b49, 0x3065, 0x3c2b, 0x3939,
01477 0x6841, 0x4d77, 0x684a, 0x4e76, 0x556d, 0x4156, 0x6844, 0x4336,
01478 0x397b, 0x5626, 0x6848, 0x4a60, 0x5466, 0x6840, 0x6845, 0x6847,
01479 0x4739, 0x3763, 0x6849, 0x3f5d, 0x6852, 0x6857, 0x6855, 0x3c5c,
01480 0x3c4f, 0x685b, 0x685e, 0x685a, 0x317a, 0x3058, 0x4433, 0x384c,
01481 0x4662, 0x483c, 0x4861, 0x684f, 0x6854, 0x6856, 0x3971, 0x6858,
01482 0x5775, 0x447b, 0x685c, 0x3269, 0x6851, 0x3c6d, 0x3f42, 0x684d,
01483 0x5679, 0x4178, 0x3271, 0x685f, 0x4a41, 0x6859, 0x5524, 0x316a,
01484 0x553b, 0x684e, 0x6850, 0x3630, 0x6853, 0x685d, 0x4038, 0x4a77,
01485 0x4b28, 0x465c, 0x4075, 0x6869, 0x5023, 0x6872, 0x566a, 0x6860,
01486 0x6861, 0x5179, 0x3a4b, 0x3879, 0x3871, 0x5454, 0x686f, 0x686e,
01487 0x686c, 0x3970, 0x4c52, 0x6866, 0x4e26, 0x3f72, 0x3038, 0x6871,
01488 0x6870, 0x5740, 0x6864, 0x4d29, 0x4923, 0x3b38, 0x3d5b, 0x686a,
01489 0x6862, 0x6863, 0x6865, 0x3535, 0x6867, 0x4745, 0x686b, 0x686d,
01490 0x3d30, 0x572e, 0x6878, 0x6875, 0x4d30, 0x6876, 0x413a, 0x6868,
01491 0x4337, 0x3070, 0x6874, 0x6877, 0x3923, 0x4952, 0x434e, 0x4e60,
01492 0x4066, 0x4b73, 0x4c5d, 0x5035, 0x4a61, 0x6873, 0x3c6c, 0x6879,
01493 0x435e, 0x4665, 0x3977, 0x3074, 0x5758, 0x3c2c, 0x456f, 0x4c44,
01494 0x6926, 0x492d, 0x6922, 0x4062, 0x3f43, 0x687e, 0x3957, 0x687b,
01495 0x6924, 0x524e, 0x6923, 0x5632, 0x5735, 0x6927, 0x3d37, 0x687c,
01496 0x687d, 0x6921, 0x4d56, 0x522c, 0x6932, 0x6929, 0x342a, 0x343b,
01497 0x692b, 0x5028, 0x6925, 0x337e, 0x692c, 0x4063, 0x692a, 0x6939,
01498 0x6938, 0x692e, 0x687a, 0x6928, 0x3f2c, 0x6931, 0x693a, 0x4225,
01499 0x692f, 0x3845, 0x692d, 0x535c, 0x6934, 0x6935, 0x6937, 0x6947,
01500 0x4046, 0x6945, 0x6930, 0x693b, 0x3071, 0x693c, 0x5525, 0x693e,
01501 0x693f, 0x6941, 0x4171, 0x4836, 0x693d, 0x6942, 0x6943, 0x6933,
01502 0x6936, 0x3b31, 0x6940, 0x3c77, 0x6944, 0x6946, 0x694a, 0x694e,
```


01503 0x325b, 0x6948, 0x372e, 0x694b, 0x694c, 0x5541, 0x4423, 0x6958,
01504 0x3a61, 0x6949, 0x5323, 0x6954, 0x6957, 0x6950, 0x694f, 0x4741,
01505 0x6952, 0x6959, 0x3348, 0x6953, 0x4f70, 0x694d, 0x3377, 0x6956,
01506 0x695a, 0x4c34, 0x4f2d, 0x6955, 0x695c, 0x695b, 0x695e, 0x6951,
01507 0x695d, 0x695f, 0x434a, 0x4737, 0x344e, 0x3b36, 0x5040, 0x6c23,
01508 0x4537, 0x537b, 0x6c24, 0x6c25, 0x465b, 0x3f6e, 0x6c26, 0x6c27,
01509 0x502a, 0x4738, 0x3868, 0x6c28, 0x5639, 0x557d, 0x344b, 0x323d,
01510 0x4e64, 0x4667, 0x4d61, 0x3475, 0x4b40, 0x3c5f, 0x6962, 0x6963,
01511 0x516a, 0x6965, 0x3479, 0x6964, 0x5133, 0x4a62, 0x3250, 0x6968,
01512 0x6966, 0x6967, 0x5633, 0x6969, 0x696a, 0x696b, 0x696c, 0x6c2f,
01513 0x4539, 0x364e, 0x5273, 0x356e, 0x3b59, 0x6c31, 0x5263, 0x4e63,
01514 0x4438, 0x433f, 0x363e, 0x5839, 0x3148, 0x314f, 0x3151, 0x457e,
01515 0x3150, 0x432b, 0x5531, 0x6b24, 0x3a41, 0x4c3a, 0x6b25, 0x6b27,
01516 0x6b28, 0x6b26, 0x6b29, 0x6b2b, 0x6b2a, 0x6b2c, 0x4a4f, 0x5835,
01517 0x4371, 0x4325, 0x4678, 0x6b2d, 0x444a, 0x6b2e, 0x6b2f, 0x6b30,
01518 0x3755, 0x377a, 0x6b31, 0x4762, 0x6b33, 0x3a24, 0x5175, 0x3031,
01519 0x6b32, 0x6b34, 0x352a, 0x4248, 0x4768, 0x6b35, 0x4b2e, 0x635f,
01520 0x5340, 0x595b, 0x4d21, 0x562d, 0x4773, 0x5960, 0x3b63, 0x3a3a,
01521 0x6362, 0x4f2b, 0x6360, 0x4947, 0x3a39, 0x5134, 0x6361, 0x486a,
01522 0x392f, 0x3d2d, 0x3358, 0x4e5b, 0x4c40, 0x6368, 0x6369, 0x4d74,
01523 0x4c2d, 0x3c33, 0x636a, 0x636b, 0x505a, 0x467b, 0x375a, 0x475f,
01524 0x524a, 0x4e56, 0x6364, 0x636c, 0x4972, 0x3341, 0x6367, 0x4663,
01525 0x6365, 0x6d33, 0x6366, 0x4933, 0x4566, 0x3935, 0x433b, 0x6363,
01526 0x453d, 0x4124, 0x4259, 0x3257, 0x636d, 0x3b26, 0x442d, 0x6370,
01527 0x3e5a, 0x637b, 0x6375, 0x3a53, 0x3750, 0x534d, 0x564e, 0x5553,
01528 0x3941, 0x5534, 0x5158, 0x5039, 0x4776, 0x482a, 0x3234, 0x435a,
01529 0x636e, 0x637c, 0x636f, 0x3728, 0x6377, 0x6374, 0x373a, 0x4522,
01530 0x6376, 0x455d, 0x3228, 0x467c, 0x4460, 0x5722, 0x4061, 0x6379,
01531 0x637a, 0x637d, 0x4c29, 0x6373, 0x533e, 0x3143, 0x6d34, 0x6371,
01532 0x6372, 0x6378, 0x503a, 0x4643, 0x5473, 0x637e, 0x3d60, 0x6427,
01533 0x6426, 0x5173, 0x6423, 0x6429, 0x4877, 0x4f34, 0x6428, 0x642e,
01534 0x4265, 0x3634, 0x3d72, 0x6422, 0x3a69, 0x642a, 0x642c, 0x367d,
01535 0x565e, 0x6432, 0x642d, 0x6421, 0x3b6e, 0x4d5d, 0x4722, 0x4549,
01536 0x4177, 0x6424, 0x4733, 0x3d2c, 0x3d3d, 0x6425, 0x5747, 0x3262,
01537 0x642b, 0x3c43, 0x642f, 0x3b6b, 0x6430, 0x4528, 0x6431, 0x5563,
01538 0x3f23, 0x643a, 0x6437, 0x643b, 0x643d, 0x4656, 0x3a46, 0x404b,
01539 0x3821, 0x6434, 0x5421, 0x3a23, 0x3d7e, 0x643c, 0x4d3f, 0x4479,
01540 0x4f7b, 0x4966, 0x533f, 0x4f51, 0x6433, 0x6438, 0x6439, 0x4c69,
01541 0x4c4e, 0x4054, 0x6435, 0x4130, 0x6436, 0x4e50, 0x3b41, 0x3553,
01542 0x4873, 0x3d27, 0x5547, 0x492c, 0x3822, 0x644a, 0x644c, 0x5144,
01543 0x523a, 0x3a2d, 0x3a54, 0x6443, 0x356d, 0x574d, 0x6440, 0x4f7d,
01544 0x643f, 0x415c, 0x4c4a, 0x4a67, 0x4457, 0x4c54, 0x6448, 0x6447,
01545 0x6441, 0x6444, 0x352d, 0x5359, 0x6446, 0x5279, 0x3463, 0x3b34,
01546 0x496e, 0x343e, 0x3b6c, 0x514d, 0x4c6d, 0x6d35, 0x4765, 0x5428,
01547 0x644b, 0x5755, 0x6442, 0x3d25, 0x6445, 0x5366, 0x6449, 0x4978,
01548 0x643e, 0x5365, 0x477e, 0x3649, 0x547c, 0x3233, 0x6457, 0x4e42,
01549 0x644d, 0x4e3c, 0x385b, 0x6456, 0x3f4a, 0x534e, 0x436c, 0x4548,
01550 0x6458, 0x4d44, 0x644f, 0x6454, 0x6455, 0x3a7e, 0x4f66, 0x553f,
01551 0x6452, 0x6450, 0x644e, 0x4d65, 0x4a2a, 0x4023, 0x3d26, 0x6453,
01552 0x3848, 0x6467, 0x5434, 0x645b, 0x416f, 0x6469, 0x5267, 0x645f,
01553 0x6460, 0x4f2a, 0x4b5d, 0x645a, 0x6451, 0x6465, 0x485c, 0x6463,
01554 0x4467, 0x6462, 0x6461, 0x337c, 0x6468, 0x3561, 0x574c, 0x6466,
01555 0x3b2c, 0x5752, 0x4c4f, 0x6b78, 0x6464, 0x3976, 0x564d, 0x6459,
01556 0x645c, 0x427a, 0x645e, 0x424b, 0x4044, 0x4250, 0x3175, 0x4c32,
01557 0x354e, 0x646f, 0x462f, 0x4661, 0x6475, 0x4229, 0x406c, 0x515d,
01558 0x646e, 0x442e, 0x646d, 0x6476, 0x6474, 0x427e, 0x645d, 0x6470,
01559 0x4a7e, 0x5544, 0x6471, 0x517a, 0x646b, 0x646c, 0x6472, 0x4e2b,
01560 0x454b, 0x4731, 0x423a, 0x646a, 0x414a, 0x4c36, 0x3331, 0x647b,
01561 0x6473, 0x647a, 0x647d, 0x647c, 0x334e, 0x333a, 0x6477, 0x6479,
01562 0x6478, 0x456c, 0x403d, 0x5468, 0x6522, 0x3044, 0x6524, 0x6523,
01563 0x3c24, 0x6525, 0x6521, 0x647e, 0x3174, 0x6528, 0x6529, 0x6526,
01564 0x6527, 0x652a, 0x4659, 0x652b, 0x652d, 0x652c, 0x652f, 0x652e,
01565 0x3960, 0x6530, 0x6531, 0x3b70, 0x6c61, 0x4370, 0x3546, 0x3b52,
01566 0x4169, 0x546e, 0x3e44, 0x5746, 0x5456, 0x3253, 0x6c3e, 0x6a41,
01567 0x422f, 0x3436, 0x5157, 0x3334, 0x4832, 0x3f3b, 0x6c40, 0x564b,
01568 0x6c3f, 0x6c41, 0x6c45, 0x3e66, 0x4c3f, 0x455a, 0x3e3c, 0x6c46,
01569 0x317e, 0x6c44, 0x5528, 0x3563, 0x6c42, 0x4136, 0x3363, 0x6c43,
01570 0x4b38, 0x4043, 0x4c7e, 0x4152, 0x6c48, 0x3a66, 0x4053, 0x5672,
01571 0x514c, 0x3f33, 0x3733, 0x4955, 0x6c47, 0x3b62, 0x4c4c, 0x3d7d,
01572 0x4848, 0x4f29, 0x4d69, 0x456b, 0x3769, 0x5149, 0x3a38, 0x6c49,
01573 0x6c4a, 0x3b40, 0x6c4b, 0x6c62, 0x313a, 0x3759, 0x3d39, 0x6c4c,
01574 0x5166, 0x6c4d, 0x483b, 0x6c51, 0x6c53, 0x3b4d, 0x3c65, 0x6c4f,
01575 0x4937, 0x433a, 0x6c63, 0x5555, 0x6c50, 0x5673, 0x6c52, 0x6c4e,
01576 0x6c54, 0x6c55, 0x493f, 0x4f28, 0x505c, 0x512c, 0x485b, 0x6c56,
01577 0x4e75, 0x4a6c, 0x6c5a, 0x6c59, 0x303e, 0x6c57, 0x6c58, 0x6c64,
01578 0x483c, 0x4147, 0x6c5c, 0x5160, 0x6c5b, 0x546f, 0x6c5d, 0x5b46,
01579 0x6c5e, 0x312c, 0x6c5f, 0x6c60, 0x5726, 0x4540, 0x6b3c, 0x302e,
01580 0x3e74, 0x3838, 0x522f, 0x3056, 0x3579, 0x5833, 0x4b2c, 0x635d,
01581 0x462c, 0x3066, 0x4546, 0x6b39, 0x6b3a, 0x6b3b, 0x5140, 0x4523,
01582 0x6a72, 0x4432, 0x4435, 0x404e, 0x6a73, 0x4441, 0x4e6f, 0x6a70,
01583 0x6a74, 0x497c, 0x4723, 0x4c58, 0x4e7e, 0x6a75, 0x6a76, 0x4f2c,
01584 0x4067, 0x6a77, 0x363f, 0x6a78, 0x6a79, 0x6a7a, 0x6a7b, 0x6a71,
01585 0x482e, 0x616b, 0x3738, 0x616c, 0x616d, 0x5734, 0x616e, 0x616f,
01586 0x534c, 0x6171, 0x3f71, 0x6170, 0x3552, 0x3137, 0x6173, 0x6172,
01587 0x3a7c, 0x6174, 0x3937, 0x3e51, 0x447c, 0x3a5d, 0x3d46, 0x6175,
01588 0x6177, 0x3640, 0x4f41, 0x4a28, 0x6176, 0x5578, 0x537c, 0x6178,
01589 0x617c, 0x6179, 0x617a, 0x406a, 0x617e, 0x6221, 0x4047, 0x617b,

```
01590 0x617d, 0x6225, 0x4154, 0x6223, 0x6228, 0x327e, 0x6222, 0x434d,
01591 0x3242, 0x6227, 0x6226, 0x6224, 0x6229, 0x622b, 0x5049, 0x566d,
01592 0x4328, 0x622c, 0x4f57, 0x622e, 0x3a6f, 0x6960, 0x622d, 0x622a,
01593 0x3b2b, 0x5433, 0x6230, 0x622f, 0x6961, 0x6231, 0x6232, 0x6233,
01594 0x4c21, 0x6234, 0x6235, 0x507e, 0x424a, 0x5371, 0x4d75, 0x6760,
01595 0x6761, 0x3e41, 0x426a, 0x6764, 0x6763, 0x4d66, 0x4335, 0x6762,
01596 0x3b37, 0x4f56, 0x4161, 0x6769, 0x6768, 0x6774, 0x3223, 0x676a,
01597 0x6766, 0x676c, 0x676b, 0x493a, 0x5564, 0x6765, 0x3729, 0x6767,
01598 0x676e, 0x6773, 0x5669, 0x676d, 0x6772, 0x6771, 0x3060, 0x6775,
01599 0x4772, 0x4045, 0x406d, 0x4170, 0x6770, 0x6776, 0x4b76, 0x6822,
01600 0x6821, 0x5741, 0x677a, 0x6779, 0x677b, 0x6777, 0x677e, 0x677d,
01601 0x677c, 0x4155, 0x4759, 0x457d, 0x4543, 0x476d, 0x6823, 0x6826,
01602 0x6825, 0x6827, 0x3a77, 0x6778, 0x6824, 0x4870, 0x492a, 0x6829,
01603 0x3965, 0x517e, 0x6828, 0x682a, 0x682d, 0x682e, 0x4127, 0x682f,
01604 0x6830, 0x682c, 0x6834, 0x682b, 0x6831, 0x6835, 0x6832, 0x6833,
01605 0x6837, 0x6836, 0x394f, 0x702c, 0x702d, 0x4630, 0x306a, 0x483f,
01606 0x4d5f, 0x4e4d, 0x6a31, 0x6a32, 0x463f, 0x3449, 0x6a33, 0x5567,
01607 0x5d79, 0x6a34, 0x6a35, 0x6a36, 0x384a, 0x5f30, 0x4975, 0x4c70,
01608 0x497a, 0x497b, 0x5343, 0x4b26, 0x3826, 0x702e, 0x3142, 0x6538,
01609 0x4c6f, 0x5349, 0x3c57, 0x496a, 0x3567, 0x4450, 0x3569, 0x6e2e,
01610 0x3b2d, 0x675e, 0x6e2f, 0x3329, 0x6e32, 0x6e31, 0x3d67, 0x6e30,
01611 0x4e37, 0x454f, 0x4174, 0x5b4e, 0x6e33, 0x5073, 0x4254, 0x4668,
01612 0x372c, 0x6e34, 0x336b, 0x3b7b, 0x6e35, 0x675c, 0x6e36, 0x3d2e,
01613 0x7162, 0x4a68, 0x5249, 0x705a, 0x705b, 0x705c, 0x4146, 0x386d,
01614 0x3e4e, 0x705e, 0x4531, 0x705d, 0x5171, 0x7060, 0x304c, 0x3d6a,
01615 0x525f, 0x705f, 0x342f, 0x3768, 0x7066, 0x7065, 0x4623, 0x7061,
01616 0x7062, 0x3443, 0x7063, 0x556e, 0x4c5b, 0x3e52, 0x3c32, 0x7068,
01617 0x7067, 0x7064, 0x3221, 0x5622, 0x5338, 0x3e37, 0x482c, 0x706a,
01618 0x5177, 0x564c, 0x3a5b, 0x7069, 0x363b, 0x4d34, 0x4626, 0x4121,
01619 0x706b, 0x706e, 0x706d, 0x7070, 0x706c, 0x3b3e, 0x706f, 0x4c35,
01620 0x7072, 0x3355, 0x3154, 0x7073, 0x7074, 0x7076, 0x3461, 0x7071,
01621 0x7077, 0x707a, 0x7078, 0x7075, 0x707d, 0x7079, 0x707c, 0x707e,
01622 0x7121, 0x4e41, 0x7124, 0x7123, 0x4176, 0x707b, 0x4a5d, 0x3471,
01623 0x3171, 0x4c31, 0x7126, 0x7127, 0x712c, 0x554e, 0x7129, 0x4833,
01624 0x7122, 0x712b, 0x7128, 0x7125, 0x712a, 0x3029, 0x712d, 0x712f,
01625 0x7131, 0x7130, 0x712e, 0x5122, 0x7132, 0x7133, 0x396f, 0x3547,
01626 0x3057, 0x3059, 0x546d, 0x3544, 0x3d54, 0x3b4a, 0x7027, 0x385e,
01627 0x7028, 0x3028, 0x7029, 0x4d6e, 0x702a, 0x702b, 0x4624, 0x5665,
01628 0x7164, 0x7165, 0x4373, 0x535b, 0x5651, 0x4568, 0x532f, 0x5266,
01629 0x6e41, 0x303b, 0x5535, 0x514e, 0x3c60, 0x3a50, 0x3f78, 0x3847,
01630 0x3541, 0x454c, 0x4a22, 0x434b, 0x6e42, 0x443f, 0x3622, 0x6d6c,
01631 0x4324, 0x5631, 0x4f60, 0x6d6f, 0x454e, 0x365c, 0x4a21, 0x6d6d,
01632 0x6d70, 0x6d71, 0x433c, 0x3f34, 0x6d6e, 0x6d74, 0x6d72, 0x5566,
01633 0x435f, 0x6d73, 0x6d76, 0x5523, 0x5123, 0x6d75, 0x4350, 0x6d77,
01634 0x3f74, 0x3e6c, 0x6d78, 0x4c77, 0x515b, 0x5745, 0x5576, 0x6d7c,
01635 0x6d7b, 0x6d79, 0x6d7a, 0x6d7d, 0x3e26, 0x4b2f, 0x6e21, 0x363d,
01636 0x6e22, 0x4440, 0x6d7e, 0x3d5e, 0x3247, 0x3643, 0x6e25, 0x583a,
01637 0x6e23, 0x6e26, 0x4369, 0x3372, 0x6e27, 0x6e24, 0x4f39, 0x6e28,
01638 0x4277, 0x6e29, 0x6e2a, 0x5e2b, 0x4633, 0x4746, 0x5675, 0x3549,
01639 0x4b32, 0x6e2b, 0x4d2b, 0x6e2c, 0x5530, 0x6e2d, 0x7644, 0x5b47,
01640 0x3423, 0x432c, 0x7166, 0x4a38, 0x5253, 0x562a, 0x6f72, 0x3e58,
01641 0x3d43, 0x6f73, 0x364c, 0x302b, 0x4a2f, 0x6d36, 0x6d37, 0x4e79,
01642 0x372f, 0x3f73, 0x6d38, 0x426b, 0x4930, 0x6d39, 0x4676, 0x3f33,
01643 0x6d3c, 0x4578, 0x5150, 0x5729, 0x6d3a, 0x6d3b, 0x5162, 0x6d3f,
01644 0x6d40, 0x6d44, 0x6d48, 0x6d46, 0x6d4e, 0x5568, 0x6d49, 0x6d47,
01645 0x6d3e, 0x4569, 0x4646, 0x4969, 0x5452, 0x6d41, 0x6d42, 0x6d43,
01646 0x6d45, 0x4079, 0x3421, 0x3968, 0x6d50, 0x6d51, 0x6d4a, 0x6d4f,
01647 0x4e78, 0x4b36, 0x6d4c, 0x6d4d, 0x4f75, 0x6d52, 0x4172, 0x5332,
01648 0x6d4b, 0x4837, 0x3c6f, 0x4570, 0x6d56, 0x356f, 0x4235, 0x302d,
01649 0x4b69, 0x312e, 0x6d54, 0x4d6b, 0x3562, 0x6d55, 0x6d53, 0x6d57,
01650 0x357a, 0x6d58, 0x6d59, 0x6d5c, 0x314c, 0x4576, 0x3c6e, 0x6d5a,
01651 0x4c3c, 0x326a, 0x6d5b, 0x446b, 0x3445, 0x3075, 0x6d5f, 0x405a,
01652 0x3468, 0x454d, 0x6d5d, 0x3f44, 0x6d5e, 0x4425, 0x6d60, 0x6d61,
01653 0x6d63, 0x4157, 0x3b47, 0x3d38, 0x6d62, 0x6d64, 0x6d66, 0x6d65,
01654 0x6d67, 0x4a3e, 0x6c6a, 0x4071, 0x4967, 0x6c6b, 0x466e, 0x6c6c,
01655 0x466d, 0x6c6d, 0x6c70, 0x5766, 0x6c73, 0x6c71, 0x6c6e, 0x6c6f,
01656 0x5723, 0x4971, 0x4b6e, 0x6c74, 0x6c72, 0x4f69, 0x6c76, 0x4631,
01657 0x3c40, 0x6c75, 0x353b, 0x3b76, 0x6c77, 0x5977, 0x3d7b, 0x423b,
01658 0x6c78, 0x6c79, 0x3823, 0x6c7a, 0x6c7b, 0x6c7c, 0x536d, 0x582e,
01659 0x406b, 0x475d, 0x3a4c, 0x5063, 0x4b3d, 0x4d3a, 0x3851, 0x317c,
01660 0x476f, 0x5656, 0x3f46, 0x436b, 0x6f75, 0x4358, 0x5762, 0x6f77,
01661 0x3353, 0x4758, 0x516d, 0x5648, 0x6f78, 0x6f76, 0x3b7d, 0x3346,
01662 0x3d55, 0x5246, 0x3b60, 0x4f21, 0x6f7c, 0x6f7b, 0x6f79, 0x334c,
01663 0x4954, 0x4b30, 0x6f7e, 0x305e, 0x5649, 0x6f7d, 0x336d, 0x7655,
01664 0x4e48, 0x7022, 0x7021, 0x353e, 0x3c5a, 0x3b7c, 0x3865, 0x4442,
01665 0x7023, 0x4b6b, 0x7026, 0x5128, 0x3e3f, 0x476e, 0x7136, 0x7137,
01666 0x3f55, 0x3429, 0x7138, 0x4d3b, 0x4754, 0x552d, 0x7139, 0x713a,
01667 0x474f, 0x5224, 0x564f, 0x713b, 0x3d51, 0x3430, 0x3e3d, 0x345c,
01668 0x4e51, 0x3f5f, 0x713d, 0x3f7a, 0x713c, 0x713f, 0x713e, 0x7140,
01669 0x7141, 0x417e, 0x4122, 0x4a7a, 0x553e, 0x3e3a, 0x3e39, 0x5542,
01670 0x3f22, 0x4d2f, 0x7135, 0x3d5f, 0x364b, 0x5671, 0x7343, 0x7344,
01671 0x384d, 0x7346, 0x7347, 0x304a, 0x7345, 0x7349, 0x4b71, 0x734b,
01672 0x5026, 0x314a, 0x7348, 0x734f, 0x3551, 0x7357, 0x7352, 0x7354,
01673 0x7353, 0x377b, 0x313f, 0x734e, 0x734a, 0x355a, 0x7350, 0x7351,
01674 0x7355, 0x734d, 0x3c63, 0x417d, 0x7356, 0x735a, 0x734c, 0x3548,
01675 0x3d6e, 0x735c, 0x3724, 0x3f70, 0x567e, 0x4d32, 0x3470, 0x325f,
01676 0x7358, 0x7359, 0x4938, 0x735d, 0x735e, 0x7361, 0x735f, 0x7363,
```

01677 0x7362, 0x735b, 0x3f6a, 0x336f, 0x7360, 0x4729, 0x3c72, 0x736b,
01678 0x393f, 0x7364, 0x322d, 0x3b7e, 0x4b63, 0x736d, 0x7369, 0x395c,
01679 0x736e, 0x7365, 0x7366, 0x736a, 0x4261, 0x736c, 0x736f, 0x7368,
01680 0x3c7d, 0x4f64, 0x7370, 0x7367, 0x7372, 0x572d, 0x462a, 0x7373,
01681 0x7371, 0x4228, 0x385d, 0x7375, 0x7374, 0x345b, 0x7376, 0x7377,
01682 0x7378, 0x403a, 0x4069, 0x4571, 0x737b, 0x737a, 0x3458, 0x737e,
01683 0x7379, 0x737c, 0x737d, 0x7421, 0x7423, 0x3b49, 0x7422, 0x7424,
01684 0x323e, 0x7426, 0x7425, 0x3c2e, 0x4357, 0x5961, 0x4060, 0x744c,
01685 0x5751, 0x375b, 0x744e, 0x4123, 0x4649, 0x3456, 0x5533, 0x7450,
01686 0x744f, 0x7451, 0x4b5a, 0x7452, 0x5441, 0x5660, 0x3760, 0x4138,
01687 0x413b, 0x7453, 0x3e2c, 0x3462, 0x7454, 0x7455, 0x3e2b, 0x7456,
01688 0x745b, 0x7457, 0x745a, 0x3a7d, 0x7458, 0x7459, 0x3862, 0x4c47,
01689 0x745c, 0x325a, 0x4353, 0x5463, 0x3f37, 0x745d, 0x4534, 0x7469,
01690 0x4f35, 0x4e49, 0x4b58, 0x4b77, 0x3d74, 0x574f, 0x405b, 0x5075,
01691 0x746a, 0x746b, 0x746c, 0x746e, 0x7763, 0x3731, 0x746d, 0x576b, 0x746e,
01692 0x6679, 0x3e40, 0x667a, 0x3a6c, 0x667b, 0x4f4b, 0x667c, 0x543c,
01693 0x3c36, 0x667d, 0x667e, 0x3c4d, 0x4852, 0x4e33, 0x6721, 0x343f,
01694 0x6722, 0x4934, 0x3859, 0x4449, 0x575d, 0x425a, 0x3757, 0x563d,
01695 0x4e46, 0x3744, 0x4526, 0x6723, 0x4f5f, 0x6724, 0x6725, 0x6726,
01696 0x4137, 0x5769, 0x4970, 0x4f38, 0x562f, 0x5655, 0x6727, 0x306d,
01697 0x6728, 0x6729, 0x495c, 0x526f, 0x3e2d, 0x672a, 0x3073, 0x485e,
01698 0x3d61, 0x672b, 0x4846, 0x672c, 0x3b66, 0x3878, 0x5124, 0x672d,
01699 0x4267, 0x3e78, 0x3d4a, 0x4d33, 0x672e, 0x672f, 0x3e6e, 0x5065,
01700 0x4b67, 0x4c50, 0x3c4c, 0x6730, 0x3c28, 0x5077, 0x6731, 0x5078,
01701 0x6732, 0x6733, 0x3442, 0x6734, 0x6735, 0x497e, 0x4e2c, 0x4360,
01702 0x6737, 0x3141, 0x3371, 0x6738, 0x6739, 0x575b, 0x5540, 0x673a,
01703 0x424c, 0x573a, 0x673b, 0x673c, 0x673d, 0x3c6a, 0x4365, 0x4042,
01704 0x673e, 0x673f, 0x3c29, 0x6740, 0x6741, 0x6736, 0x3650, 0x6742,
01705 0x6743, 0x6744, 0x3b3a, 0x355e, 0x4246, 0x3160, 0x6745, 0x5435,
01706 0x6746, 0x383f, 0x6748, 0x6747, 0x376c, 0x6749, 0x3278, 0x674a,
01707 0x674b, 0x674c, 0x674d, 0x674e, 0x674f, 0x6750, 0x5327, 0x4b75,
01708 0x6751, 0x6752, 0x6753, 0x6754, 0x4949, 0x6755, 0x6756, 0x6757,
01709 0x6758, 0x6759, 0x3d49, 0x675a, 0x733e, 0x3857, 0x4831, 0x733f,
01710 0x7340, 0x7341, 0x395e, 0x4d78, 0x5868, 0x3a31, 0x425e, 0x6e37,
01711 0x3723, 0x6e39, 0x6e38, 0x3055, 0x6e3b, 0x5556, 0x576f, 0x5643,
01712 0x6e3d, 0x4a70, 0x6e3c, 0x6e3e, 0x6e40, 0x6e3f, 0x5172, 0x473c,
01713 0x4340, 0x3861, 0x4167, 0x7446, 0x505f, 0x7447, 0x4f5b, 0x483a,
01714 0x7448, 0x7449, 0x744a, 0x744b, 0x597a, 0x387e, 0x6571, 0x5370,
01715 0x7460, 0x4e4c, 0x3361, 0x7134, 0x526e, 0x7461, 0x4f68, 0x7462,
01716 0x474c, 0x3554, 0x3464, 0x7464, 0x7463, 0x7465, 0x7466, 0x7467,
01717 0x3a32, 0x303f, 0x7468, 0x372d, 0x526d, 0x522b, 0x404f, 0x3f3c,
01718 0x6b23, 0x555e, 0x6a48, 0x7173, 0x3678, 0x4b23, 0x444d, 0x7167,
01719 0x7168, 0x387b, 0x7169, 0x3a44, 0x5445, 0x3052, 0x716a, 0x716b,
01720 0x716c, 0x716d, 0x716e, 0x716f, 0x7171, 0x7170, 0x4555, 0x7172,
01721 0x367a, 0x7174, 0x522e, 0x5e47, 0x4b4a, 0x335c, 0x3522, 0x3922,
01722 0x4474, 0x7175, 0x7176, 0x4144, 0x417b, 0x5630, 0x7177, 0x7178,
01723 0x412a, 0x4638, 0x3e5b, 0x7179, 0x344f, 0x717a, 0x6d32, 0x6d31,
01724 0x4b60, 0x525e, 0x4b41, 0x5558, 0x4862, 0x405f, 0x3c21, 0x6b41,
01725 0x5024, 0x5662, 0x3647, 0x3858, 0x6b40, 0x384e, 0x6b3f, 0x3326,
01726 0x3949, 0x562b, 0x3774, 0x374a, 0x3c67, 0x373e, 0x6b46, 0x6b47,
01727 0x3039, 0x3f4f, 0x6b45, 0x537d, 0x6b48, 0x6b49, 0x374e, 0x6b42,
01728 0x6b44, 0x4976, 0x5657, 0x554d, 0x5032, 0x6b4f, 0x4e38, 0x6b50,
01729 0x3528, 0x3133, 0x6b52, 0x4c25, 0x4556, 0x6b53, 0x6b51, 0x455f,
01730 0x6b4e, 0x4a24, 0x6b55, 0x307b, 0x3a7a, 0x5837, 0x7163, 0x6b4a,
01731 0x6b4b, 0x6b4c, 0x6b4d, 0x6b56, 0x6640, 0x6b59, 0x3f68, 0x5248,
01732 0x6b57, 0x6b5c, 0x386c, 0x6b58, 0x3d3a, 0x5058, 0x3037, 0x6b5d,
01733 0x445c, 0x562c, 0x3460, 0x4276, 0x3c39, 0x6b5a, 0x6b5b, 0x5460,
01734 0x466a, 0x4454, 0x6b5f, 0x4527, 0x5975, 0x3231, 0x6b64, 0x3d45,
01735 0x6b62, 0x6b63, 0x382c, 0x4d51, 0x6b65, 0x6b61, 0x4133, 0x4622,
01736 0x4c73, 0x6b66, 0x4030, 0x5238, 0x6b67, 0x382f, 0x382d, 0x6b68,
01737 0x473b, 0x4d73, 0x6b6a, 0x6b6b, 0x6b6d, 0x5048, 0x6b72, 0x6b6e,
01738 0x6b71, 0x4879, 0x517c, 0x6b6c, 0x6b69, 0x3839, 0x4f59, 0x4465,
01739 0x6b6f, 0x6b70, 0x4c5a, 0x4d48, 0x3072, 0x6b76, 0x6b75, 0x3232,
01740 0x3860, 0x6b77, 0x316c, 0x4c45, 0x4424, 0x4f25, 0x6b79, 0x6c22,
01741 0x4572, 0x6b7a, 0x4945, 0x625f, 0x6b7e, 0x4d4e, 0x6c21, 0x315b,
01742 0x5337, 0x525c, 0x6b7d, 0x6b7b, 0x333c, 0x6a30, 0x5754, 0x742b,
01743 0x3374, 0x5641, 0x5642, 0x5569, 0x3e4a, 0x7427, 0x5228, 0x7428,
01744 0x7429, 0x742a, 0x3e4b, 0x535f, 0x4960, 0x4961, 0x7342, 0x4a66,
01745 0x4c72, 0x6236, 0x4b34, 0x4e68, 0x565b, 0x742d, 0x742e, 0x742f,
01746 0x7432, 0x3a3d, 0x7433, 0x3063, 0x7430, 0x7431, 0x3d22, 0x3255,
01747 0x7436, 0x7437, 0x3666, 0x3230, 0x4f4f, 0x7434, 0x342c, 0x7435,
01748 0x7438, 0x7439, 0x4d27, 0x743a, 0x743b, 0x743c, 0x4b52, 0x743d,
01749 0x743e, 0x743f, 0x745e, 0x413c, 0x3c68, 0x492b, 0x515e, 0x6575,
01750 0x5c33, 0x5255, 0x5c34, 0x302c, 0x5c35, 0x3d5a, 0x5c39, 0x5842,
01751 0x5c37, 0x5373, 0x4956, 0x5c3a, 0x5c36, 0x5c3b, 0x4322, 0x5c3c,
01752 0x5c45, 0x5c3d, 0x4e5f, 0x5625, 0x5c4f, 0x5c4d, 0x5c52, 0x3d66,
01753 0x422b, 0x5c38, 0x5c4b, 0x5c4e, 0x5c3e, 0x3752, 0x3045, 0x5c47,
01754 0x503e, 0x5c41, 0x3b28, 0x373c, 0x5c4c, 0x5c46, 0x5c3f, 0x475b,
01755 0x513f, 0x5c40, 0x5c4a, 0x5c50, 0x4e2d, 0x5c42, 0x5c43, 0x5c48,
01756 0x5c49, 0x3254, 0x5c51, 0x4b55, 0x5437, 0x5c5b, 0x5c5f, 0x4c26,
01757 0x5c66, 0x4367, 0x5c5c, 0x3f41, 0x5c59, 0x307a, 0x3936, 0x5c65,
01758 0x5c53, 0x5c44, 0x5c56, 0x4874, 0x3f60, 0x493b, 0x313d, 0x5322,
01759 0x5c5a, 0x5c55, 0x463b, 0x5c5e, 0x5742, 0x432f, 0x3736, 0x4751,
01760 0x4329, 0x5c62, 0x5c58, 0x5c6b, 0x5c54, 0x5c5d, 0x3e25, 0x5c57,
01761 0x5c60, 0x5c63, 0x5c64, 0x5c78, 0x5c61, 0x5d22, 0x5c67, 0x3c6b,
01762 0x3444, 0x4323, 0x3267, 0x5c7a, 0x5c72, 0x5c6f, 0x5c7c, 0x5c6e,
01763 0x5270, 0x3268, 0x4857, 0x4863, 0x5c7b, 0x5c6d, 0x5c77, 0x5c75,

```

01764 0x3e23, 0x5c74, 0x325d, 0x5c73, 0x3c76, 0x5c68, 0x3b44, 0x4073,
01765 0x3c54, 0x5c69, 0x5c6a, 0x5c71, 0x5c76, 0x5c79, 0x3534, 0x4859,
01766 0x3b67, 0x5c7e, 0x5c7d, 0x532b, 0x5d21, 0x5d23, 0x5d25, 0x5271,
01767 0x5d24, 0x5d26, 0x5d27, 0x5229, 0x3a49, 0x5d29, 0x5d36, 0x5d31,
01768 0x5d34, 0x5d30, 0x464e, 0x4072, 0x492f, 0x5c6c, 0x5d2e, 0x5d37,
01769 0x5c70, 0x5d2f, 0x5d38, 0x5d2c, 0x5d39, 0x5d33, 0x5d2d, 0x442a,
01770 0x5d28, 0x4033, 0x412b, 0x5d2a, 0x5d2b, 0x5d32, 0x3b71, 0x5d35,
01771 0x5328, 0x5d3a, 0x5d3b, 0x4327, 0x5d52, 0x5d3c, 0x5d51, 0x393d,
01772 0x3e55, 0x3e7a, 0x3a4a, 0x5d4a, 0x5d45, 0x5d3f, 0x324b, 0x5d43,
01773 0x5d4b, 0x3224, 0x5d55, 0x5d3e, 0x4650, 0x5d50, 0x5d54, 0x4162,
01774 0x3746, 0x5d4e, 0x5d4f, 0x5d44, 0x5d3d, 0x5d4d, 0x4c51, 0x5d49,
01775 0x5d42, 0x4348, 0x463c, 0x4e2e, 0x5d4c, 0x5d48, 0x5d41, 0x5d46,
01776 0x425c, 0x5329, 0x532a, 0x5d53, 0x4f74, 0x4878, 0x5d66, 0x5d47,
01777 0x5d60, 0x4264, 0x5d61, 0x5d57, 0x5678, 0x5d59, 0x5d58, 0x3870,
01778 0x5d56, 0x464f, 0x464d, 0x362d, 0x5d62, 0x3a79, 0x5461, 0x5d67, 0x3450,
01779 0x5d5a, 0x3f7b, 0x5d63, 0x5d5f, 0x5d5d, 0x3559, 0x5d5b, 0x5d5c,
01780 0x5d5e, 0x3d2f, 0x5d64, 0x5d65, 0x5d75, 0x4349, 0x4b62, 0x5d72,
01781 0x5861, 0x4651, 0x5d74, 0x5574, 0x5d73, 0x5d70, 0x5d6c, 0x5d6f,
01782 0x5d68, 0x506e, 0x4858, 0x5d6e, 0x5d69, 0x5d6a, 0x4b72, 0x5d6d,
01783 0x314d, 0x4036, 0x3c3b, 0x5d71, 0x5d77, 0x5d76, 0x5d6b, 0x456e,
01784 0x5d7b, 0x5e24, 0x5e23, 0x5d78, 0x436f, 0x427b, 0x5561, 0x4e35,
01785 0x5d7d, 0x324c, 0x4468, 0x4a5f, 0x473e, 0x5d7a, 0x5d7c, 0x5d7e,
01786 0x5e22, 0x302a, 0x314e, 0x5e2c, 0x5e26, 0x3d36, 0x486f, 0x5e21,
01787 0x5e25, 0x5e29, 0x5e28, 0x5e27, 0x5e2d, 0x544c, 0x5e33, 0x5e2a,
01788 0x5e2e, 0x4059, 0x3121, 0x5e36, 0x5e31, 0x5e32, 0x5126, 0x5e35,
01789 0x5e2f, 0x5e30, 0x503d, 0x5e34, 0x4a6d, 0x5e39, 0x5e38, 0x5e37,
01790 0x5e3b, 0x3d65, 0x3258, 0x436a, 0x5e3a, 0x453a, 0x5e3c, 0x4c59,
01791 0x372a, 0x5465, 0x5e3d, 0x5e3f, 0x4422, 0x5e41, 0x5e3e, 0x5e40,
01792 0x553a, 0x5e42, 0x722e, 0x3b22, 0x4232, 0x4530, 0x4247, 0x722f,
01793 0x5069, 0x535d, 0x6b3d, 0x3366, 0x7230, 0x7231, 0x4a2d, 0x3a67,
01794 0x7233, 0x7235, 0x7234, 0x4b64, 0x4f3a, 0x7232, 0x4a34, 0x524f,
01795 0x426c, 0x4e43, 0x7238, 0x3076, 0x7237, 0x723e, 0x324f, 0x5141,
01796 0x723a, 0x723c, 0x7265, 0x5469, 0x723b, 0x7236, 0x723f, 0x723d,
01797 0x7247, 0x7244, 0x7246, 0x724a, 0x7242, 0x7240, 0x7245, 0x567b,
01798 0x7241, 0x4779, 0x495f, 0x7248, 0x3946, 0x3530, 0x7243, 0x7249,
01799 0x7250, 0x7256, 0x3b57, 0x7255, 0x4d5c, 0x566b, 0x7252, 0x7254,
01800 0x3872, 0x724b, 0x724e, 0x4279, 0x555d, 0x724c, 0x724d, 0x724f,
01801 0x7253, 0x7259, 0x533c, 0x366a, 0x4a71, 0x3764, 0x7257, 0x7258,
01802 0x725a, 0x725d, 0x725b, 0x725c, 0x5151, 0x7251, 0x4d49, 0x4e4f,
01803 0x5629, 0x7263, 0x435b, 0x7260, 0x402f, 0x726c, 0x725e, 0x7261,
01804 0x7268, 0x7262, 0x7267, 0x7266, 0x7269, 0x725f, 0x7264, 0x726a,
01805 0x532c, 0x7265, 0x3275, 0x7272, 0x502b, 0x7275, 0x3b48, 0x7279,
01806 0x7270, 0x7276, 0x7278, 0x727a, 0x7273, 0x7271, 0x3a7b, 0x357b,
01807 0x726f, 0x7277, 0x726d, 0x726e, 0x726b, 0x7326, 0x7323, 0x7322,
01808 0x7274, 0x485a, 0x727b, 0x7325, 0x4378, 0x727d, 0x7327, 0x7329,
01809 0x7324, 0x727c, 0x732b, 0x732a, 0x425d, 0x732e, 0x7330, 0x7321,
01810 0x7331, 0x732c, 0x732f, 0x727e, 0x732d, 0x7332, 0x7334, 0x7328,
01811 0x7333, 0x7335, 0x5037, 0x7338, 0x5979, 0x7339, 0x7337, 0x4864,
01812 0x7336, 0x733a, 0x733b, 0x3440, 0x6e43, 0x733c, 0x733d, 0x512a,
01813 0x742c, 0x5046, 0x5050, 0x515c, 0x4f4e, 0x3d56, 0x5143, 0x3a62,
01814 0x6169, 0x5242, 0x7142, 0x3239, 0x316d, 0x7143, 0x4940, 0x3344,
01815 0x5972, 0x4b25, 0x7144, 0x5654, 0x7145, 0x7440, 0x7146, 0x542c,
01816 0x7147, 0x3040, 0x7441, 0x7442, 0x347c, 0x455b, 0x4c3b, 0x5064,
01817 0x4d60, 0x7148, 0x5973, 0x313b, 0x4f2e, 0x3824, 0x714a, 0x714b,
01818 0x3243, 0x4151, 0x5730, 0x7149, 0x714c, 0x714e, 0x5976, 0x5261,
01819 0x5423, 0x7443, 0x4839, 0x7444, 0x714d, 0x714f, 0x3f63, 0x7150,
01820 0x7154, 0x7156, 0x7151, 0x4951, 0x4561, 0x4263, 0x397c, 0x7153,
01821 0x7155, 0x3953, 0x715b, 0x3a56, 0x307d, 0x7159, 0x7158, 0x7152,
01822 0x715a, 0x7157, 0x486c, 0x4d4a, 0x715d, 0x653d, 0x715c, 0x715e,
01823 0x715f, 0x4f65, 0x7445, 0x3d73, 0x7160, 0x7161, 0x4e77, 0x522a,
01824 0x717b, 0x3832, 0x3c7b, 0x395b, 0x3966, 0x4359, 0x4a53, 0x6a68,
01825 0x4040, 0x3e75, 0x6a69, 0x6a6a, 0x6a6b, 0x6a6c, 0x6a6d, 0x6a6e,
01826 0x6a6f, 0x3d47, 0x757b, 0x757d, 0x757e, 0x757c, 0x3d62, 0x7621,
01827 0x3425, 0x7622, 0x7623, 0x6c32, 0x5154, 0x596a, 0x7624, 0x6e3a,
01828 0x5532, 0x537e, 0x4c5c, 0x4a44, 0x6540, 0x7625, 0x3e2f, 0x4629,
01829 0x5a25, 0x3c46, 0x3629, 0x383c, 0x484f, 0x3c25, 0x5a26, 0x5a27,
01830 0x4c56, 0x4843, 0x5a28, 0x467d, 0x5135, 0x5269, 0x5136, 0x3c47,
01831 0x3d32, 0x3b64, 0x5a29, 0x5a2a, 0x5148, 0x5a2b, 0x506d, 0x366f,
01832 0x425b, 0x4b4f, 0x376d, 0x4968, 0x3743, 0x3e77, 0x5624, 0x5a2c,
01833 0x5a2d, 0x4640, 0x5767, 0x4a36, 0x5529, 0x4b5f, 0x556f, 0x5a2e,
01834 0x565f, 0x344a, 0x5a30, 0x5a2f, 0x526b, 0x5a31, 0x5a32, 0x5a33,
01835 0x4a54, 0x5a34, 0x4a2b, 0x5a35, 0x5a36, 0x334f, 0x566f, 0x5a37,
01836 0x3b30, 0x352e, 0x5a38, 0x5a39, 0x396e, 0x512f, 0x5268, 0x5a3a,
01837 0x3843, 0x4f6a, 0x326f, 0x5a3b, 0x5a3c, 0x3d6b, 0x4e5c, 0x536f,
01838 0x5a3d, 0x4e73, 0x5a3e, 0x5355, 0x3b65, 0x5a3f, 0x4b35, 0x4b50,
01839 0x5a40, 0x476b, 0x566e, 0x5a41, 0x4535, 0x3641, 0x5a42, 0x374c,
01840 0x3f4e, 0x5a43, 0x5a44, 0x4b2d, 0x5a45, 0x3577, 0x5a46, 0x4142,
01841 0x573b, 0x5a47, 0x4c38, 0x526a, 0x4431, 0x5a48, 0x357d, 0x3b51,
01842 0x5a49, 0x5033, 0x5a4a, 0x5a4b, 0x4e3d, 0x5a4c, 0x5a4d, 0x5a4e,
01843 0x3277, 0x5a51, 0x5a4f, 0x5168, 0x5a50, 0x4355, 0x5a52, 0x5a53,
01844 0x5a54, 0x5a55, 0x503b, 0x5225, 0x3079, 0x5a56, 0x472b, 0x5a57,
01845 0x3d77, 0x4321, 0x5a58, 0x5a59, 0x437d, 0x4c37, 0x5a5a, 0x5a5b,
01846 0x403e, 0x4657, 0x5a5c, 0x5a5d, 0x4734, 0x5a5e, 0x5a5f, 0x3948,
01847 0x3b6d, 0x3639, 0x7478, 0x7479, 0x4d63, 0x7539, 0x6b60, 0x4f73,
01848 0x3b3f, 0x3a40, 0x5425, 0x6159, 0x7574, 0x312a, 0x3272, 0x7575,
01849 0x7577, 0x3a51, 0x7576, 0x4332, 0x7579, 0x7578, 0x3134, 0x556a,
01850 0x383a, 0x3931, 0x3246, 0x5470, 0x4f4d, 0x305c, 0x554b, 0x3b75,

```

01851 0x564a, 0x3737, 0x4c30, 0x4636, 0x3161, 0x393a, 0x567c, 0x3961,
01852 0x3721, 0x3c7a, 0x6a5a, 0x6a5b, 0x4c79, 0x3973, 0x6a5e, 0x347b,
01853 0x4333, 0x3751, 0x3a58, 0x6a5d, 0x5474, 0x6a5e, 0x3c56, 0x3b5f,
01854 0x6a5f, 0x415e, 0x4238, 0x545f, 0x574a, 0x6a60, 0x6a61, 0x6a64,
01855 0x6a62, 0x6a63, 0x495e, 0x3833, 0x3644, 0x6a65, 0x4a6a, 0x494d,
01856 0x344d, 0x6259, 0x4562, 0x6a66, 0x4035, 0x5738, 0x6a67, 0x572c,
01857 0x487c, 0x5853, 0x584d, 0x545e, 0x5479, 0x4944, 0x532e, 0x3853,
01858 0x3360, 0x4962, 0x7476, 0x3a55, 0x7477, 0x575f, 0x7471, 0x3830,
01859 0x5554, 0x384f, 0x4670, 0x3343, 0x7472, 0x332c, 0x543d, 0x4777,
01860 0x7474, 0x7473, 0x4c4b, 0x4824, 0x7475, 0x5763, 0x453f, 0x7540,
01861 0x753b, 0x7543, 0x7542, 0x563a, 0x7541, 0x543e, 0x7544, 0x754c,
01862 0x304f, 0x3578, 0x7549, 0x754a, 0x455c, 0x7545, 0x7546, 0x7547,
01863 0x754b, 0x3e60, 0x7548, 0x387a, 0x7550, 0x7553, 0x3f67, 0x3972,
01864 0x753c, 0x754d, 0x4237, 0x4c78, 0x3c79, 0x754e, 0x754f, 0x7551,
01865 0x3665, 0x7552, 0x7555, 0x755d, 0x7554, 0x533b, 0x336c, 0x4c24,
01866 0x7556, 0x7557, 0x3e61, 0x7558, 0x4c5f, 0x755b, 0x3248, 0x5759,
01867 0x7559, 0x755a, 0x755c, 0x7562, 0x7560, 0x755f, 0x755d, 0x7561,
01868 0x755e, 0x7564, 0x7565, 0x4c63, 0x653f, 0x3538, 0x7563, 0x7568,
01869 0x4c23, 0x7566, 0x7567, 0x753e, 0x3144, 0x753f, 0x3545, 0x3264,
01870 0x756c, 0x7569, 0x3657, 0x756d, 0x756a, 0x756b, 0x345a, 0x546a,
01871 0x756e, 0x3379, 0x756f, 0x7571, 0x7570, 0x7572, 0x7573, 0x496d,
01872 0x392a, 0x477b, 0x3663, 0x4c49, 0x6a26, 0x3335, 0x547e, 0x396c,
01873 0x5079, 0x696d, 0x572a, 0x696e, 0x4256, 0x486d, 0x3a64, 0x696f,
01874 0x6970, 0x6971, 0x5661, 0x6972, 0x6973, 0x6975, 0x6976, 0x6977,
01875 0x6977, 0x4761, 0x6978, 0x5458, 0x6979, 0x3d4e, 0x697a, 0x697b,
01876 0x3d4f, 0x697c, 0x3828, 0x413e, 0x697d, 0x3132, 0x3b54, 0x3975,
01877 0x697e, 0x6a21, 0x6a22, 0x6a23, 0x3778, 0x3c2d, 0x4a64, 0x604e,
01878 0x542f, 0x4f3d, 0x5537, 0x6a24, 0x555e, 0x6a25, 0x5041, 0x393c,
01879 0x3447, 0x3159, 0x4031, 0x3166, 0x3167, 0x3168, 0x333d, 0x4868,
01880 0x6541, 0x315f, 0x4149, 0x346f, 0x4728, 0x5358, 0x4679, 0x5138,
01881 0x397d, 0x4275, 0x532d, 0x544b, 0x3d7c, 0x6542, 0x3735, 0x6543,
01882 0x3b39, 0x5562, 0x3d78, 0x5436, 0x4e25, 0x412c, 0x3359, 0x4c76,
01883 0x6546, 0x6544, 0x6548, 0x654a, 0x6547, 0x354f, 0x4648, 0x357c,
01884 0x6545, 0x4a76, 0x6549, 0x4354, 0x3145, 0x3c23, 0x5737, 0x4d4b,
01885 0x4b4d, 0x4a4a, 0x4c53, 0x654c, 0x654b, 0x4466, 0x5121, 0x5137,
01886 0x654d, 0x6550, 0x4d38, 0x5670, 0x654f, 0x355d, 0x4d3e, 0x6551,
01887 0x363a, 0x4d28, 0x3964, 0x4a45, 0x3351, 0x4b59, 0x546c, 0x6552,
01888 0x376a, 0x654e, 0x6555, 0x347e, 0x6556, 0x6553, 0x6554, 0x525d,
01889 0x425f, 0x3146, 0x5362, 0x365d, 0x4b6c, 0x6557, 0x5376, 0x3169,
01890 0x3674, 0x655a, 0x6558, 0x6559, 0x3540, 0x5245, 0x655c, 0x655e,
01891 0x655d, 0x4732, 0x5223, 0x655b, 0x5462, 0x555a, 0x6560, 0x5771,
01892 0x6561, 0x315c, 0x517b, 0x6562, 0x6564, 0x6563, 0x6565, 0x5258,
01893 0x354b, 0x675f, 0x5a75, 0x5a78, 0x5a76, 0x5a77, 0x5a7a, 0x504f,
01894 0x4447, 0x306e, 0x5030, 0x5a79, 0x534a, 0x3a2a, 0x5b22, 0x4771,
01895 0x5a7c, 0x5a7b, 0x495b, 0x5a7d, 0x5b21, 0x575e, 0x5a7e, 0x415a,
01896 0x5b25, 0x5374, 0x5b27, 0x5b24, 0x5b28, 0x3d3c, 0x4049, 0x5b23,
01897 0x5b26, 0x5623, 0x5b29, 0x5b2d, 0x5b2e, 0x5b2c, 0x3a42, 0x3f24,
01898 0x5b2b, 0x5b2a, 0x5447, 0x323f, 0x5b2f, 0x3979, 0x5b30, 0x333b,
01899 0x3526, 0x363c, 0x5b31, 0x3675, 0x5b32, 0x3149, 0x5b34, 0x5b33,
01900 0x5b35, 0x5b37, 0x5b36, 0x5b38, 0x5b39, 0x5b3a, 0x534f, 0x747a,
01901 0x4775, 0x5743, 0x4564, 0x747c, 0x747d, 0x747b, 0x3e46, 0x506f,
01902 0x3753, 0x544d, 0x4c2a, 0x7522, 0x7521, 0x3a28, 0x747e, 0x4b56,
01903 0x7524, 0x4052, 0x336a, 0x4d2a, 0x7525, 0x7523, 0x3d34, 0x7528,
01904 0x7529, 0x3d4d, 0x4338, 0x3f61, 0x4b61, 0x752a, 0x7526, 0x7527,
01905 0x4470, 0x752c, 0x343c, 0x576d, 0x3457, 0x752b, 0x752e, 0x752d,
01906 0x752f, 0x5051, 0x4351, 0x4829, 0x7530, 0x7531, 0x7532, 0x7533,
01907 0x7534, 0x7535, 0x7537, 0x7536, 0x7538, 0x3249, 0x5354, 0x4a4d,
01908 0x406f, 0x5658, 0x5230, 0x413f, 0x3d70, 0x382a, 0x3c78, 0x7646,
01909 0x7647, 0x7648, 0x7649, 0x764a, 0x764c, 0x764b, 0x7769, 0x764d,
01910 0x764e, 0x6e44, 0x6e45, 0x6e46, 0x556b, 0x3624, 0x6e48, 0x6e47,
01911 0x6e49, 0x6e4a, 0x4725, 0x6e4b, 0x6e4c, 0x3730, 0x3576, 0x6e4d,
01912 0x6e4f, 0x6e4e, 0x3846, 0x6e50, 0x6e51, 0x6e52, 0x365b, 0x332e,
01913 0x5653, 0x4446, 0x3135, 0x3856, 0x6e53, 0x6e54, 0x543f, 0x4755,
01914 0x3e7b, 0x4e59, 0x3933, 0x6e56, 0x6e55, 0x6e58, 0x6e57, 0x4525,
01915 0x6e59, 0x6e5a, 0x472e, 0x6e5b, 0x472f, 0x6e5c, 0x3227, 0x6e5d,
01916 0x6e5e, 0x6e5f, 0x6e60, 0x6e61, 0x576a, 0x6e62, 0x6e63, 0x3c58,
01917 0x6e64, 0x534b, 0x4c7a, 0x322c, 0x4165, 0x6e65, 0x4726, 0x432d,
01918 0x6e66, 0x6e67, 0x6e68, 0x6e69, 0x6e6a, 0x6e6b, 0x6e6c, 0x6e6d,
01919 0x6e6e, 0x6e6f, 0x6e70, 0x6e71, 0x6e72, 0x6e74, 0x6e73, 0x6e75,
01920 0x4d2d, 0x4241, 0x6e76, 0x6e77, 0x6e78, 0x5521, 0x6e79, 0x4f33,
01921 0x6e7a, 0x6e7b, 0x6e7c, 0x6e7d, 0x6f21, 0x6e7e, 0x6f22, 0x3875,
01922 0x437a, 0x6f23, 0x6f24, 0x3d42, 0x523f, 0x3279, 0x6f25, 0x6f26,
01923 0x6f27, 0x5278, 0x6f28, 0x567d, 0x6f29, 0x464c, 0x6f2a, 0x6f2b,
01924 0x4134, 0x6f2c, 0x4f7a, 0x4b78, 0x6f2e, 0x6f2d, 0x337a, 0x3978,
01925 0x6f2f, 0x6f30, 0x5062, 0x6f31, 0x6f32, 0x7366, 0x503f, 0x6f33,
01926 0x6f34, 0x6f35, 0x4871, 0x4c60, 0x6f36, 0x6f37, 0x6f38, 0x6f39,
01927 0x6f3a, 0x5560, 0x6f3b, 0x346d, 0x432a, 0x6f3c, 0x6f3d, 0x6f3e,
01928 0x6f3f, 0x4e7d, 0x6f40, 0x4260, 0x3438, 0x5736, 0x3d75, 0x4f47,
01929 0x6f43, 0x6f41, 0x6f42, 0x6f44, 0x3627, 0x3c7c, 0x3e62, 0x434c,
01930 0x6f45, 0x6f46, 0x6f47, 0x6f4f, 0x6f48, 0x6f49, 0x6f4a, 0x4742,
01931 0x6f71, 0x364d, 0x6f4b, 0x6f4c, 0x6f4d, 0x3646, 0x433e, 0x6f4e,
01932 0x6f50, 0x6f51, 0x6f52, 0x5572, 0x6f53, 0x4477, 0x6f54, 0x4478,
01933 0x6f55, 0x6f56, 0x3864, 0x3077, 0x6f57, 0x6f58, 0x6f59, 0x6f5a,
01934 0x6f5b, 0x6f5c, 0x6f5d, 0x6f5e, 0x3e35, 0x6f61, 0x6f5f, 0x6f60,
01935 0x6f62, 0x6f63, 0x414d, 0x6f64, 0x6f65, 0x6f66, 0x6f67, 0x6f68,
01936 0x6f69, 0x6f6a, 0x6f6b, 0x6f6c, 0x4058, 0x6f6d, 0x412d, 0x6f6e,
01937 0x6f6f, 0x6f70, 0x4f62, 0x3324, 0x4345, 0x6345, 0x4941, 0x6346,

```
01938 0x3155, 0x4e4a, 0x3433, 0x4872, 0x6347, 0x4f50, 0x6348, 0x3c64,
01939 0x6349, 0x634a, 0x4346, 0x5522, 0x4456, 0x396b, 0x4e45, 0x634b,
01940 0x4376, 0x634c, 0x3727, 0x3873, 0x3a52, 0x634d, 0x634e, 0x5444,
01941 0x634f, 0x6350, 0x514b, 0x6351, 0x6352, 0x6353, 0x6354, 0x5156,
01942 0x6355, 0x327b, 0x403b, 0x6356, 0x402b, 0x6357, 0x6358, 0x6359,
01943 0x635a, 0x635b, 0x3837, 0x5a62, 0x3653, 0x5a64, 0x5a63, 0x5a66,
01944 0x486e, 0x5a65, 0x3740, 0x5174, 0x5275, 0x5573, 0x3d57, 0x5768,
01945 0x5a68, 0x5a67, 0x3022, 0x4d53, 0x5a69, 0x383d, 0x3c4a, 0x423d,
01946 0x4224, 0x3342, 0x5a6a, 0x422a, 0x4430, 0x3d35, 0x4f5e, 0x5a6b,
01947 0x4942, 0x315d, 0x5a6c, 0x3638, 0x543a, 0x337d, 0x5a6d, 0x5449,
01948 0x4f55, 0x4563, 0x5a6e, 0x5a6f, 0x5a70, 0x416a, 0x4c55, 0x4f5d,
01949 0x5367, 0x4221, 0x5a71, 0x4b65, 0x5a72, 0x4b66, 0x527e, 0x3874,
01950 0x5a73, 0x302f, 0x4f36, 0x554f, 0x4b6d, 0x5a74, 0x6344, 0x4125,
01951 0x763f, 0x7640, 0x7641, 0x4451, 0x4838, 0x5163, 0x505b, 0x5145,
01952 0x3c2f, 0x394d, 0x6f74, 0x3446, 0x533a, 0x7642, 0x337b, 0x7643,
01953 0x3571, 0x7645, 0x536a, 0x7627, 0x5129, 0x7629, 0x7628, 0x4163,
01954 0x4057, 0x3122, 0x4e6d, 0x5068, 0x762b, 0x4f76, 0x762a, 0x5570,
01955 0x762c, 0x4339, 0x3b74, 0x762e, 0x762d, 0x445e, 0x4158, 0x4b2a,
01956 0x4f3c, 0x762f, 0x7630, 0x7631, 0x4236, 0x3054, 0x4579, 0x7632,
01957 0x4760, 0x7626, 0x3e38, 0x3e32, 0x3565, 0x3747, 0x3f3f, 0x4352,
01958 0x4366, 0x584c, 0x584e, 0x386f, 0x3d79, 0x5125, 0x3050, 0x7731,
01959 0x502c, 0x3030, 0x7732, 0x7733, 0x7734, 0x474a, 0x3e4f, 0x7737,
01960 0x7736, 0x315e, 0x7735, 0x7738, 0x7739, 0x4e24, 0x484d, 0x3a2b,
01961 0x6838, 0x6839, 0x683a, 0x3e42, 0x5274, 0x544f, 0x4958, 0x5233,
01962 0x3625, 0x476a, 0x717c, 0x4f6e, 0x4b33, 0x506b, 0x676f, 0x4d67,
01963 0x394b, 0x3659, 0x717d, 0x3064, 0x4b4c, 0x717e, 0x5424, 0x422d,
01964 0x416c, 0x4644, 0x3e31, 0x7221, 0x3c55, 0x7222, 0x7223, 0x7224,
01965 0x5243, 0x4635, 0x4d47, 0x7225, 0x5331, 0x3f45, 0x4c62, 0x7226,
01966 0x7227, 0x5155, 0x366e, 0x7228, 0x7229, 0x355f, 0x722a, 0x722b,
01967 0x327c, 0x722c, 0x722d, 0x4827, 0x3767, 0x6c29, 0x6c2a, 0x6c2b,
01968 0x6c2c, 0x462e, 0x6c2d, 0x6c2e, 0x3749, 0x4a33, 0x6238, 0x774f,
01969 0x7750, 0x324d, 0x7751, 0x7753, 0x7752, 0x623b, 0x3c22, 0x623c,
01970 0x623d, 0x623e, 0x623f, 0x6240, 0x6241, 0x3739, 0x527b, 0x3d24,
01971 0x4a4e, 0x3125, 0x4b47, 0x6242, 0x367c, 0x4844, 0x6243, 0x3d48,
01972 0x317d, 0x6244, 0x3676, 0x6245, 0x4459, 0x6246, 0x4f5a, 0x395d,
01973 0x6247, 0x4021, 0x6248, 0x3276, 0x6249, 0x4173, 0x624a, 0x624b,
01974 0x4278, 0x624c, 0x624d, 0x624e, 0x4a57, 0x5838, 0x5965, 0x4f63,
01975 0x7025, 0x5c30, 0x426d, 0x5426, 0x4d54, 0x5131, 0x335b, 0x477d,
01976 0x3235, 0x423f, 0x6660, 0x4a3b, 0x6661, 0x6662, 0x3e54, 0x6663,
01977 0x5724, 0x4d55, 0x6665, 0x3c5d, 0x6664, 0x6666, 0x6667, 0x426e,
01978 0x3d3e, 0x6668, 0x4266, 0x3a27, 0x6669, 0x666a, 0x3352, 0x5169,
01979 0x3f25, 0x666b, 0x466f, 0x666c, 0x666d, 0x666e, 0x462d, 0x666f,
01980 0x4927, 0x6670, 0x6671, 0x6672, 0x6539, 0x6673, 0x6674, 0x4262,
01981 0x6675, 0x6676, 0x5668, 0x6677, 0x6678, 0x3947, 0x773b, 0x773a,
01982 0x773e, 0x773c, 0x3a21, 0x773f, 0x7740, 0x7742, 0x7741, 0x7744,
01983 0x7743, 0x7745, 0x7746, 0x7747, 0x4b68, 0x385f, 0x7754, 0x7755,
01984 0x7756, 0x7758, 0x775a, 0x7757, 0x775b, 0x7759, 0x5757, 0x775c,
01985 0x775d, 0x775e, 0x775f, 0x7760, 0x5b4b, 0x582a, 0x6577, 0x396d,
01986 0x3f7d, 0x3b6a, 0x7749, 0x4647, 0x7748, 0x774a, 0x774c, 0x774b,
01987 0x774d, 0x4e3a, 0x774e, 0x4427, 0x5363, 0x764f, 0x4233, 0x7650,
01988 0x7651, 0x7652, 0x7653, 0x7654, 0x7656, 0x312b, 0x7657, 0x7658,
01989 0x7659, 0x765a, 0x765b, 0x765c, 0x765d, 0x765e, 0x4f4a, 0x765f,
01990 0x7660, 0x7661, 0x7662, 0x7663, 0x7664, 0x4070, 0x7665, 0x7666,
01991 0x7667, 0x7668, 0x7669, 0x766a, 0x766b, 0x766c, 0x766d, 0x766e,
01992 0x766f, 0x7670, 0x7671, 0x7672, 0x7673, 0x7674, 0x3e28, 0x7675,
01993 0x7676, 0x7677, 0x7678, 0x487a, 0x7679, 0x767a, 0x767b, 0x767c,
01994 0x767d, 0x767e, 0x7721, 0x7722, 0x7723, 0x7724, 0x7725, 0x7726,
01995 0x7727, 0x7728, 0x316e, 0x7729, 0x772a, 0x772b, 0x772c, 0x772d,
01996 0x415b, 0x772e, 0x772f, 0x4471, 0x702f, 0x3c26, 0x7030, 0x4379,
01997 0x4538, 0x513b, 0x7031, 0x7032, 0x7033, 0x7034, 0x7035, 0x513c,
01998 0x516c, 0x7037, 0x7036, 0x5427, 0x4d52, 0x7038, 0x703a, 0x7039,
01999 0x703b, 0x703c, 0x386b, 0x703d, 0x3a68, 0x703e, 0x703f, 0x3e69,
02000 0x7040, 0x366c, 0x7041, 0x7042, 0x7043, 0x7044, 0x4835, 0x7045,
02001 0x7046, 0x7047, 0x4574, 0x7048, 0x7049, 0x704a, 0x773d, 0x704b,
02002 0x704c, 0x704d, 0x704e, 0x704f, 0x3a57, 0x7050, 0x7051, 0x7052,
02003 0x7053, 0x7054, 0x7055, 0x7056, 0x7058, 0x5325, 0x7057, 0x7059,
02004 0x753a, 0x4239, 0x7764, 0x7765, 0x7766, 0x7767, 0x7768, 0x4234,
02005 0x776a, 0x776b, 0x4273, 0x7470, 0x746f, 0x4269, 0x7761, 0x7762,
02006 0x3b46, 0x5964, 0x4a72, 0x4068, 0x7024, 0x3a5a, 0x472d, 0x442c,
02007 0x776c, 0x776d, 0x776e, 0x7770, 0x776f, 0x7771, 0x7774, 0x7773,
02008 0x7772, 0x7775, 0x7776, 0x6d69, 0x6d6a, 0x6d6b, 0x763c, 0x763d,
02009 0x763e, 0x3626, 0x583e, 0x3944, 0x583b, 0x5c31, 0x4a73, 0x7777,
02010 0x7778, 0x7779, 0x777b, 0x777a, 0x3147, 0x777c, 0x777d, 0x777e,
02011 0x466b, 0x6c34, 0x335d, 0x7633, 0x7634, 0x4164, 0x7635, 0x7636,
02012 0x7637, 0x7638, 0x7639, 0x763a, 0x4823, 0x763b, 0x417a, 0x3928,
02013 0x6d68, 0x396a, 0x595f, 0x2321, 0x2322, 0x2323, 0x2167, 0x2325,
02014 0x2326, 0x2327, 0x2328, 0x2329, 0x232a, 0x232b, 0x232c, 0x232d,
02015 0x232e, 0x232f, 0x2330, 0x2331, 0x2332, 0x2333, 0x2334, 0x2335,
02016 0x2336, 0x2337, 0x2338, 0x2339, 0x233a, 0x233b, 0x233c, 0x233d,
02017 0x233e, 0x233f, 0x2340, 0x2341, 0x2342, 0x2343, 0x2344, 0x2345,
02018 0x2346, 0x2347, 0x2348, 0x2349, 0x234a, 0x234b, 0x234c, 0x234d,
02019 0x234e, 0x234f, 0x2350, 0x2351, 0x2352, 0x2353, 0x2354, 0x2355,
02020 0x2356, 0x2357, 0x2358, 0x2359, 0x235a, 0x235b, 0x235c, 0x235d,
02021 0x235e, 0x235f, 0x2360, 0x2361, 0x2362, 0x2363, 0x2364, 0x2365,
02022 0x2366, 0x2367, 0x2368, 0x2369, 0x236a, 0x236b, 0x236c, 0x236d,
02023 0x236e, 0x236f, 0x2370, 0x2371, 0x2372, 0x2373, 0x2374, 0x2375,
02024 0x2376, 0x2377, 0x2378, 0x2379, 0x237a, 0x237b, 0x237c, 0x237d,
```

```
02025     0x212b, 0x2169, 0x216a, 0x237e, 0x2324,
02026 };
02027
02028 static const Summary16 gb2312_uni2indx_page00[70] = {
02029     /* 0x0000 */
02030     { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 },
02031     { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 },
02032     { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0190 }, { 3, 0x0003 },
02033     { 5, 0x0000 }, { 5, 0x0080 }, { 6, 0x3703 }, { 13, 0x168c },
02034     /* 0x0100 */
02035     { 19, 0x0002 }, { 20, 0x0808 }, { 22, 0x0800 }, { 23, 0x0000 },
02036     { 23, 0x2000 }, { 24, 0x0000 }, { 24, 0x0800 }, { 25, 0x0000 },
02037     { 25, 0x0000 }, { 25, 0x0000 }, { 25, 0x0000 }, { 25, 0x0000 },
02038     { 25, 0x4000 }, { 26, 0x1555 }, { 33, 0x0000 }, { 33, 0x0000 },
02039     /* 0x0200 */
02040     { 33, 0x0000 }, { 33, 0x0000 }, { 33, 0x0000 }, { 33, 0x0000 },
02041     { 33, 0x0000 }, { 33, 0x0000 }, { 33, 0x0000 }, { 33, 0x0000 },
02042     { 33, 0x0000 }, { 33, 0x0000 }, { 33, 0x0000 }, { 33, 0x0000 },
02043     { 33, 0x0280 }, { 35, 0x0000 }, { 35, 0x0000 }, { 35, 0x0000 },
02044     /* 0x0300 */
02045     { 35, 0x0000 }, { 35, 0x0000 }, { 35, 0x0000 }, { 35, 0x0000 },
02046     { 35, 0x0000 }, { 35, 0x0000 }, { 35, 0x0000 }, { 35, 0x0000 },
02047     { 35, 0x0000 }, { 35, 0xffff }, { 50, 0x03fb }, { 59, 0xffff },
02048     { 74, 0x03fb }, { 83, 0x0000 }, { 83, 0x0000 }, { 83, 0x0000 },
02049     /* 0x0400 */
02050     { 83, 0x0002 }, { 84, 0xffff }, { 100, 0xffff }, { 116, 0xffff },
02051     { 132, 0xffff }, { 148, 0x0002 },
02052 };
02053 static const Summary16 gb2312_uni2indx_page20[101] = {
02054     /* 0x2000 */
02055     { 149, 0x0000 }, { 149, 0x3360 }, { 155, 0x0040 }, { 156, 0x080d },
02056     { 160, 0x0000 }, { 160, 0x0000 }, { 160, 0x0000 }, { 160, 0x0000 },
02057     { 160, 0x0000 }, { 160, 0x0000 }, { 160, 0x0000 }, { 160, 0x0000 },
02058     { 160, 0x0000 }, { 160, 0x0000 }, { 160, 0x0000 }, { 160, 0x0000 },
02059     /* 0x2100 */
02060     { 160, 0x0008 }, { 161, 0x0040 }, { 162, 0x0000 }, { 162, 0x0000 },
02061     { 162, 0x0000 }, { 162, 0x0000 }, { 162, 0x0fff }, { 174, 0x0000 },
02062     { 174, 0x0000 }, { 174, 0x000f }, { 178, 0x0000 }, { 178, 0x0000 },
02063     { 178, 0x0000 }, { 178, 0x0000 }, { 178, 0x0000 }, { 178, 0x0000 },
02064     /* 0x2200 */
02065     { 178, 0x8100 }, { 180, 0x6402 }, { 184, 0x4fa1 }, { 192, 0x20f0 },
02066     { 197, 0x1100 }, { 199, 0x0000 }, { 199, 0xc033 }, { 205, 0x0000 },
02067     { 205, 0x0000 }, { 205, 0x0200 }, { 206, 0x0020 }, { 207, 0x0000 },
02068     { 207, 0x0000 }, { 207, 0x0000 }, { 207, 0x0000 }, { 207, 0x0000 },
02069     /* 0x2300 */
02070     { 207, 0x0000 }, { 207, 0x0004 }, { 208, 0x0000 }, { 208, 0x0000 },
02071     { 208, 0x0000 }, { 208, 0x0000 }, { 208, 0x0000 }, { 208, 0x0000 },
02072     { 208, 0x0000 }, { 208, 0x0000 }, { 208, 0x0000 }, { 208, 0x0000 },
02073     { 208, 0x0000 }, { 208, 0x0000 }, { 208, 0x0000 }, { 208, 0x0000 },
02074     /* 0x2400 */
02075     { 208, 0x0000 }, { 208, 0x0000 }, { 208, 0x0000 }, { 208, 0x0000 },
02076     { 208, 0x0000 }, { 208, 0x0000 }, { 208, 0x03ff }, { 218, 0xffff0 },
02077     { 230, 0xffff }, { 246, 0x0fff }, { 258, 0x0000 }, { 258, 0x0000 },
02078     { 258, 0x0000 }, { 258, 0x0000 }, { 258, 0x0000 }, { 258, 0x0000 },
02079     /* 0x2500 */
02080     { 258, 0xffff }, { 274, 0xffff }, { 290, 0xffff }, { 306, 0xffff },
02081     { 322, 0x0fff }, { 334, 0x0000 }, { 334, 0x0000 }, { 334, 0x0000 },
02082     { 334, 0x0000 }, { 334, 0x0000 }, { 334, 0x0003 }, { 336, 0x000c },
02083     { 338, 0xc8c0 }, { 343, 0x0000 }, { 343, 0x0000 }, { 343, 0x0000 },
02084     /* 0x2600 */
02085     { 343, 0x0060 }, { 345, 0x0000 }, { 345, 0x0000 }, { 345, 0x0000 },
02086     { 345, 0x0005 },
02087 };
02088 static const Summary16 gb2312_uni2indx_page30[35] = {
02089     /* 0x3000 */
02090     { 347, 0xff2f }, { 360, 0x00fb }, { 367, 0x0000 }, { 367, 0x0000 },
02091     { 367, 0xffff }, { 382, 0xffff }, { 398, 0xffff }, { 414, 0xffff },
02092     { 430, 0xffff }, { 446, 0x000f }, { 450, 0xffff }, { 465, 0xffff },
02093     { 481, 0xffff }, { 497, 0xffff }, { 513, 0xffff }, { 529, 0x087f },
02094     /* 0x3100 */
02095     { 537, 0xffe0 }, { 548, 0xffff }, { 564, 0x03ff }, { 574, 0x0000 },
02096     { 574, 0x0000 }, { 574, 0x0000 }, { 574, 0x0000 }, { 574, 0x0000 },
02097     { 574, 0x0000 }, { 574, 0x0000 }, { 574, 0x0000 }, { 574, 0x0000 },
02098     { 574, 0x0000 }, { 574, 0x0000 }, { 574, 0x0000 }, { 574, 0x0000 },
02099     /* 0x3200 */
02100     { 574, 0x0000 }, { 574, 0x0000 }, { 574, 0x03ff },
02101 };
02102 static const Summary16 gb2312_uni2indx_page4e[1263] = {
02103     /* 0x4e00 */
02104     { 584, 0x7f8b }, { 595, 0x7f7b }, { 608, 0x3db4 }, { 617, 0xef55 },
02105     { 628, 0xfba8 }, { 638, 0xf35d }, { 649, 0x0243 }, { 653, 0x400b },
02106     { 657, 0xfb40 }, { 665, 0x8d3e }, { 674, 0x7bf7 }, { 687, 0x8c2c },
02107     { 693, 0x6eff }, { 706, 0xe3fa }, { 717, 0x1d3a }, { 725, 0xa8ed },
02108     /* 0x4f00 */
02109     { 734, 0xe602 }, { 740, 0xcf83 }, { 749, 0x8cf5 }, { 758, 0x3555 },
02110     { 766, 0xe048 }, { 771, 0xffab }, { 784, 0x92b9 }, { 792, 0xd859 },
02111     { 800, 0xab18 }, { 807, 0x2892 }, { 812, 0xd7e9 }, { 823, 0x8020 },
```

```

02112 { 825, 0xc438 }, { 831, 0xf583 }, { 840, 0xe74a }, { 849, 0x450a },
02113 /* 0x5000 */
02114 { 854, 0xb000 }, { 857, 0x9714 }, { 864, 0x7762 }, { 873, 0x5400 },
02115 { 876, 0xd188 }, { 882, 0x1420 }, { 885, 0x1020 }, { 887, 0xc8c0 },
02116 { 892, 0x2121 }, { 896, 0x0000 }, { 896, 0x13a8 }, { 902, 0x0c04 },
02117 { 905, 0x8000 }, { 906, 0x0440 }, { 908, 0x70c0 }, { 913, 0x0828 },
02118 /* 0x5100 */
02119 { 916, 0x08c0 }, { 919, 0x0004 }, { 920, 0x0002 }, { 921, 0x8000 },
02120 { 922, 0x2b7b }, { 932, 0x1472 }, { 938, 0x7924 }, { 945, 0x3bfb },
02121 { 957, 0x3327 }, { 965, 0x1ae4 }, { 972, 0x9835 }, { 979, 0x38ef },
02122 { 989, 0x9ad1 }, { 997, 0x2802 }, { 1000, 0xa813 }, { 1006, 0xbf69 },
02123 /* 0x5200 */
02124 { 1017, 0x65cf }, { 1027, 0x2fc6 }, { 1036, 0x6b11 }, { 1043, 0xafc9 },
02125 { 1053, 0x340f }, { 1060, 0x5053 }, { 1066, 0x86a2 }, { 1072, 0xa004 },
02126 { 1075, 0x0106 }, { 1078, 0xe809 }, { 1084, 0x3f0f }, { 1094, 0xc00e },
02127 { 1099, 0x0a88 }, { 1103, 0x8145 }, { 1108, 0x0010 }, { 1109, 0xc601 },
02128 /* 0x5300 */
02129 { 1114, 0xa161 }, { 1120, 0x26e1 }, { 1127, 0x444b }, { 1133, 0xce00 },
02130 { 1138, 0xc7aa }, { 1147, 0xd4ee }, { 1157, 0xcadf }, { 1168, 0x85bb },
02131 { 1177, 0x3a74 }, { 1185, 0xa520 }, { 1190, 0x436c }, { 1197, 0x8840 },
02132 { 1200, 0x3f06 }, { 1208, 0x8bd2 }, { 1216, 0xff79 }, { 1229, 0x3bef },
02133 /* 0x5400 */
02134 { 1241, 0xf75a }, { 1252, 0xe8ef }, { 1263, 0xfbc9 }, { 1275, 0x5b36 },
02135 { 1284, 0x0d49 }, { 1290, 0x1bfd }, { 1301, 0x0154 }, { 1305, 0x39ee },
02136 { 1315, 0xd855 }, { 1323, 0x2e75 }, { 1332, 0xbfd8 }, { 1343, 0xa91a },
02137 { 1350, 0xf3d7 }, { 1362, 0xf6bf }, { 1375, 0x67e0 }, { 1383, 0xb40c },
02138 /* 0x5500 */
02139 { 1389, 0x82c2 }, { 1394, 0x0813 }, { 1398, 0xd49d }, { 1407, 0xd08b },
02140 { 1414, 0x065a }, { 1420, 0x1061 }, { 1424, 0x74f2 }, { 1433, 0x59e0 },
02141 { 1440, 0x8f9f }, { 1451, 0xb312 }, { 1458, 0x0080 }, { 1459, 0x6aaa },
02142 { 1467, 0x3230 }, { 1472, 0xb05e }, { 1480, 0x9d7a }, { 1490, 0x60ac },
02143 /* 0x5600 */
02144 { 1496, 0xd303 }, { 1503, 0xc900 }, { 1507, 0x3098 }, { 1512, 0x8a56 },
02145 { 1519, 0x7000 }, { 1522, 0x1390 }, { 1527, 0x1f14 }, { 1534, 0x1842 },
02146 { 1538, 0xc060 }, { 1542, 0x0008 }, { 1543, 0x8008 }, { 1545, 0x1080 },
02147 { 1547, 0x0400 }, { 1548, 0xec90 }, { 1555, 0x2817 }, { 1561, 0xe633 },
02148 /* 0x5700 */
02149 { 1570, 0x0758 }, { 1576, 0x9000 }, { 1578, 0xf708 }, { 1586, 0x4e09 },
02150 { 1592, 0xf485 }, { 1600, 0xfc83 }, { 1609, 0xaf53 }, { 1619, 0x18c8 },
02151 { 1624, 0x187c }, { 1631, 0x080c }, { 1634, 0x6adf }, { 1645, 0x0114 },
02152 { 1648, 0xc80c }, { 1653, 0xa734 }, { 1661, 0xa011 }, { 1665, 0x2710 },
02153 /* 0x5800 */
02154 { 1670, 0x28c5 }, { 1676, 0x4222 }, { 1680, 0x0413 }, { 1684, 0x0021 },
02155 { 1686, 0x3010 }, { 1689, 0x4112 }, { 1693, 0x1820 }, { 1696, 0x4000 },
02156 { 1697, 0x022b }, { 1702, 0xc60c }, { 1708, 0x0300 }, { 1710, 0x1000 },
02157 { 1711, 0x0022 }, { 1713, 0x0022 }, { 1715, 0x5810 }, { 1719, 0x0249 },
02158 /* 0x5900 */
02159 { 1723, 0xa094 }, { 1728, 0x9670 }, { 1735, 0xeeb0 }, { 1744, 0x1792 },
02160 { 1751, 0xcb96 }, { 1760, 0x05f2 }, { 1767, 0x0025 }, { 1770, 0x2358 },
02161 { 1776, 0x25de }, { 1785, 0x42cc }, { 1791, 0xcf38 }, { 1800, 0x4a04 },
02162 { 1804, 0x0c40 }, { 1807, 0x359f }, { 1817, 0x1128 }, { 1821, 0x8a00 },
02163 /* 0x5a00 */
02164 { 1824, 0x13fa }, { 1833, 0x910a }, { 1838, 0x0229 }, { 1842, 0x1056 },
02165 { 1847, 0x0641 }, { 1851, 0x0420 }, { 1853, 0x0484 }, { 1856, 0x84f0 },
02166 { 1862, 0x0000 }, { 1862, 0x0c04 }, { 1865, 0x0400 }, { 1866, 0x412c },
02167 { 1871, 0x1206 }, { 1875, 0x1154 }, { 1880, 0x0a4b }, { 1886, 0x0002 },
02168 /* 0x5b00 */
02169 { 1887, 0x0200 }, { 1888, 0x00c0 }, { 1890, 0x0000 }, { 1890, 0x0094 },
02170 { 1893, 0x0001 }, { 1894, 0xbfb9 }, { 1907, 0x167c }, { 1915, 0x242b },
02171 { 1921, 0x9bbb }, { 1932, 0x7fa8 }, { 1942, 0x0c7f }, { 1951, 0xe379 },
02172 { 1961, 0x10f4 }, { 1967, 0xe00d }, { 1973, 0x4132 }, { 1978, 0x9f01 },
02173 /* 0x5c00 */
02174 { 1985, 0x8652 }, { 1991, 0x3572 }, { 1999, 0x10b4 }, { 2004, 0xff12 },
02175 { 2014, 0xcf27 }, { 2024, 0x4223 }, { 2029, 0xc06b }, { 2036, 0x8602 },
02176 { 2040, 0x3106 }, { 2045, 0x1fd3 }, { 2055, 0x3a0c }, { 2061, 0xa1aa },
02177 { 2068, 0x0812 }, { 2071, 0x0204 }, { 2073, 0x2572 }, { 2080, 0x0801 },
02178 /* 0x5d00 */
02179 { 2082, 0x40cc }, { 2087, 0x4850 }, { 2091, 0x62d0 }, { 2097, 0x6010 },
02180 { 2100, 0x1c80 }, { 2104, 0x2900 }, { 2107, 0x9a00 }, { 2111, 0x0010 },
02181 { 2112, 0x0004 }, { 2113, 0x2200 }, { 2115, 0x0000 }, { 2115, 0x0080 },
02182 { 2116, 0x2020 }, { 2118, 0x6800 }, { 2121, 0xcbe6 }, { 2131, 0x609e },
02183 /* 0x5e00 */
02184 { 2138, 0x916e }, { 2146, 0x3f73 }, { 2157, 0x60c0 }, { 2161, 0x3982 },
02185 { 2167, 0x1034 }, { 2171, 0x4830 }, { 2175, 0x0006 }, { 2177, 0xbd5c },
02186 { 2187, 0x8cd1 }, { 2194, 0xd6fb }, { 2206, 0x20e1 }, { 2211, 0x43e8 },
02187 { 2218, 0x0600 }, { 2220, 0x084e }, { 2225, 0x0500 }, { 2227, 0xc4d0 },
02188 /* 0x5f00 */
02189 { 2233, 0x8d1f }, { 2242, 0x89aa }, { 2249, 0xa6e1 }, { 2257, 0x1602 },
02190 { 2261, 0x0001 }, { 2262, 0x21ed }, { 2270, 0x3656 }, { 2278, 0x1a8b },
02191 { 2285, 0x1fb7 }, { 2296, 0x13a5 }, { 2303, 0x6502 }, { 2308, 0x30a0 },
02192 { 2312, 0xb278 }, { 2320, 0x23c7 }, { 2328, 0x6c93 }, { 2336, 0xe922 },
02193 /* 0x6000 */
02194 { 2343, 0xe47f }, { 2354, 0x3a74 }, { 2362, 0x8fe3 }, { 2372, 0x9820 },
02195 { 2376, 0x280e }, { 2381, 0x2625 }, { 2387, 0xbf9c }, { 2398, 0xbf49 },
02196 { 2408, 0x3218 }, { 2413, 0xac54 }, { 2420, 0xb949 }, { 2428, 0x1916 },
02197 { 2434, 0x0c60 }, { 2438, 0xb522 }, { 2445, 0xfbc1 }, { 2455, 0x0659 },
02198 /* 0x6100 */

```



```
02199 { 2461, 0xe343 }, { 2469, 0x8420 }, { 2472, 0x08d9 }, { 2478, 0x8000 },
02200 { 2479, 0x5500 }, { 2483, 0x2022 }, { 2486, 0x0184 }, { 2489, 0x00a1 },
02201 { 2492, 0x4800 }, { 2494, 0x2010 }, { 2496, 0x1380 }, { 2500, 0x4080 },
02202 { 2502, 0x0d04 }, { 2506, 0x0016 }, { 2509, 0x0040 }, { 2510, 0x8020 },
02203 /* 0x6200 */
02204 { 2512, 0xfd40 }, { 2520, 0x8de7 }, { 2530, 0x5436 }, { 2537, 0xe098 },
02205 { 2543, 0x7b8b }, { 2553, 0x091e }, { 2559, 0xfec8 }, { 2569, 0xd249 },
02206 { 2576, 0x0611 }, { 2580, 0x8dee }, { 2590, 0x1937 }, { 2598, 0xba22 },
02207 { 2605, 0x77f4 }, { 2616, 0x9fdd }, { 2628, 0xf3ec }, { 2639, 0xf0da },
02208 /* 0x6300 */
02209 { 2648, 0x4386 }, { 2654, 0xec42 }, { 2661, 0x8d3f }, { 2671, 0x2604 },
02210 { 2675, 0xfa6c }, { 2685, 0xc021 }, { 2689, 0x628e }, { 2696, 0x0cc2 },
02211 { 2701, 0xd785 }, { 2710, 0x0145 }, { 2714, 0x77ad }, { 2725, 0x5599 },
02212 { 2733, 0xe250 }, { 2739, 0x4045 }, { 2743, 0x260b }, { 2749, 0xa154 },
02213 /* 0x6400 */
02214 { 2755, 0x9827 }, { 2762, 0x5819 }, { 2768, 0x3443 }, { 2774, 0xa410 },
02215 { 2778, 0x05f2 }, { 2785, 0x4114 }, { 2789, 0x2280 }, { 2792, 0x0700 },
02216 { 2795, 0x00b4 }, { 2799, 0x4266 }, { 2805, 0x7210 }, { 2810, 0x15a1 },
02217 { 2816, 0x6025 }, { 2821, 0x4185 }, { 2826, 0x0054 }, { 2829, 0x0000 },
02218 /* 0x6500 */
02219 { 2829, 0x0201 }, { 2831, 0x0104 }, { 2833, 0xc820 }, { 2837, 0xcb70 },
02220 { 2845, 0x9320 }, { 2850, 0x6a62 }, { 2857, 0x184c }, { 2862, 0x0095 },
02221 { 2866, 0x1880 }, { 2869, 0x9a8b }, { 2877, 0xaab2 }, { 2885, 0x3201 },
02222 { 2889, 0xd87a }, { 2898, 0x00c4 }, { 2901, 0xf3e5 }, { 2912, 0x04c3 },
02223 /* 0x6600 */
02224 { 2917, 0xd44d }, { 2925, 0xa238 }, { 2931, 0xala1 }, { 2937, 0x5072 },
02225 { 2943, 0x980a }, { 2948, 0x84fc }, { 2956, 0xc152 }, { 2962, 0x44d1 },
02226 { 2968, 0x1094 }, { 2972, 0x20c2 }, { 2976, 0x4180 }, { 2979, 0x4210 },
02227 { 2982, 0x0000 }, { 2982, 0x3a00 }, { 2986, 0x0240 }, { 2988, 0xd29d },
02228 /* 0x6700 */
02229 { 2997, 0x2f01 }, { 3003, 0xa8b1 }, { 3010, 0xbd40 }, { 3017, 0x2432 },
02230 { 3022, 0xd34d }, { 3031, 0xd04b }, { 3038, 0xa723 }, { 3046, 0xd0ad },
02231 { 3054, 0x0a92 }, { 3059, 0x75a1 }, { 3067, 0xadac }, { 3076, 0x01e9 },
02232 { 3082, 0x801a }, { 3086, 0x771f }, { 3097, 0x9225 }, { 3103, 0xa01b },
02233 /* 0x6800 */
02234 { 3109, 0xdfa1 }, { 3119, 0x20ca }, { 3124, 0x0602 }, { 3127, 0x738c },
02235 { 3135, 0x577f }, { 3147, 0x003b }, { 3152, 0x0bff }, { 3163, 0x00d0 },
02236 { 3166, 0x806a }, { 3171, 0x0088 }, { 3173, 0xalc4 }, { 3179, 0x0029 },
02237 { 3182, 0x2a05 }, { 3187, 0x0524 }, { 3191, 0x4009 }, { 3194, 0x1623 },
02238 /* 0x6900 */
02239 { 3200, 0x6822 }, { 3205, 0x8005 }, { 3208, 0x2011 }, { 3211, 0xa211 },
02240 { 3216, 0x0004 }, { 3217, 0x6490 }, { 3222, 0x4849 }, { 3227, 0x1382 },
02241 { 3232, 0x23d5 }, { 3240, 0x1930 }, { 3245, 0x2980 }, { 3249, 0x0892 },
02242 { 3253, 0x5402 }, { 3257, 0x8811 }, { 3261, 0x2001 }, { 3263, 0xa004 },
02243 /* 0x6a00 */
02244 { 3266, 0x0400 }, { 3267, 0x8180 }, { 3270, 0x8502 }, { 3274, 0x6022 },
02245 { 3278, 0x0090 }, { 3280, 0x0b01 }, { 3284, 0x0022 }, { 3286, 0x1202 },
02246 { 3289, 0x4011 }, { 3292, 0x0083 }, { 3295, 0x1a01 }, { 3299, 0x0000 },
02247 { 3299, 0x0000 }, { 3299, 0x0000 }, { 3299, 0x0000 }, { 3299, 0x0000 },
02248 /* 0x6b00 */
02249 { 3299, 0x0000 }, { 3299, 0x0000 }, { 3299, 0x009f }, { 3305, 0x4684 },
02250 { 3310, 0x12c8 }, { 3315, 0x0200 }, { 3316, 0x04fc }, { 3323, 0x1a00 },
02251 { 3326, 0x2ede }, { 3336, 0x0c4c }, { 3341, 0x0402 }, { 3343, 0x80b8 },
02252 { 3348, 0xa826 }, { 3354, 0x0afc }, { 3362, 0x8c02 }, { 3366, 0x2228 },
02253 /* 0x6c00 */
02254 { 3370, 0xa0e0 }, { 3375, 0x8f7b }, { 3386, 0xc7d6 }, { 3396, 0x2135 },
02255 { 3402, 0x06c7 }, { 3409, 0xf8b1 }, { 3418, 0x0713 }, { 3424, 0x6255 },
02256 { 3431, 0x936e }, { 3440, 0x8a19 }, { 3446, 0x6efa }, { 3457, 0xfb0e },
02257 { 3467, 0x1630 }, { 3472, 0x48f9 }, { 3480, 0xcd2f }, { 3490, 0x7deb },
02258 /* 0x6d00 */
02259 { 3502, 0x5892 }, { 3508, 0x4e84 }, { 3514, 0x4ca0 }, { 3519, 0x7a2e },
02260 { 3528, 0xedea }, { 3539, 0x561e }, { 3547, 0xc649 }, { 3554, 0x1190 },
02261 { 3558, 0x5324 }, { 3564, 0xe83a }, { 3572, 0xcfdb }, { 3584, 0x8124 },
02262 { 3588, 0x18f1 }, { 3595, 0x6342 }, { 3601, 0x5853 }, { 3608, 0x1a8a },
02263 /* 0x6e00 */
02264 { 3614, 0x7420 }, { 3619, 0x24d3 }, { 3626, 0xaa3b }, { 3635, 0x0514 },
02265 { 3639, 0x6018 }, { 3643, 0x8958 }, { 3649, 0x4800 }, { 3651, 0xc000 },
02266 { 3653, 0x8268 }, { 3658, 0x9101 }, { 3662, 0x84a4 }, { 3667, 0x2cd6 },
02267 { 3675, 0x8886 }, { 3680, 0xc4ba }, { 3688, 0x0377 }, { 3696, 0x0210 },
02268 /* 0x6f00 */
02269 { 3698, 0x8244 }, { 3702, 0x0038 }, { 3705, 0xae11 }, { 3712, 0x404a },
02270 { 3716, 0x28c0 }, { 3720, 0x5100 }, { 3723, 0x6044 }, { 3727, 0x1514 },
02271 { 3732, 0x7310 }, { 3738, 0x1000 }, { 3739, 0x0082 }, { 3741, 0x0248 },
02272 { 3744, 0x0205 }, { 3747, 0x4006 }, { 3750, 0xc003 }, { 3754, 0x0000 },
02273 /* 0x7000 */
02274 { 3754, 0x0000 }, { 3754, 0x0c02 }, { 3757, 0x0008 }, { 3758, 0x0220 },
02275 { 3760, 0x9000 }, { 3762, 0x4000 }, { 3763, 0xb800 }, { 3767, 0xd161 },
02276 { 3774, 0x4621 }, { 3779, 0x3274 }, { 3786, 0xf800 }, { 3791, 0x3b8a },
02277 { 3799, 0x050f }, { 3805, 0x8b00 }, { 3809, 0xbbd0 }, { 3818, 0x2280 },
02278 /* 0x7100 */
02279 { 3821, 0x0600 }, { 3823, 0x0769 }, { 3830, 0x8040 }, { 3832, 0x0043 },
02280 { 3835, 0x5420 }, { 3839, 0x5000 }, { 3841, 0x41d0 }, { 3846, 0x250c },
02281 { 3851, 0x8410 }, { 3854, 0x8310 }, { 3858, 0x1101 }, { 3861, 0x0228 },
02282 { 3864, 0x4008 }, { 3866, 0x0030 }, { 3868, 0x40a1 }, { 3872, 0x0200 },
02283 /* 0x7200 */
02284 { 3873, 0x0040 }, { 3874, 0x2000 }, { 3875, 0x1500 }, { 3878, 0xab3e },
02285 { 3888, 0x3180 }, { 3892, 0xaa44 }, { 3898, 0xc2c6 }, { 3905, 0xc624 },
```

```

02286 { 3911, 0xac13 }, { 3918, 0x8004 }, { 3920, 0xb000 }, { 3923, 0x03d1 },
02287 { 3929, 0x611e }, { 3936, 0x4285 }, { 3941, 0xf303 }, { 3949, 0x1d9f },
02288 /* 0x7300 */
02289 { 3959, 0x440a }, { 3963, 0x78e8 }, { 3971, 0x5e26 }, { 3979, 0xc392 },
02290 { 3986, 0x2000 }, { 3987, 0x0085 }, { 3990, 0xb001 }, { 3994, 0x4000 },
02291 { 3995, 0x4a90 }, { 4000, 0x8842 }, { 4004, 0xca04 }, { 4009, 0x0c8d },
02292 { 4015, 0xa705 }, { 4022, 0x4203 }, { 4026, 0x22a1 }, { 4031, 0x0004 },
02293 /* 0x7400 */
02294 { 4032, 0x8668 }, { 4038, 0x0c01 }, { 4041, 0x5564 }, { 4048, 0x1079 },
02295 { 4054, 0x0002 }, { 4055, 0xdea0 }, { 4063, 0x2000 }, { 4064, 0x40c1 },
02296 { 4068, 0x488b }, { 4074, 0x5001 }, { 4077, 0x0380 }, { 4080, 0x0400 },
02297 { 4081, 0x0000 }, { 4081, 0x5004 }, { 4084, 0xc05d }, { 4091, 0x80d0 },
02298 /* 0x7500 */
02299 { 4095, 0xa010 }, { 4098, 0x970a }, { 4105, 0xbb20 }, { 4112, 0x4daf },
02300 { 4122, 0xd921 }, { 4129, 0x1e10 }, { 4134, 0x0460 }, { 4137, 0x8314 },
02301 { 4142, 0x8848 }, { 4146, 0xa6d6 }, { 4155, 0xd83b }, { 4164, 0x733f },
02302 { 4175, 0x27bc }, { 4184, 0x4974 }, { 4191, 0x0ddc }, { 4199, 0x9213 },
02303 /* 0x7600 */
02304 { 4205, 0x142b }, { 4211, 0x8ba1 }, { 4218, 0x2e75 }, { 4227, 0xd139 },
02305 { 4235, 0x3009 }, { 4239, 0x5050 }, { 4243, 0x8808 }, { 4246, 0x6900 },
02306 { 4250, 0x49d4 }, { 4257, 0x024a }, { 4261, 0x4010 }, { 4263, 0x8016 },
02307 { 4267, 0xe564 }, { 4275, 0x89d7 }, { 4284, 0xc020 }, { 4287, 0x5316 },
02308 /* 0x7700 */
02309 { 4294, 0x2b92 }, { 4301, 0x8600 }, { 4304, 0xa345 }, { 4311, 0x15e0 },
02310 { 4317, 0x008b }, { 4321, 0x0c03 }, { 4325, 0x196e }, { 4333, 0xe200 },
02311 { 4337, 0x7031 }, { 4343, 0x8006 }, { 4346, 0x16a5 }, { 4353, 0xa829 },
02312 { 4359, 0x2000 }, { 4360, 0x1880 }, { 4363, 0x7aac }, { 4372, 0xe148 },
02313 /* 0x7800 */
02314 { 4378, 0x3207 }, { 4384, 0xb5d6 }, { 4394, 0x32e8 }, { 4401, 0x5f91 },
02315 { 4410, 0x50a1 }, { 4415, 0x20e5 }, { 4421, 0x7c00 }, { 4426, 0x1080 },
02316 { 4428, 0x7280 }, { 4433, 0x9d8a }, { 4441, 0x00aa }, { 4445, 0x421f },
02317 { 4452, 0x0e22 }, { 4457, 0x0231 }, { 4461, 0x1100 }, { 4463, 0x0494 },
02318 /* 0x7900 */
02319 { 4467, 0x0022 }, { 4469, 0x4008 }, { 4471, 0x0010 }, { 4472, 0x5c10 },
02320 { 4477, 0x0343 }, { 4482, 0xfcc8 }, { 4491, 0xa1a5 }, { 4498, 0x0580 },
02321 { 4501, 0x8433 }, { 4507, 0x0400 }, { 4508, 0x0080 }, { 4509, 0x6e08 },
02322 { 4515, 0x2a4b }, { 4522, 0x8126 }, { 4527, 0xaad8 }, { 4535, 0x2901 },
02323 /* 0x7a00 */
02324 { 4539, 0x684d }, { 4546, 0x4490 }, { 4550, 0x0009 }, { 4552, 0xba88 },
02325 { 4559, 0x0040 }, { 4560, 0x0082 }, { 4562, 0x0000 }, { 4562, 0x87d1 },
02326 { 4570, 0x215b }, { 4577, 0xb1e6 }, { 4586, 0x3161 }, { 4592, 0x8008 },
02327 { 4594, 0x0800 }, { 4595, 0xc240 }, { 4599, 0xa069 }, { 4605, 0xa600 },
02328 /* 0x7b00 */
02329 { 4609, 0x8d58 }, { 4616, 0x4a32 }, { 4622, 0x5d71 }, { 4631, 0x550a },
02330 { 4637, 0x9aa0 }, { 4643, 0x2d57 }, { 4652, 0x4005 }, { 4655, 0x4aa6 },
02331 { 4662, 0x2021 }, { 4665, 0x30b1 }, { 4671, 0x3fc6 }, { 4681, 0x0112 },
02332 { 4684, 0x10c2 }, { 4688, 0x260a }, { 4693, 0x4462 }, { 4698, 0x5082 },
02333 /* 0x7c00 */
02334 { 4702, 0x9880 }, { 4706, 0x8040 }, { 4708, 0x04c0 }, { 4711, 0x8100 },
02335 { 4713, 0x2003 }, { 4716, 0x0000 }, { 4716, 0x0000 }, { 4716, 0x3818 },
02336 { 4721, 0x0200 }, { 4722, 0xf1a6 }, { 4731, 0x4434 }, { 4736, 0x720e },
02337 { 4743, 0x35a2 }, { 4750, 0x92e0 }, { 4756, 0x8101 }, { 4759, 0x0900 },
02338 /* 0x7d00 */
02339 { 4761, 0x0400 }, { 4762, 0x0000 }, { 4762, 0x8885 }, { 4767, 0x0000 },
02340 { 4767, 0x0000 }, { 4767, 0x0000 }, { 4767, 0x4000 }, { 4768, 0x0080 },
02341 { 4769, 0x0000 }, { 4769, 0x0000 }, { 4769, 0x4040 }, { 4771, 0x0000 },
02342 { 4771, 0x0000 }, { 4771, 0x0000 }, { 4771, 0x0000 }, { 4771, 0x0000 },
02343 /* 0x7e00 */
02344 { 4771, 0x0000 }, { 4771, 0x0000 }, { 4771, 0x0000 }, { 4771, 0x0800 },
02345 { 4772, 0x0082 }, { 4774, 0x0000 }, { 4774, 0x0000 }, { 4774, 0x0000 },
02346 { 4774, 0x0004 }, { 4775, 0x8800 }, { 4777, 0xbfff }, { 4792, 0xe7ef },
02347 { 4805, 0xffff }, { 4821, 0xffbf }, { 4836, 0xefef }, { 4850, 0xfdf },
02348 /* 0x7f00 */
02349 { 4865, 0xfbff }, { 4880, 0xbffe }, { 4894, 0xffff }, { 4910, 0x057f },
02350 { 4919, 0x0034 }, { 4922, 0x85b3 }, { 4930, 0x706 }, { 4936, 0x4216 },
02351 { 4941, 0x5402 }, { 4945, 0xe410 }, { 4950, 0x8092 }, { 4954, 0xb305 },
02352 { 4961, 0x5422 }, { 4966, 0x8130 }, { 4970, 0x4263 }, { 4976, 0x180b },
02353 /* 0x8000 */
02354 { 4981, 0x387b }, { 4990, 0x13f5 }, { 4999, 0x07e5 }, { 5007, 0xa9ea },
02355 { 5016, 0x3c4c }, { 5023, 0x0514 }, { 5027, 0x0600 }, { 5029, 0x8002 },
02356 { 5031, 0x1ad9 }, { 5039, 0xbd48 }, { 5047, 0xee37 }, { 5058, 0xf496 },
02357 { 5067, 0x705f }, { 5076, 0x7ec0 }, { 5084, 0xbfb2 }, { 5095, 0x355f },
02358 /* 0x8100 */
02359 { 5105, 0xe644 }, { 5112, 0x455f }, { 5121, 0x9000 }, { 5123, 0x4146 },
02360 { 5128, 0x1d40 }, { 5133, 0x063b }, { 5140, 0x62a1 }, { 5146, 0xfe13 },
02361 { 5156, 0x8505 }, { 5161, 0x3902 }, { 5166, 0x0548 }, { 5170, 0x0c08 },
02362 { 5173, 0x144f }, { 5180, 0x0000 }, { 5180, 0x3488 }, { 5185, 0x5818 },
02363 /* 0x8200 */
02364 { 5190, 0x3077 }, { 5198, 0xd815 }, { 5205, 0xbd0e }, { 5214, 0x4bfb },
02365 { 5225, 0x8a90 }, { 5230, 0x8500 }, { 5233, 0xc100 }, { 5236, 0xe61d },
02366 { 5245, 0xed14 }, { 5253, 0xb386 }, { 5261, 0xff72 }, { 5273, 0x639b },
02367 { 5282, 0xfd92 }, { 5292, 0xd9be }, { 5303, 0x887b }, { 5311, 0xa92 },
02368 /* 0x8300 */
02369 { 5316, 0xd3fe }, { 5328, 0x1cb2 }, { 5335, 0xb980 }, { 5341, 0x177a },
02370 { 5350, 0x82c9 }, { 5356, 0xdc17 }, { 5365, 0xffff }, { 5380, 0x3980 },
02371 { 5385, 0x4260 }, { 5389, 0x590c }, { 5395, 0x0f01 }, { 5400, 0x37df },
02372 { 5412, 0x94a3 }, { 5419, 0xb150 }, { 5425, 0x0623 }, { 5430, 0x2307 },

```

```
02373  /* 0x8400 */
02374  { 5436, 0xf85a }, { 5445, 0x3102 }, { 5449, 0x01f0 }, { 5454, 0x3102 },
02375  { 5458, 0x0040 }, { 5459, 0x1e82 }, { 5465, 0x3a0a }, { 5471, 0x056a },
02376  { 5477, 0x5b84 }, { 5484, 0x1280 }, { 5487, 0x8002 }, { 5489, 0xa714 },
02377  { 5496, 0x2612 }, { 5501, 0xa04b }, { 5507, 0x1069 }, { 5512, 0x9001 },
02378  /* 0x8500 */
02379  { 5515, 0x1000 }, { 5516, 0x848a }, { 5521, 0x1802 }, { 5524, 0x3f80 },
02380  { 5531, 0x0708 }, { 5535, 0x4240 }, { 5538, 0x0110 }, { 5540, 0x4e14 },
02381  { 5546, 0x80b0 }, { 5550, 0x1800 }, { 5552, 0xc510 }, { 5557, 0x0281 },
02382  { 5560, 0x8202 }, { 5563, 0x1029 }, { 5567, 0x0210 }, { 5569, 0x8800 },
02383  /* 0x8600 */
02384  { 5571, 0x0020 }, { 5572, 0x0042 }, { 5574, 0x0280 }, { 5576, 0x1100 },
02385  { 5578, 0xe000 }, { 5581, 0x4413 }, { 5586, 0x5804 }, { 5590, 0xfe02 },
02386  { 5598, 0x3c07 }, { 5605, 0x3028 }, { 5609, 0x9798 }, { 5617, 0x0473 },
02387  { 5623, 0xcdc1 }, { 5632, 0xcb13 }, { 5640, 0x6210 }, { 5644, 0x431f },
02388  /* 0x8700 */
02389  { 5652, 0x278d }, { 5660, 0x55ac }, { 5668, 0x422e }, { 5674, 0xc892 },
02390  { 5680, 0x5380 }, { 5685, 0x0288 }, { 5688, 0x4039 }, { 5693, 0x7851 },
02391  { 5700, 0x292c }, { 5706, 0x8088 }, { 5709, 0xb900 }, { 5714, 0x2428 },
02392  { 5718, 0x0c41 }, { 5722, 0x080e }, { 5726, 0x4421 }, { 5730, 0x4200 },
02393  /* 0x8800 */
02394  { 5732, 0x0408 }, { 5734, 0x0868 }, { 5738, 0x0006 }, { 5740, 0x1204 },
02395  { 5743, 0x3031 }, { 5748, 0x0290 }, { 5751, 0x5b3e }, { 5761, 0xe085 },
02396  { 5767, 0x2936 }, { 5774, 0x1044 }, { 5777, 0x2814 }, { 5781, 0x1082 },
02397  { 5784, 0x4266 }, { 5790, 0x8334 }, { 5796, 0x013c }, { 5801, 0x531b },
02398  /* 0x8900 */
02399  { 5809, 0x0404 }, { 5811, 0x0e0d }, { 5817, 0x0c22 }, { 5821, 0x0051 },
02400  { 5824, 0x0012 }, { 5826, 0xc000 }, { 5828, 0x0040 }, { 5829, 0x8800 },
02401  { 5831, 0x004a }, { 5834, 0x0000 }, { 5834, 0x0000 }, { 5834, 0x0000 },
02402  { 5834, 0xdff6 }, { 5847, 0x5447 }, { 5854, 0x8868 }, { 5859, 0x0008 },
02403  /* 0x8a00 */
02404  { 5860, 0x0081 }, { 5862, 0x0000 }, { 5862, 0x0000 }, { 5862, 0x4000 },
02405  { 5863, 0x0100 }, { 5864, 0x0000 }, { 5864, 0x0000 }, { 5864, 0x0200 },
02406  { 5865, 0x0600 }, { 5867, 0x0008 }, { 5868, 0x0000 }, { 5868, 0x0000 },
02407  { 5868, 0x0000 }, { 5868, 0x0000 }, { 5868, 0x0000 }, { 5868, 0x0000 },
02408  /* 0x8b00 */
02409  { 5868, 0x0080 }, { 5869, 0x0000 }, { 5869, 0x0040 }, { 5870, 0x0000 },
02410  { 5870, 0x0000 }, { 5870, 0x0000 }, { 5870, 0x1040 }, { 5872, 0x0000 },
02411  { 5872, 0x0000 }, { 5872, 0x0000 }, { 5872, 0xefff }, { 5887, 0xf7fd },
02412  { 5901, 0xff7f }, { 5916, 0xfffe }, { 5931, 0xfbff }, { 5946, 0xffff },
02413  /* 0x8c00 */
02414  { 5962, 0xfdfc }, { 5977, 0xbfff }, { 5992, 0xffff }, { 6008, 0x00ff },
02415  { 6016, 0x12c2 }, { 6021, 0x0420 }, { 6023, 0x0c06 }, { 6027, 0x0708 },
02416  { 6031, 0x1624 }, { 6036, 0x0110 }, { 6038, 0x0000 }, { 6038, 0x0000 },
02417  { 6038, 0x0000 }, { 6038, 0x0000 }, { 6038, 0x0000 }, { 6038, 0x0000 },
02418  /* 0x8d00 */
02419  { 6038, 0x0000 }, { 6038, 0xe000 }, { 6041, 0xffff }, { 6056, 0xffff },
02420  { 6072, 0xffff }, { 6088, 0x7f79 }, { 6100, 0x28df }, { 6109, 0x00f9 },
02421  { 6115, 0x0c32 }, { 6120, 0x8012 }, { 6123, 0x0008 }, { 6124, 0xd53a },
02422  { 6133, 0xd858 }, { 6140, 0xecc2 }, { 6148, 0x9d18 }, { 6155, 0x2fa8 },
02423  /* 0x8e00 */
02424  { 6163, 0x9620 }, { 6168, 0xe010 }, { 6172, 0xd60c }, { 6179, 0x2622 },
02425  { 6184, 0x0f97 }, { 6193, 0x0206 }, { 6196, 0xb240 }, { 6201, 0x9055 },
02426  { 6207, 0x80a2 }, { 6211, 0x5011 }, { 6215, 0x9800 }, { 6218, 0x0404 },
02427  { 6220, 0x4000 }, { 6221, 0x0000 }, { 6221, 0x0000 }, { 6221, 0x0000 },
02428  /* 0x8f00 */
02429  { 6221, 0x0000 }, { 6221, 0x0000 }, { 6221, 0x0000 }, { 6221, 0x0000 },
02430  { 6221, 0x0000 }, { 6221, 0x0000 }, { 6221, 0xfb00 }, { 6230, 0xffff },
02431  { 6246, 0xeffe }, { 6260, 0xdffb }, { 6274, 0x0b08 }, { 6278, 0x6243 },
02432  { 6284, 0x41b6 }, { 6291, 0xfb3b }, { 6303, 0x6f74 }, { 6313, 0x2389 },
02433  /* 0x9000 */
02434  { 6319, 0xae7f }, { 6331, 0xecd7 }, { 6342, 0xe047 }, { 6349, 0x5960 },
02435  { 6355, 0xa096 }, { 6361, 0x098f }, { 6368, 0x612c }, { 6374, 0xa030 },
02436  { 6378, 0x090d }, { 6383, 0x2aaa }, { 6390, 0xd44e }, { 6398, 0x4f7b },
02437  { 6409, 0xc4b2 }, { 6416, 0x388b }, { 6423, 0xa9c6 }, { 6431, 0x6110 },
02438  /* 0x9100 */
02439  { 6435, 0x0014 }, { 6437, 0x4200 }, { 6439, 0x800c }, { 6442, 0x0202 },
02440  { 6444, 0xfe48 }, { 6453, 0x6485 }, { 6459, 0xd63e }, { 6469, 0xe3f7 },
02441  { 6481, 0x3aa0 }, { 6487, 0x0c07 }, { 6492, 0xe40c }, { 6498, 0x0430 },
02442  { 6501, 0xf680 }, { 6508, 0x1002 }, { 6510, 0x0000 }, { 6510, 0x0000 },
02443  /* 0x9200 */
02444  { 6510, 0x0000 }, { 6510, 0x0000 }, { 6510, 0x0000 }, { 6510, 0x0000 },
02445  { 6510, 0x0000 }, { 6510, 0x0000 }, { 6510, 0x0000 }, { 6510, 0x0010 },
02446  { 6511, 0x4000 }, { 6512, 0x0000 }, { 6512, 0x4000 }, { 6513, 0x0000 },
02447  { 6513, 0x0100 }, { 6514, 0x0000 }, { 6514, 0x0000 }, { 6514, 0x0000 },
02448  /* 0x9300 */
02449  { 6514, 0x0000 }, { 6514, 0x0000 }, { 6514, 0x0000 }, { 6514, 0x4000 },
02450  { 6515, 0x0000 }, { 6515, 0x0000 }, { 6515, 0x0400 }, { 6516, 0x0000 },
02451  { 6516, 0x8000 }, { 6517, 0x0000 }, { 6517, 0x0000 }, { 6517, 0x0000 },
02452  { 6517, 0x0400 }, { 6518, 0x0040 }, { 6519, 0x0000 }, { 6519, 0x0000 },
02453  /* 0x9400 */
02454  { 6519, 0x0000 }, { 6519, 0x0000 }, { 6519, 0x0000 }, { 6519, 0x4000 },
02455  { 6520, 0x0000 }, { 6520, 0x0000 }, { 6520, 0x0800 }, { 6521, 0x0000 },
02456  { 6521, 0xffe0 }, { 6532, 0xfebd }, { 6545, 0xffff }, { 6561, 0xffff },
02457  { 6577, 0x7f7f }, { 6591, 0xfb7e }, { 6604, 0xffbf }, { 6619, 0xf7ff },
02458  /* 0x9500 */
02459  { 6634, 0xffff }, { 6650, 0xffff }, { 6665, 0xff7e }, { 6679, 0xdff7 },
```

```

02460 { 6693, 0xf6f7 }, { 6706, 0xf6df }, { 6720, 0xbffe }, { 6734, 0x804f },
02461 { 6740, 0x0000 }, { 6740, 0x0000 }, { 6740, 0x0000 }, { 6740, 0x0000 },
02462 { 6740, 0x0000 }, { 6740, 0x0000 }, { 6740, 0xef00 }, { 6747, 0x7fff },
02463 /* 0x9600 */
02464 { 6762, 0xff7f }, { 6777, 0xb6f7 }, { 6789, 0x4406 }, { 6793, 0xb87e },
02465 { 6803, 0x3bf5 }, { 6814, 0x8831 }, { 6819, 0x1796 }, { 6827, 0x00f4 },
02466 { 6832, 0xa960 }, { 6838, 0x1391 }, { 6844, 0x0080 }, { 6845, 0x7249 },
02467 { 6852, 0xf2f3 }, { 6863, 0x0024 }, { 6865, 0x8701 }, { 6870, 0x42c8 },
02468 /* 0x9700 */
02469 { 6875, 0xe3d3 }, { 6885, 0x5048 }, { 6889, 0x2400 }, { 6891, 0x4305 },
02470 { 6896, 0x0000 }, { 6896, 0x4a4c }, { 6902, 0x0227 }, { 6907, 0x1058 },
02471 { 6911, 0x2820 }, { 6914, 0x0116 }, { 6918, 0xa809 }, { 6923, 0x0014 },
02472 { 6925, 0x0000 }, { 6925, 0x0000 }, { 6925, 0x3ec0 }, { 6932, 0x0068 },
02473 /* 0x9800 */
02474 { 6935, 0x0000 }, { 6935, 0x0000 }, { 6935, 0x0000 }, { 6935, 0x0000 },
02475 { 6935, 0x0000 }, { 6935, 0x0000 }, { 6935, 0x0000 }, { 6935, 0xffe0 },
02476 { 6946, 0xb7ff }, { 6960, 0xfddb }, { 6973, 0x00f7 }, { 6980, 0x0000 },
02477 { 6980, 0x4000 }, { 6981, 0xc72e }, { 6990, 0x0180 }, { 6992, 0x0000 },
02478 /* 0x9900 */
02479 { 6992, 0x2000 }, { 6993, 0x0001 }, { 6994, 0x4000 }, { 6995, 0x0000 },
02480 { 6995, 0x0000 }, { 6995, 0x0030 }, { 6997, 0xffa8 }, { 7008, 0xb4f7 },
02481 { 7019, 0xadf3 }, { 7030, 0x03ff }, { 7040, 0x0120 }, { 7042, 0x0000 },
02482 { 7042, 0x0000 }, { 7042, 0x0000 }, { 7042, 0x0000 }, { 7042, 0x0000 },
02483 /* 0x9a00 */
02484 { 7042, 0x0000 }, { 7042, 0x0000 }, { 7042, 0x0000 }, { 7042, 0x0000 },
02485 { 7042, 0x0000 }, { 7042, 0x0000 }, { 7042, 0xf000 }, { 7046, 0xffffb },
02486 { 7061, 0x9df7 }, { 7073, 0xfdcf }, { 7086, 0x01bf }, { 7094, 0x15c3 },
02487 { 7101, 0x1827 }, { 7107, 0x810a }, { 7111, 0xa842 }, { 7116, 0x0a00 },
02488 /* 0x9b00 */
02489 { 7118, 0x8108 }, { 7121, 0x8008 }, { 7123, 0x8008 }, { 7125, 0x1804 },
02490 { 7128, 0xa3be }, { 7138, 0x0012 }, { 7140, 0x0000 }, { 7140, 0x0000 },
02491 { 7140, 0x0000 }, { 7140, 0x0000 }, { 7140, 0x0000 }, { 7140, 0x0000 },
02492 { 7140, 0x0000 }, { 7140, 0x0000 }, { 7140, 0x0000 }, { 7140, 0x0000 },
02493 /* 0x9c00 */
02494 { 7140, 0x0000 }, { 7140, 0x0000 }, { 7140, 0x0000 }, { 7140, 0x0000 },
02495 { 7140, 0x0000 }, { 7140, 0x0000 }, { 7140, 0x0000 }, { 7140, 0x9000 },
02496 { 7142, 0x69e6 }, { 7151, 0xdc37 }, { 7161, 0x6bff }, { 7174, 0x3dff },
02497 { 7187, 0xfcf8 }, { 7198, 0xf3f9 }, { 7210, 0x0004 },
02498 };
02499 static const Summary16 gb2312_uni2indx_page9e[27] = {
02500 /* 0x9e00 */
02501 { 7211, 0x0000 }, { 7211, 0x8000 }, { 7212, 0xb6f6 }, { 7225, 0xe7ee },
02502 { 7237, 0xdffe }, { 7251, 0x5da2 }, { 7259, 0x3fd8 }, { 7269, 0xc00b },
02503 { 7274, 0x0984 }, { 7278, 0xa00c }, { 7282, 0x0040 }, { 7283, 0x6910 },
02504 { 7288, 0xe210 }, { 7293, 0xb912 }, { 7300, 0x86a5 }, { 7307, 0x5a00 },
02505 /* 0x9f00 */
02506 { 7311, 0x6800 }, { 7314, 0x0289 }, { 7318, 0x9005 }, { 7322, 0x6a80 },
02507 { 7327, 0x0010 }, { 7328, 0x0003 }, { 7330, 0x0000 }, { 7330, 0x8000 },
02508 { 7331, 0x1ff9 }, { 7342, 0x8e00 }, { 7346, 0x0001 },
02509 };
02510 static const Summary16 gb2312_uni2indx_pageff[15] = {
02511 /* 0xff00 */
02512 { 7347, 0xffff }, { 7362, 0xffff }, { 7378, 0xffff }, { 7394, 0xffff },
02513 { 7410, 0xffff }, { 7426, 0xffff }, { 7441, 0x0000 }, { 7441, 0x0000 },
02514 { 7441, 0x0000 }, { 7441, 0x0000 }, { 7441, 0x0000 }, { 7441, 0x0000 },
02515 { 7441, 0x0000 }, { 7441, 0x0000 }, { 7441, 0x002b },
02516 };
02517
02518 static int
02519 gb2312_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
02520 {
02521     (void)conv;
02522     if (n >= 2) {
02523         const Summary16 *summary = NULL;
02524         if (wc < 0x0460)
02525             summary = &gb2312_uni2indx_page00[(wc>>4)];
02526         else if (wc >= 0x2000 && wc < 0x2650)
02527             summary = &gb2312_uni2indx_page20[(wc>>4)-0x200];
02528         else if (wc >= 0x3000 && wc < 0x3230)
02529             summary = &gb2312_uni2indx_page30[(wc>>4)-0x300];
02530         else if (wc >= 0x4e00 && wc < 0x9cf0)
02531             summary = &gb2312_uni2indx_page4e[(wc>>4)-0x4e0];
02532         else if (wc >= 0x9e00 && wc < 0x9fb0)
02533             summary = &gb2312_uni2indx_page9e[(wc>>4)-0x9e0];
02534         else if (wc >= 0xff00 && wc < 0xffff)
02535             summary = &gb2312_uni2indx_pageff[(wc>>4)-0xff0];
02536         if (summary) {
02537             unsigned short used = summary->used;
02538             unsigned int i = wc & 0x0f;
02539             if (used & ((unsigned short) 1 << i)) {
02540                 unsigned short c;
02541                 /* Keep in 'used' only the bits 0..i-1. */
02542                 used &= ((unsigned short) 1 << i) - 1;
02543                 /* Add 'summary->indx' and the number of bits set in 'used'. */
02544                 used = (used & 0x5555) + ((used & 0xaaaa) >> 1);
02545                 used = (used & 0x3333) + ((used & 0xcccc) >> 2);
02546                 used = (used & 0x0f0f) + ((used & 0xf0f0) >> 4);

```

```

02547         used = (used & 0x00ff) + (used » 8);
02548         c = gb2312_2charset[summary->indx + used];
02549         r[0] = (c » 8); r[1] = (c & 0xff);
02550         return 2;
02551     }
02552 }
02553 return RET_ILSEQ;
02554 }
02555 return RET_TOOSMALL;
02556 }
02557 #endif /* NEED_TOMB */

```

12.267 georgian_academy.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/georgian_academy.h,v 1.3 2000/11/29 17:40:29 dawes Exp $ */
00002
00003 /*
00004  * GEORGIAN-ACADEMY
00005  */
00006
00007 static const unsigned short georgian_academy_2uni[32] = {
00008     /* 0x80 */
00009     0x0080, 0x0081, 0x201a, 0x0192, 0x201e, 0x2026, 0x2020, 0x2021,
00010     0x02c6, 0x2030, 0x0160, 0x2039, 0x0152, 0x008d, 0x008e, 0x008f,
00011     /* 0x90 */
00012     0x0090, 0x2018, 0x2019, 0x201c, 0x201d, 0x2022, 0x2013, 0x2014,
00013     0x02dc, 0x2122, 0x0161, 0x203a, 0x0153, 0x009d, 0x009e, 0x0178,
00014 };
00015
00016 static int
00017 georgian_academy_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00018 {
00019     unsigned char c = *s;
00020     if (c >= 0x80 && c < 0xa0)
00021         *pwc = (ucs4_t) georgian_academy_2uni[c-0x80];
00022     else if (c >= 0xc0 && c < 0xe7)
00023         *pwc = (ucs4_t) c + 0x1010;
00024     else
00025         *pwc = (ucs4_t) c;
00026     return 1;
00027 }
00028
00029 static const unsigned char georgian_academy_page00[32] = {
00030     0x80, 0x81, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00031     0x00, 0x00, 0x00, 0x00, 0x00, 0x8d, 0x8e, 0x8f, /* 0x88-0x8f */
00032     0x90, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00033     0x00, 0x00, 0x00, 0x00, 0x00, 0x9d, 0x9e, 0x00, /* 0x98-0x9f */
00034 };
00035 static const unsigned char georgian_academy_page01[72] = {
00036     0x00, 0x00, 0x8c, 0x9c, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00037     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00038     0x8a, 0x9a, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00039     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00040     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00041     0x9f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00042     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00043     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00044     0x00, 0x00, 0x83, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00045 };
00046 static const unsigned char georgian_academy_page02[32] = {
00047     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x88, 0x00, /* 0xc0-0xc7 */
00048     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00049     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00050     0x00, 0x00, 0x00, 0x00, 0x98, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00051 };
00052 static const unsigned char georgian_academy_page20[48] = {
00053     0x00, 0x00, 0x00, 0x96, 0x97, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00054     0x91, 0x92, 0x82, 0x00, 0x93, 0x94, 0x84, 0x00, /* 0x18-0x1f */
00055     0x86, 0x87, 0x95, 0x00, 0x00, 0x00, 0x85, 0x00, /* 0x20-0x27 */
00056     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00057     0x89, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00058     0x00, 0x8b, 0x9b, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00059 };
00060
00061 static int
00062 georgian_academy_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00063 {
00064     unsigned char c = 0;
00065     if (wc < 0x0080) {
00066         *r = wc;
00067         return 1;
00068     }
00069     else if (wc >= 0x0080 && wc < 0x00a0)
00070         c = georgian_academy_page00[wc-0x0080];
00071     else if ((wc >= 0x00a0 && wc < 0x00c0) || (wc >= 0x00e7 && wc < 0x0100))

```

```

00072     c = wc;
00073     else if (wc >= 0x0150 && wc < 0x0198)
00074         c = georgian_academy_page01[wc-0x0150];
00075     else if (wc >= 0x02c0 && wc < 0x02e0)
00076         c = georgian_academy_page02[wc-0x02c0];
00077     else if (wc >= 0x10d0 && wc < 0x10f7)
00078         c = wc-0x1010;
00079     else if (wc >= 0x2010 && wc < 0x2040)
00080         c = georgian_academy_page20[wc-0x2010];
00081     else if (wc == 0x2122)
00082         c = 0x99;
00083     if (c != 0) {
00084         *r = c;
00085         return 1;
00086     }
00087     return RET_ILSEQ;
00088 }

```

12.268 georgian_ps.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/georgian_ps.h,v 1.3 2000/11/29 17:40:29 dawes Exp $ */
00002
00003 /*
00004  * GEORGIAN-PS
00005  */
00006
00007 static const unsigned short georgian_ps_2uni_1[32] = {
00008     /* 0x80 */
00009     0x0080, 0x0081, 0x201a, 0x0192, 0x201e, 0x2026, 0x2020, 0x2021,
00010     0x02c6, 0x2030, 0x0160, 0x2039, 0x0152, 0x008d, 0x008e, 0x008f,
00011     /* 0x90 */
00012     0x0090, 0x2018, 0x2019, 0x201c, 0x201d, 0x2022, 0x2013, 0x2014,
00013     0x02dc, 0x2122, 0x0161, 0x203a, 0x0153, 0x009d, 0x009e, 0x0178,
00014 };
00015 static const unsigned short georgian_ps_2uni_2[39] = {
00016     /* 0xc0 */
00017     0x10d0, 0x10d1, 0x10d2, 0x10d3, 0x10d4, 0x10d5, 0x10d6, 0x10f1,
00018     0x10d7, 0x10d8, 0x10d9, 0x10da, 0x10db, 0x10dc, 0x10f2, 0x10dd,
00019     /* 0xd0 */
00020     0x10de, 0x10df, 0x10e0, 0x10e1, 0x10e2, 0x10f3, 0x10e3, 0x10e4,
00021     0x10e5, 0x10e6, 0x10e7, 0x10e8, 0x10e9, 0x10ea, 0x10eb, 0x10ec,
00022     /* 0xe0 */
00023     0x10ed, 0x10ee, 0x10f4, 0x10ef, 0x10f0, 0x10f5,
00024 };
00025
00026 static int
00027 georgian_ps_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00028 {
00029     unsigned char c = *s;
00030     if (c >= 0x80 && c < 0xa0)
00031         *pwc = (ucs4_t) georgian_ps_2uni_1[c-0x80];
00032     else if (c >= 0xc0 && c < 0xe6)
00033         *pwc = (ucs4_t) georgian_ps_2uni_2[c-0xc0];
00034     else
00035         *pwc = (ucs4_t) c;
00036     return 1;
00037 }
00038
00039 static const unsigned char georgian_ps_page00[32] = {
00040     0x80, 0x81, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00041     0x00, 0x00, 0x00, 0x00, 0x00, 0x8d, 0x8e, 0x8f, /* 0x88-0x8f */
00042     0x90, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00043     0x00, 0x00, 0x00, 0x00, 0x00, 0x9d, 0x9e, 0x00, /* 0x98-0x9f */
00044 };
00045 static const unsigned char georgian_ps_page01[72] = {
00046     0x00, 0x00, 0x8c, 0x9c, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00047     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00048     0x8a, 0x9a, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00049     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00050     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00051     0x9f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00052     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00053     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00054     0x00, 0x00, 0x83, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00055 };
00056 static const unsigned char georgian_ps_page02[32] = {
00057     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x88, 0x00, /* 0xc0-0xc7 */
00058     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00059     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00060     0x00, 0x00, 0x00, 0x00, 0x98, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00061 };
00062 static const unsigned char georgian_ps_page10[40] = {
00063     0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc8, /* 0xd0-0xd7 */
00064     0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xcf, 0xd0, 0xd1, /* 0xd8-0xdf */
00065     0xd2, 0xd3, 0xd4, 0xd6, 0xd7, 0xd8, 0xd9, 0xda, /* 0xe0-0xe7 */

```

```

00066 0xdb, 0xdc, 0xdd, 0xde, 0xdf, 0xe0, 0xe1, 0xe3, /* 0xe8-0xef */
00067 0xe4, 0xc7, 0xce, 0xd5, 0xe2, 0xe5, 0x00, 0x00, /* 0xf0-0xf7 */
00068 };
00069 static const unsigned char georgian_ps_page20[48] = {
00070 0x00, 0x00, 0x00, 0x96, 0x97, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00071 0x91, 0x92, 0x82, 0x00, 0x93, 0x94, 0x84, 0x00, /* 0x18-0x1f */
00072 0x86, 0x87, 0x95, 0x00, 0x00, 0x00, 0x85, 0x00, /* 0x20-0x27 */
00073 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00074 0x89, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00075 0x00, 0x8b, 0x9b, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00076 };
00077
00078 static int
00079 georgian_ps_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00080 {
00081     unsigned char c = 0;
00082     if (wc < 0x0080) {
00083         *r = wc;
00084         return 1;
00085     }
00086     else if (wc >= 0x0080 && wc < 0x00a0)
00087         c = georgian_ps_page00[wc-0x0080];
00088     else if ((wc >= 0x00a0 && wc < 0x00c0) || (wc >= 0x00e6 && wc < 0x0100))
00089         c = wc;
00090     else if (wc >= 0x0150 && wc < 0x0198)
00091         c = georgian_ps_page01[wc-0x0150];
00092     else if (wc >= 0x02c0 && wc < 0x02e0)
00093         c = georgian_ps_page02[wc-0x02c0];
00094     else if (wc >= 0x10d0 && wc < 0x10f8)
00095         c = georgian_ps_page10[wc-0x10d0];
00096     else if (wc >= 0x2010 && wc < 0x2040)
00097         c = georgian_ps_page20[wc-0x2010];
00098     else if (wc == 0x2122)
00099         c = 0x99;
00100     if (c != 0) {
00101         *r = c;
00102         return 1;
00103     }
00104     return RET_ILSEQ;
00105 }

```

12.269 iso8859_1.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_1.h,v 1.3 2000/11/29 17:40:30 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-1
00005  */
00006
00007 static int
00008 iso8859_1_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00009 {
00010     unsigned char c = *s;
00011     *pwc = (ucs4_t) c;
00012     return 1;
00013 }
00014
00015 static int
00016 iso8859_1_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00017 {
00018     if (wc < 0x0100) {
00019         *r = wc;
00020         return 1;
00021     }
00022     return RET_ILSEQ;
00023 }

```

12.270 iso8859_10.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_10.h,v 1.3 2000/11/29 17:40:30 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-10
00005  */
00006
00007 #ifdef NEED_TOWC
00008 static const unsigned short iso8859_10_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0x0104, 0x0112, 0x0122, 0x012a, 0x0128, 0x0136, 0x00a7,
00011     0x013b, 0x0110, 0x0160, 0x0166, 0x017d, 0x00ad, 0x016a, 0x014a,
00012     /* 0xb0 */
00013     0x00b0, 0x0105, 0x0113, 0x0123, 0x012b, 0x0129, 0x0137, 0x00b7,

```

```

00014 0x013c, 0x0111, 0x0161, 0x0167, 0x017e, 0x2015, 0x016b, 0x014b,
00015 /* 0xc0 */
00016 0x0100, 0x00c1, 0x00c2, 0x00c3, 0x00c4, 0x00c5, 0x00c6, 0x012e,
00017 0x010c, 0x00c9, 0x0118, 0x00cb, 0x0116, 0x00cd, 0x00ce, 0x00cf,
00018 /* 0xd0 */
00019 0x00d0, 0x0145, 0x014c, 0x00d3, 0x00d4, 0x00d5, 0x00d6, 0x0168,
00020 0x00d8, 0x0172, 0x00da, 0x00db, 0x00dc, 0x00dd, 0x00de, 0x00df,
00021 /* 0xe0 */
00022 0x0101, 0x00e1, 0x00e2, 0x00e3, 0x00e4, 0x00e5, 0x00e6, 0x012f,
00023 0x010d, 0x00e9, 0x0119, 0x00eb, 0x0117, 0x00ed, 0x00ee, 0x00ef,
00024 /* 0xf0 */
00025 0x00f0, 0x0146, 0x014d, 0x00f3, 0x00f4, 0x00f5, 0x00f6, 0x0169,
00026 0x00f8, 0x0173, 0x00fa, 0x00fb, 0x00fc, 0x00fd, 0x00fe, 0x0138,
00027 };
00028
00029 static int
00030 iso8859_10_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c < 0xa0)
00034         *pwc = (ucs4_t) c;
00035     else
00036         *pwc = (ucs4_t) iso8859_10_2uni[c-0xa0];
00037     return 1;
00038 }
00039 #endif /* NEED_TOWC */
00040
00041 #ifdef NEED_TOMB
00042 static const unsigned char iso8859_10_page00[224] = {
00043     0xa0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xa7, /* 0xa0-0xa7 */
00044     0x00, 0x00, 0x00, 0x00, 0x00, 0xad, 0x00, 0x00, /* 0xa8-0xaf */
00045     0xb0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xb7, /* 0xb0-0xb7 */
00046     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00047     0x00, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0x00, /* 0xc0-0xc7 */
00048     0x00, 0xc9, 0x00, 0xcb, 0x00, 0xcd, 0xce, 0xcf, /* 0xc8-0xcf */
00049     0xd0, 0x00, 0x00, 0xd3, 0xd4, 0xd5, 0xd6, 0x00, /* 0xd0-0xd7 */
00050     0xd8, 0x00, 0xda, 0xdb, 0xdc, 0xdd, 0xde, 0xdf, /* 0xd8-0xdf */
00051     0x00, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0x00, /* 0xe0-0xe7 */
00052     0x00, 0xe9, 0x00, 0xeb, 0x00, 0xed, 0xee, 0xef, /* 0xe8-0xef */
00053     0xf0, 0x00, 0x00, 0xf3, 0xf4, 0xf5, 0xf6, 0x00, /* 0xf0-0xf7 */
00054     0xf8, 0x00, 0xfa, 0xfb, 0xfc, 0xfd, 0xfe, 0x00, /* 0xf8-0xff */
00055     /* 0x0100 */
00056     0xc0, 0xe0, 0x00, 0x00, 0xa1, 0xb1, 0x00, 0x00, /* 0x00-0x07 */
00057     0x00, 0x00, 0x00, 0x00, 0xc8, 0xe8, 0x00, 0x00, /* 0x08-0x0f */
00058     0xa9, 0xb9, 0xa2, 0xb2, 0x00, 0x00, 0xcc, 0xec, /* 0x10-0x17 */
00059     0xca, 0xea, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00060     0x00, 0x00, 0xa3, 0xb3, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00061     0xa5, 0xb5, 0xa4, 0xb4, 0x00, 0x00, 0xc7, 0xe7, /* 0x28-0x2f */
00062     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xa6, 0xb6, /* 0x30-0x37 */
00063     0xff, 0x00, 0x00, 0xa8, 0xb8, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00064     0x00, 0x00, 0x00, 0x00, 0x00, 0xd1, 0xf1, 0x00, /* 0x40-0x47 */
00065     0x00, 0x00, 0xaf, 0xbf, 0xd2, 0xf2, 0x00, 0x00, /* 0x48-0x4f */
00066     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00067     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00068     0xaa, 0xba, 0x00, 0x00, 0x00, 0x00, 0xab, 0xbb, /* 0x60-0x67 */
00069     0xd7, 0xf7, 0xae, 0xbe, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00070     0x00, 0x00, 0xd9, 0xf9, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00071     0x00, 0x00, 0x00, 0x00, 0x00, 0xac, 0xbc, 0x00, /* 0x78-0x7f */
00072 };
00073
00074 static int
00075 iso8859_10_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00076 {
00077     (void)conv; (void)n;
00078     unsigned char c = 0;
00079     if (wc < 0x00a0) {
00080         *r = wc;
00081         return 1;
00082     }
00083     else if (wc >= 0x00a0 && wc < 0x0180)
00084         c = iso8859_10_page00[wc-0x00a0];
00085     else if (wc == 0x2015)
00086         c = 0xbd;
00087     if (c != 0) {
00088         *r = c;
00089         return 1;
00090     }
00091     return RET_ILSEQ;
00092 }
00093 #endif /* NEED_TOMB */

```

12.271 iso8859_11.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_11.h,v 1.2 2002/10/09 16:38:19 tsi Exp $ */
00002

```



```

00003 /*
00004  * ISO8859-11
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_11_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0x0e01, 0x0e02, 0x0e03, 0x0e04, 0x0e05, 0x0e06, 0x0e07,
00011     0x0e08, 0x0e09, 0x0e0a, 0x0e0b, 0x0e0c, 0x0e0d, 0x0e0e, 0x0e0f,
00012     /* 0xb0 */
00013     0x0e10, 0x0e11, 0x0e12, 0x0e13, 0x0e14, 0x0e15, 0x0e16, 0x0e17,
00014     0x0e18, 0x0e19, 0x0e1a, 0x0e1b, 0x0e1c, 0x0e1d, 0x0e1e, 0x0e1f,
00015     /* 0xc0 */
00016     0x0e20, 0x0e21, 0x0e22, 0x0e23, 0x0e24, 0x0e25, 0x0e26, 0x0e27,
00017     0x0e28, 0x0e29, 0x0e2a, 0x0e2b, 0x0e2c, 0x0e2d, 0x0e2e, 0x0e2f,
00018     /* 0xd0 */
00019     0x0e30, 0x0e31, 0x0e32, 0x0e33, 0x0e34, 0x0e35, 0x0e36, 0x0e37,
00020     0x0e38, 0x0e39, 0x0e3a, 0xffff, 0xffff, 0xffff, 0xffff, 0x0e3f,
00021     /* 0xe0 */
00022     0x0e40, 0x0e41, 0x0e42, 0x0e43, 0x0e44, 0x0e45, 0x0e46, 0x0e47,
00023     0x0e48, 0x0e49, 0x0e4a, 0x0e4b, 0x0e4c, 0x0e4d, 0x0e4e, 0x0e4f,
00024     /* 0xf0 */
00025     0x0e50, 0x0e51, 0x0e52, 0x0e53, 0x0e54, 0x0e55, 0x0e56, 0x0e57,
00026     0x0e58, 0x0e59, 0x0e5a, 0x0e5b, 0xffff, 0xffff, 0xffff, 0xffff,
00027 };
00028
00029 static int
00030 iso8859_11_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c < 0x80) {
00034         *pwc = (ucs4_t) c;
00035         return 1;
00036     }
00037     else if (c < 0xa0) {
00038     }
00039     else {
00040         unsigned short wc = iso8859_11_2uni[c-0xa0];
00041         if (wc != 0xffff) {
00042             *pwc = (ucs4_t) wc;
00043             return 1;
00044         }
00045     }
00046     return RET_ILSEQ;
00047 }
00048 #endif /* NEED_TOWC */
00049
00050 #ifndef NEED_TOMB
00051 static const unsigned char iso8859_11_page0e[96] = {
00052     0x00, 0xa1, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, /* 0x00-0x07 */
00053     0xa8, 0xa9, 0xaa, 0xab, 0xac, 0xad, 0xae, 0xaf, /* 0x08-0x0f */
00054     0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5, 0xb6, 0xb7, /* 0x10-0x17 */
00055     0xb8, 0xb9, 0xba, 0xbb, 0xbc, 0xbd, 0xbe, 0xbf, /* 0x18-0x1f */
00056     0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0x20-0x27 */
00057     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0x28-0x2f */
00058     0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, /* 0x30-0x37 */
00059     0xd8, 0xd9, 0xda, 0xdb, 0xdc, 0xdd, 0xde, 0xdf, /* 0x38-0x3f */
00060     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0x40-0x47 */
00061     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0x48-0x4f */
00062     0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0x50-0x57 */
00063     0xf8, 0xf9, 0xfa, 0xfb, 0xfc, 0xfd, 0xfe, 0xff, /* 0x58-0x5f */
00064 };
00065
00066 static int
00067 iso8859_11_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00068 {
00069     (void)conv; (void)n;
00070     unsigned char c = 0;
00071     if (wc < 0x0080 || wc == 0x00a0) {
00072         *r = wc;
00073         return 1;
00074     }
00075     else if (wc >= 0x0e00 && wc < 0x0e60)
00076         c = iso8859_11_page0e[wc-0x0e00];
00077     if (c != 0) {
00078         *r = c;
00079         return 1;
00080     }
00081     return RET_ILSEQ;
00082 }
00083 #endif /* NEED_TOMB */

```

12.272 iso8859_13.h

```
00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_13.h,v 1.2 2000/11/28 16:10:26 dawes Exp $ */
```

```

00002
00003 /*
00004  * ISO-8859-13
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_13_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0x201d, 0x00a2, 0x00a3, 0x00a4, 0x201e, 0x00a6, 0x00a7,
00011     0x00d8, 0x00a9, 0x0156, 0x00ab, 0x00ac, 0x00ad, 0x00ae, 0x00c6,
00012     /* 0xb0 */
00013     0x00b0, 0x00b1, 0x00b2, 0x00b3, 0x201c, 0x00b5, 0x00b6, 0x00b7,
00014     0x00f8, 0x00b9, 0x0157, 0x00bb, 0x00bc, 0x00bd, 0x00be, 0x00e6,
00015     /* 0xc0 */
00016     0x0104, 0x012e, 0x0100, 0x0106, 0x00c4, 0x00c5, 0x0118, 0x0112,
00017     0x010c, 0x00c9, 0x0179, 0x0116, 0x0122, 0x0136, 0x012a, 0x013b,
00018     /* 0xd0 */
00019     0x0160, 0x0143, 0x0145, 0x00d3, 0x014c, 0x00d5, 0x00d6, 0x00d7,
00020     0x0172, 0x0141, 0x015a, 0x016a, 0x00dc, 0x017b, 0x017d, 0x00df,
00021     /* 0xe0 */
00022     0x0105, 0x012f, 0x0101, 0x0107, 0x00e4, 0x00e5, 0x0119, 0x0113,
00023     0x010d, 0x00e9, 0x017a, 0x0117, 0x0123, 0x0137, 0x012b, 0x013c,
00024     /* 0xf0 */
00025     0x0161, 0x0144, 0x0146, 0x00f3, 0x014d, 0x00f5, 0x00f6, 0x00f7,
00026     0x0173, 0x0142, 0x015b, 0x016b, 0x00fc, 0x017c, 0x017e, 0x2019,
00027 };
00028
00029 static int
00030 iso8859_13_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c < 0xa0)
00034         *pwc = (ucs4_t) c;
00035     else
00036         *pwc = (ucs4_t) iso8859_13_2uni[c-0xa0];
00037     return 1;
00038 }
00039 #endif /* NEED_TOWC */
00040
00041 #ifndef NEED_TOMB
00042 static const unsigned char iso8859_13_page00[224] = {
00043     0xa0, 0x00, 0xa2, 0xa3, 0xa4, 0x00, 0xa6, 0xa7, /* 0xa0-0xa7 */
00044     0x00, 0xa9, 0x00, 0xab, 0xac, 0xad, 0xae, 0x00, /* 0xa8-0xaf */
00045     0xb0, 0xb1, 0xb2, 0xb3, 0x00, 0xb5, 0xb6, 0xb7, /* 0xb0-0xb7 */
00046     0x00, 0xb9, 0x00, 0xbb, 0xbc, 0xbd, 0xbe, 0x00, /* 0xb8-0xbf */
00047     0x00, 0x00, 0x00, 0x00, 0xc4, 0xc5, 0xaf, 0x00, /* 0xc0-0xc7 */
00048     0x00, 0xc9, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00049     0x00, 0x00, 0x00, 0xd3, 0x00, 0xd5, 0xd6, 0xd7, /* 0xd0-0xd7 */
00050     0xa8, 0x00, 0x00, 0x00, 0xdc, 0x00, 0x00, 0xdf, /* 0xd8-0xdf */
00051     0x00, 0x00, 0x00, 0x00, 0xe4, 0xe5, 0xbf, 0x00, /* 0xe0-0xe7 */
00052     0x00, 0xe9, 0x00, 0xce, 0x00, 0x00, 0x00, 0x00, /* 0xe8-0xef */
00053     0x00, 0x00, 0x00, 0xf3, 0x00, 0xf5, 0xf6, 0xf7, /* 0xf0-0xf7 */
00054     0xb8, 0x00, 0x00, 0x00, 0xfc, 0x00, 0x00, 0x00, /* 0xf8-0xff */
00055     /* 0x0100 */
00056     0xc2, 0xe2, 0x00, 0x00, 0xc0, 0xe0, 0xc3, 0xe3, /* 0x00-0x07 */
00057     0x00, 0x00, 0x00, 0x00, 0xc8, 0xe8, 0x00, 0x00, /* 0x08-0x0f */
00058     0x00, 0x00, 0xc7, 0xe7, 0x00, 0x00, 0xcb, 0xeb, /* 0x10-0x17 */
00059     0xc6, 0xe6, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00060     0x00, 0x00, 0xcc, 0xec, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00061     0x00, 0x00, 0xce, 0xee, 0x00, 0x00, 0xc1, 0xe1, /* 0x28-0x2f */
00062     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xcd, 0xed, /* 0x30-0x37 */
00063     0x00, 0x00, 0x00, 0xcf, 0xef, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00064     0x00, 0xd9, 0xf9, 0xd1, 0xf1, 0xd2, 0xf2, 0x00, /* 0x40-0x47 */
00065     0x00, 0x00, 0x00, 0x00, 0xd4, 0xf4, 0x00, 0x00, /* 0x48-0x4f */
00066     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xaa, 0xba, /* 0x50-0x57 */
00067     0x00, 0x00, 0xda, 0xfa, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00068     0xd0, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00069     0x00, 0x00, 0xdb, 0xfb, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00070     0x00, 0x00, 0xd8, 0xf8, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00071     0x00, 0xca, 0xea, 0xdd, 0xfd, 0xde, 0xfe, 0x00, /* 0x78-0x7f */
00072 };
00073 static const unsigned char iso8859_13_page20[8] = {
00074     0x00, 0xff, 0x00, 0x00, 0xb4, 0xa1, 0xa5, 0x00, /* 0x18-0x1f */
00075 };
00076
00077 static int
00078 iso8859_13_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00079 {
00080     (void)conv; (void)n;
00081     unsigned char c = 0;
00082     if (wc < 0x00a0) {
00083         *r = wc;
00084         return 1;
00085     }
00086     else if (wc >= 0x00a0 && wc < 0x0180)
00087         c = iso8859_13_page00[wc-0x00a0];
00088     else if (wc >= 0x2018 && wc < 0x2020)

```

```

00089     c = iso8859_13_page20[wc-0x2018];
00090     if (c != 0) {
00091         *r = c;
00092         return 1;
00093     }
00094     return RET_ILSEQ;
00095 }
00096 #endif /* NEED_TOWC */

```

12.273 iso8859_14.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_14.h,v 1.3 2000/11/29 17:40:30 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-14
00005  */
00006
00007 #ifdef NEED_TOWC
00008 static const unsigned short iso8859_14_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0x1e02, 0x1e03, 0x00a3, 0x010a, 0x010b, 0x1e0a, 0x00a7,
00011     0x1e80, 0x00a9, 0x1e82, 0x1e0b, 0x1ef2, 0x00ad, 0x00ae, 0x0178,
00012     /* 0xb0 */
00013     0x1e1e, 0x1ef, 0x0120, 0x0121, 0x1e40, 0x1e41, 0x00b6, 0x1e56,
00014     0x1e81, 0x1e57, 0x1e83, 0x1e60, 0x1ef3, 0x1e84, 0x1e85, 0x1e61,
00015     /* 0xc0 */
00016     0x00c0, 0x00c1, 0x00c2, 0x00c3, 0x00c4, 0x00c5, 0x00c6, 0x00c7,
00017     0x00c8, 0x00c9, 0x00ca, 0x00cb, 0x00cc, 0x00cd, 0x00ce, 0x00cf,
00018     /* 0xd0 */
00019     0x0174, 0x00d1, 0x00d2, 0x00d3, 0x00d4, 0x00d5, 0x00d6, 0x1e6a,
00020     0x00d8, 0x00d9, 0x00da, 0x00db, 0x00dc, 0x00dd, 0x0176, 0x00df,
00021     /* 0xe0 */
00022     0x00e0, 0x00e1, 0x00e2, 0x00e3, 0x00e4, 0x00e5, 0x00e6, 0x00e7,
00023     0x00e8, 0x00e9, 0x00ea, 0x00eb, 0x00ec, 0x00ed, 0x00ee, 0x00ef,
00024     /* 0xf0 */
00025     0x0175, 0x00f1, 0x00f2, 0x00f3, 0x00f4, 0x00f5, 0x00f6, 0x1e6b,
00026     0x00f8, 0x00f9, 0x00fa, 0x00fb, 0x00fc, 0x00fd, 0x0177, 0x00ff,
00027 };
00028
00029 static int
00030 iso8859_14_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c >= 0xa0)
00034         *pwc = (ucs4_t) iso8859_14_2uni[c-0xa0];
00035     else
00036         *pwc = (ucs4_t) c;
00037     return 1;
00038 }
00039 #endif /* NEED_TOWC */
00040
00041 #ifdef NEED_TOMB
00042 static const unsigned char iso8859_14_page00[96] = {
00043     0xa0, 0x00, 0x00, 0xa3, 0x00, 0x00, 0x00, 0xa7, /* 0xa0-0xa7 */
00044     0x00, 0xa9, 0x00, 0x00, 0x00, 0xad, 0xae, 0x00, /* 0xa8-0xaf */
00045     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xb6, 0x00, /* 0xb0-0xb7 */
00046     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00047     0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0xc0-0xc7 */
00048     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0xc8-0xcf */
00049     0x00, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0x00, /* 0xd0-0xd7 */
00050     0xd8, 0xd9, 0xda, 0xdb, 0xdc, 0xdd, 0x00, 0xdf, /* 0xd8-0xdf */
00051     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0xe0-0xe7 */
00052     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0xe8-0xef */
00053     0x00, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0x00, /* 0xf0-0xf7 */
00054     0xf8, 0xf9, 0xfa, 0xfb, 0xfc, 0xfd, 0x00, 0xff, /* 0xf8-0xff */
00055 };
00056 static const unsigned char iso8859_14_page01_0[32] = {
00057     0x00, 0x00, 0xa4, 0xa5, 0x00, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00058     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00059     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00060     0xb2, 0xb3, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00061 };
00062 static const unsigned char iso8859_14_page01_1[16] = {
00063     0x00, 0x00, 0x00, 0x00, 0xd0, 0xf0, 0xde, 0xfe, /* 0x70-0x77 */
00064     0xaf, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00065 };
00066 static const unsigned char iso8859_14_page1e_0[136] = {
00067     0x00, 0x00, 0xa1, 0xa2, 0x00, 0x00, 0x00, 0x00, /* 0x00-0x07 */
00068     0x00, 0x00, 0xa6, 0xab, 0x00, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00069     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00070     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xb0, 0xb1, /* 0x18-0x1f */
00071     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00072     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00073     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00074     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */

```

```

00075 0xb4, 0xb5, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00076 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00077 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xb7, 0xb9, /* 0x50-0x57 */
00078 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00079 0xbb, 0xbf, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00080 0x00, 0x00, 0xd7, 0xf7, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00081 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00082 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00083 0xa8, 0xb8, 0xaa, 0xba, 0xbd, 0xbe, 0x00, 0x00, /* 0x80-0x87 */
00084 };
00085 static const unsigned char iso8859_14_page1e_1[8] = {
00086 0x00, 0x00, 0xac, 0xbc, 0x00, 0x00, 0x00, 0x00, /* 0xf0-0xf7 */
00087 };
00088
00089 static int
00090 iso8859_14_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00091 {
00092     (void)conv; (void)n;
00093     unsigned char c = 0;
00094     if (wc < 0x00a0) {
00095         *r = wc;
00096         return 1;
00097     }
00098     else if (wc >= 0x00a0 && wc < 0x0100)
00099         c = iso8859_14_page00[wc-0x00a0];
00100     else if (wc >= 0x0108 && wc < 0x0128)
00101         c = iso8859_14_page01_0[wc-0x0108];
00102     else if (wc >= 0x0170 && wc < 0x0180)
00103         c = iso8859_14_page01_1[wc-0x0170];
00104     else if (wc >= 0x1e00 && wc < 0x1e88)
00105         c = iso8859_14_page1e_0[wc-0x1e00];
00106     else if (wc >= 0x1ef0 && wc < 0x1ef8)
00107         c = iso8859_14_page1e_1[wc-0x1ef0];
00108     if (c != 0) {
00109         *r = c;
00110         return 1;
00111     }
00112     return RET_ILSEQ;
00113 }
00114 #endif /* NEED_TOMB */

```

12.274 iso8859_15.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_15.h,v 1.3 2000/11/29 17:40:31 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-15
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_15_2uni[32] = {
00009     /* 0xa0 */
00010     0x00a0, 0x00a1, 0x00a2, 0x00a3, 0x20ac, 0x00a5, 0x0160, 0x00a7,
00011     0x0161, 0x00a9, 0x00aa, 0x00ab, 0x00ac, 0x00ad, 0x00ae, 0x00af,
00012     /* 0xb0 */
00013     0x00b0, 0x00b1, 0x00b2, 0x00b3, 0x017d, 0x00b5, 0x00b6, 0x00b7,
00014     0x017e, 0x00b9, 0x00ba, 0x00bb, 0x0152, 0x0153, 0x0178, 0x00bf,
00015 };
00016
00017 static int
00018 iso8859_15_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00019 {
00020     unsigned char c = *s;
00021     if (c >= 0xa0 && c < 0xc0)
00022         *pwc = (ucs4_t) iso8859_15_2uni[c-0xa0];
00023     else
00024         *pwc = (ucs4_t) c;
00025     return 1;
00026 }
00027 #endif /* NEED_TOWC */
00028
00029 #ifndef NEED_TOMB
00030 static const unsigned char iso8859_15_page00[32] = {
00031     0xa0, 0xa1, 0xa2, 0xa3, 0x00, 0xa5, 0x00, 0xa7, /* 0xa0-0xa7 */
00032     0x00, 0xa9, 0xaa, 0xab, 0xac, 0xad, 0xae, 0xaf, /* 0xa8-0xaf */
00033     0xb0, 0xb1, 0xb2, 0xb3, 0x00, 0xb5, 0xb6, 0xb7, /* 0xb0-0xb7 */
00034     0x00, 0xb9, 0xba, 0xbb, 0x00, 0x00, 0x00, 0xbf, /* 0xb8-0xbf */
00035 };
00036 static const unsigned char iso8859_15_page01[48] = {
00037     0x00, 0x00, 0xbc, 0xbd, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00038     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00039     0xa6, 0xa8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00040     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00041     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00042     0xbe, 0x00, 0x00, 0x00, 0x00, 0xb4, 0xb8, 0x00, /* 0x78-0x7f */

```

```

00043 };
00044
00045 static int
00046 iso8859_15_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00047 {
00048     (void)conv; (void)n;
00049     unsigned char c = 0;
00050     if (wc < 0x00a0) {
00051         *r = wc;
00052         return 1;
00053     }
00054     else if (wc >= 0x00a0 && wc < 0x00c0)
00055         c = iso8859_15_page00[wc-0x00a0];
00056     else if (wc >= 0x00c0 && wc < 0x0100)
00057         c = wc;
00058     else if (wc >= 0x0150 && wc < 0x0180)
00059         c = iso8859_15_page01[wc-0x0150];
00060     else if (wc == 0x20ac)
00061         c = 0xa4;
00062     if (c != 0) {
00063         *r = c;
00064         return 1;
00065     }
00066     return RET_ILSEQ;
00067 }
00068 #endif /* NEED_TOMB */

```

12.275 iso8859_16.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_16.h,v 1.4 2003/07/15 17:33:45 pascal Exp $ */
00002
00003 /*
00004  * ISO-8859-16
00005  */
00006
00007 static const unsigned short iso8859_16_2uni[96] = {
00008     /* 0xa0 */
00009     0x00a0, 0x0104, 0x0105, 0x0141, 0x20ac, 0x201e, 0x0160, 0x00a7,
00010     0x0161, 0x00a9, 0x0218, 0x00ab, 0x0179, 0x00ad, 0x017a, 0x017b,
00011     /* 0xb0 */
00012     0x00b0, 0x00b1, 0x010c, 0x0142, 0x017d, 0x201d, 0x00b6, 0x00b7,
00013     0x017e, 0x010d, 0x0219, 0x00bb, 0x0152, 0x0153, 0x0178, 0x017c,
00014     /* 0xc0 */
00015     0x00c0, 0x00c1, 0x00c2, 0x0102, 0x00c4, 0x0106, 0x00c6, 0x00c7,
00016     0x00c8, 0x00c9, 0x00ca, 0x00cb, 0x00cc, 0x00cd, 0x00ce, 0x00cf,
00017     /* 0xd0 */
00018     0x0110, 0x0143, 0x00d2, 0x00d3, 0x00d4, 0x0150, 0x00d6, 0x015a,
00019     0x0170, 0x00d9, 0x00da, 0x00db, 0x00dc, 0x0118, 0x021a, 0x00df,
00020     /* 0xe0 */
00021     0x00e0, 0x00e1, 0x00e2, 0x0103, 0x00e4, 0x0107, 0x00e6, 0x00e7,
00022     0x00e8, 0x00e9, 0x00ea, 0x00eb, 0x00ec, 0x00ed, 0x00ee, 0x00ef,
00023     /* 0xf0 */
00024     0x0111, 0x0144, 0x00f2, 0x00f3, 0x00f4, 0x0151, 0x00f6, 0x015b,
00025     0x0171, 0x00f9, 0x00fa, 0x00fb, 0x00fc, 0x0119, 0x021b, 0x00ff,
00026 };
00027
00028 static int
00029 iso8859_16_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00030 {
00031     unsigned char c = *s;
00032     if (c < 0xa0)
00033         *pwc = (ucs4_t) c;
00034     else
00035         *pwc = (ucs4_t) iso8859_16_2uni[c-0xa0];
00036     return 1;
00037 }
00038
00039 static const unsigned char iso8859_16_page00[224] = {
00040     0xa0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xa7, /* 0xa0-0xa7 */
00041     0x00, 0xa9, 0x00, 0xab, 0x00, 0xad, 0x00, 0x00, /* 0xa8-0xaf */
00042     0xb0, 0xb1, 0x00, 0x00, 0x00, 0xb6, 0xb7, /* 0xb0-0xb7 */
00043     0x00, 0x00, 0x00, 0xbb, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00044     0xc0, 0xc1, 0xc2, 0x00, 0xc4, 0x00, 0xc6, 0xc7, /* 0xc0-0xc7 */
00045     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0xc8-0xcf */
00046     0x00, 0x00, 0xd2, 0xd3, 0xd4, 0x00, 0xd6, 0x00, /* 0xd0-0xd7 */
00047     0x00, 0xd9, 0xda, 0xdb, 0xdc, 0x00, 0x00, 0xdf, /* 0xd8-0xdf */
00048     0xe0, 0xe1, 0xe2, 0x00, 0xe4, 0x00, 0xe6, 0xe7, /* 0xe0-0xe7 */
00049     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0xe8-0xef */
00050     0x00, 0x00, 0xf2, 0xf3, 0xf4, 0x00, 0xf6, 0x00, /* 0xf0-0xf7 */
00051     0x00, 0xf9, 0xfa, 0xfb, 0xfc, 0x00, 0x00, 0xff, /* 0xf8-0xff */
00052     /* 0x0100 */
00053     0x00, 0x00, 0xc3, 0xe3, 0xa1, 0xa2, 0xc5, 0xe5, /* 0x00-0x07 */
00054     0x00, 0x00, 0x00, 0x00, 0xb2, 0xb9, 0x00, 0x00, /* 0x08-0x0f */
00055     0xd0, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00056     0xdd, 0xfd, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */

```

```

00057 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00058 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00059 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00060 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00061 0x00, 0xa3, 0xb3, 0xd1, 0xf1, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00062 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00063 0xd5, 0xf5, 0xbc, 0xbd, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00064 0x00, 0x00, 0xd7, 0xf7, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00065 0xa6, 0xa8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00066 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00067 0xd8, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00068 0xbe, 0xac, 0xae, 0xaf, 0xbf, 0xb4, 0xb8, 0x00, /* 0x78-0x7f */
00069 };
00070 static const unsigned char iso8859_16_page02[8] = {
00071 0xaa, 0xba, 0xde, 0xfe, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00072 };
00073 static const unsigned char iso8859_16_page20[8] = {
00074 0x00, 0x00, 0x00, 0x00, 0x00, 0xb5, 0xa5, 0x00, /* 0x18-0x1f */
00075 };
00076
00077 static int
00078 iso8859_16_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00079 {
00080     unsigned char c = 0;
00081     if (wc < 0x00a0) {
00082         *r = wc;
00083         return 1;
00084     }
00085     else if (wc >= 0x00a0 && wc < 0x0180)
00086         c = iso8859_16_page00[wc-0x00a0];
00087     else if (wc >= 0x0218 && wc < 0x0220)
00088         c = iso8859_16_page02[wc-0x0218];
00089     else if (wc >= 0x2018 && wc < 0x2020)
00090         c = iso8859_16_page20[wc-0x2018];
00091     else if (wc == 0x20ac)
00092         c = 0xa4;
00093     if (c != 0) {
00094         *r = c;
00095         return 1;
00096     }
00097     return RET_ILSEQ;
00098 }

```

12.276 iso8859_2.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_2.h,v 1.3 2000/11/29 17:40:31 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-2
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_2_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0x0104, 0x02d8, 0x0141, 0x00a4, 0x013d, 0x015a, 0x00a7,
00011     0x00a8, 0x0160, 0x015e, 0x0164, 0x0179, 0x00ad, 0x017d, 0x017b,
00012     /* 0xb0 */
00013     0x00b0, 0x0105, 0x02db, 0x0142, 0x00b4, 0x013e, 0x015b, 0x02c7,
00014     0x00b8, 0x0161, 0x015f, 0x0165, 0x017a, 0x02dd, 0x017e, 0x017c,
00015     /* 0xc0 */
00016     0x0154, 0x00c1, 0x00c2, 0x0102, 0x00c4, 0x0139, 0x0106, 0x00c7,
00017     0x010c, 0x00c9, 0x0118, 0x00cb, 0x011a, 0x00cd, 0x00ce, 0x010e,
00018     /* 0xd0 */
00019     0x0110, 0x0143, 0x0147, 0x00d3, 0x00d4, 0x0150, 0x00d6, 0x00d7,
00020     0x0158, 0x016e, 0x00da, 0x0170, 0x00dc, 0x00dd, 0x0162, 0x00df,
00021     /* 0xe0 */
00022     0x0155, 0x00e1, 0x00e2, 0x0103, 0x00e4, 0x013a, 0x0107, 0x00e7,
00023     0x010d, 0x00e9, 0x0119, 0x00eb, 0x011b, 0x00ed, 0x00ee, 0x010f,
00024     /* 0xf0 */
00025     0x0111, 0x0144, 0x0148, 0x00f3, 0x00f4, 0x0151, 0x00f6, 0x00f7,
00026     0x0159, 0x016f, 0x00fa, 0x0171, 0x00fc, 0x00fd, 0x0163, 0x02d9,
00027 };
00028
00029 static int
00030 iso8859_2_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c < 0xa0)
00034         *pwc = (ucs4_t) c;
00035     else
00036         *pwc = (ucs4_t) iso8859_2_2uni[c-0xa0];
00037     return 1;
00038 }
00039 #endif /* NEED_TOWC */
00040

```

```

00041 #ifndef NEED_TOMB
00042 static const unsigned char iso8859_2_page00[224] = {
00043     0xa0, 0x00, 0x00, 0x00, 0xa4, 0x00, 0x00, 0xa7, /* 0xa0-0xa7 */
00044     0xa8, 0x00, 0x00, 0x00, 0xad, 0x00, 0x00, /* 0xa8-0xaf */
00045     0xb0, 0x00, 0x00, 0xb4, 0x00, 0x00, /* 0xb0-0xb7 */
00046     0xb8, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00047     0xc0, 0xc1, 0xc2, 0xc4, 0xc0, 0xc0, 0xc7, /* 0xc0-0xc7 */
00048     0xc0, 0xc9, 0xc0, 0xcb, 0xc0, 0xcd, 0xce, 0xc0, /* 0xc8-0xcf */
00049     0xd0, 0x00, 0xd0, 0xd3, 0xd4, 0x00, 0xd6, 0xd7, /* 0xd0-0xd7 */
00050     0x00, 0x00, 0xda, 0x00, 0xdc, 0xdd, 0x00, 0xdf, /* 0xd8-0xdf */
00051     0x00, 0xe1, 0xe2, 0x00, 0xe4, 0x00, 0x00, 0xe7, /* 0xe0-0xe7 */
00052     0x00, 0xe9, 0x00, 0xeb, 0x00, 0xed, 0xee, 0x00, /* 0xe8-0xef */
00053     0x00, 0x00, 0x00, 0xf3, 0xf4, 0x00, 0xf6, 0xf7, /* 0xf0-0xf7 */
00054     0x00, 0x00, 0xfa, 0x00, 0xfc, 0xfd, 0x00, 0x00, /* 0xf8-0xff */
00055     /* 0x0100 */
00056     0x00, 0x00, 0xc3, 0xe3, 0xa1, 0xb1, 0xc6, 0xe6, /* 0x00-0x07 */
00057     0x00, 0x00, 0x00, 0x00, 0xc8, 0xe8, 0xcf, 0xef, /* 0x08-0x0f */
00058     0xd0, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00059     0xca, 0xea, 0xcc, 0xec, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00060     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00061     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00062     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00063     0x00, 0xc5, 0xe5, 0x00, 0x00, 0xa5, 0xb5, 0x00, /* 0x38-0x3f */
00064     0x00, 0xa3, 0xb3, 0xd1, 0xf1, 0x00, 0x00, 0xd2, /* 0x40-0x47 */
00065     0xf2, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00066     0xd5, 0xf5, 0x00, 0x00, 0xc0, 0xe0, 0x00, 0x00, /* 0x50-0x57 */
00067     0xd8, 0xf8, 0xa6, 0xb6, 0x00, 0x00, 0xaa, 0xba, /* 0x58-0x5f */
00068     0xa9, 0xb9, 0xde, 0xfe, 0xab, 0xbb, 0x00, 0x00, /* 0x60-0x67 */
00069     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xd9, 0xf9, /* 0x68-0x6f */
00070     0xdb, 0xfb, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00071     0x00, 0xac, 0xbc, 0xaf, 0xbf, 0xae, 0xbe, 0x00, /* 0x78-0x7f */
00072 };
00073 static const unsigned char iso8859_2_page02[32] = {
00074     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xb7, /* 0xc0-0xc7 */
00075     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00076     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00077     0xa2, 0xff, 0x00, 0xb2, 0x00, 0xbd, 0x00, 0x00, /* 0xd8-0xdf */
00078 };
00079
00080 /*
00081 static int
00082 iso8859_2_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00083 {
00084     unsigned char c = 0;
00085     if (wc < 0x00a0) {
00086         *r = wc;
00087         return 1;
00088     }
00089     else if (wc >= 0x00a0 && wc < 0x0180)
00090         c = iso8859_2_page00[wc-0x00a0];
00091     else if (wc >= 0x02c0 && wc < 0x02e0)
00092         c = iso8859_2_page02[wc-0x02c0];
00093     if (c != 0) {
00094         *r = c;
00095         return 1;
00096     }
00097     return RET_ILSEQ;
00098 }
00099 */
00100 #endif /* NEED_TOMB */

```

12.277 iso8859_3.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_3.h,v 1.3 2000/11/29 17:40:31 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-3
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_3_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0x0126, 0x02d8, 0x00a3, 0x00a4, 0xfffd, 0x0124, 0x00a7,
00011     0x00a8, 0x0130, 0x015e, 0x011e, 0x0134, 0x00ad, 0xfffd, 0x017b,
00012     /* 0xb0 */
00013     0x00b0, 0x0127, 0x00b2, 0x00b3, 0x00b4, 0x00b5, 0x0125, 0x00b7,
00014     0x00b8, 0x0131, 0x015f, 0x011f, 0x0135, 0x00bd, 0xfffd, 0x017c,
00015     /* 0xc0 */
00016     0x00c0, 0x00c1, 0x00c2, 0xfffd, 0x00c4, 0x010a, 0x0108, 0x00c7,
00017     0x00c8, 0x00c9, 0x00ca, 0x00cb, 0x00cc, 0x00cd, 0x00ce, 0x00cf,
00018     /* 0xd0 */
00019     0xfffd, 0x00d1, 0x00d2, 0x00d3, 0x00d4, 0x0120, 0x00d6, 0x00d7,
00020     0x011c, 0x00d9, 0x00da, 0x00db, 0x00dc, 0x016c, 0x015c, 0x00df,
00021     /* 0xe0 */
00022     0x00e0, 0x00e1, 0x00e2, 0xfffd, 0x00e4, 0x010b, 0x0109, 0x00e7,

```

```

00023     0x00e8, 0x00e9, 0x00ea, 0x00eb, 0x00ec, 0x00ed, 0x00ee, 0x00ef,
00024     /* 0xf0 */
00025     0xffffd, 0x00f1, 0x00f2, 0x00f3, 0x00f4, 0x0121, 0x00f6, 0x00f7,
00026     0x011d, 0x00f9, 0x00fa, 0x00fb, 0x00fc, 0x016d, 0x015d, 0x02d9,
00027 };
00028
00029 static int
00030 iso8859_3_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c < 0xa0) {
00034         *pwc = (ucs4_t) c;
00035         return 1;
00036     }
00037     else {
00038         unsigned short wc = iso8859_3_uni[c-0xa0];
00039         if (wc != 0xffffd) {
00040             *pwc = (ucs4_t) wc;
00041             return 1;
00042         }
00043     }
00044     return RET_ILSEQ;
00045 }
00046 #endif /* NEED_TOWC */
00047
00048 #ifdef NEED_TOMB
00049 static const unsigned char iso8859_3_page00[96] = {
00050     0xa0, 0x00, 0x00, 0xa3, 0xa4, 0x00, 0x00, 0xa7, /* 0xa0-0xa7 */
00051     0xa8, 0x00, 0x00, 0x00, 0x00, 0xad, 0x00, 0x00, /* 0xa8-0xaf */
00052     0xb0, 0x00, 0xb2, 0xb3, 0xb4, 0xb5, 0x00, 0xb7, /* 0xb0-0xb7 */
00053     0xb8, 0x00, 0x00, 0x00, 0x00, 0xbd, 0x00, 0x00, /* 0xb8-0xbf */
00054     0xc0, 0xc1, 0xc2, 0x00, 0xc4, 0x00, 0x00, 0xc7, /* 0xc0-0xc7 */
00055     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0xc8-0xcf */
00056     0x00, 0xd1, 0xd2, 0xd3, 0xd4, 0x00, 0xd6, 0xd7, /* 0xd0-0xd7 */
00057     0x00, 0xd9, 0xda, 0xdb, 0xdc, 0x00, 0x00, 0xdf, /* 0xd8-0xdf */
00058     0xe0, 0xe1, 0xe2, 0x00, 0xe4, 0x00, 0x00, 0xe7, /* 0xe0-0xe7 */
00059     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0xe8-0xef */
00060     0x00, 0xf1, 0xf2, 0xf3, 0xf4, 0x00, 0xf6, 0xf7, /* 0xf0-0xf7 */
00061     0x00, 0xf9, 0xfa, 0xfb, 0xfc, 0x00, 0x00, 0x00, /* 0xf8-0xff */
00062 };
00063 static const unsigned char iso8859_3_page01[120] = {
00064     0xc6, 0xe6, 0xc5, 0xe5, 0x00, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00065     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00066     0x00, 0x00, 0x00, 0x00, 0xd8, 0xf8, 0xab, 0xbb, /* 0x18-0x1f */
00067     0xd5, 0xf5, 0x00, 0x00, 0xa6, 0xb6, 0xa1, 0xb1, /* 0x20-0x27 */
00068     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00069     0xa9, 0xb9, 0x00, 0x00, 0xac, 0xbc, 0x00, 0x00, /* 0x30-0x37 */
00070     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00071     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00072     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00073     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00074     0x00, 0x00, 0x00, 0x00, 0xde, 0xfe, 0xaa, 0xba, /* 0x58-0x5f */
00075     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00076     0x00, 0x00, 0x00, 0x00, 0xdd, 0xfd, 0x00, 0x00, /* 0x68-0x6f */
00077     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00078     0x00, 0x00, 0x00, 0xaf, 0xbf, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00079 };
00080 static const unsigned char iso8859_3_page02[8] = {
00081     0xa2, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00082 };
00083
00084 static int
00085 iso8859_3_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00086 {
00087     (void)conv; (void)n;
00088     unsigned char c = 0;
00089     if (wc < 0x00a0) {
00090         *r = wc;
00091         return 1;
00092     }
00093     else if (wc >= 0x00a0 && wc < 0x0100)
00094         c = iso8859_3_page00[wc-0x00a0];
00095     else if (wc >= 0x0108 && wc < 0x0180)
00096         c = iso8859_3_page01[wc-0x0108];
00097     else if (wc >= 0x02d8 && wc < 0x02e0)
00098         c = iso8859_3_page02[wc-0x02d8];
00099     if (c != 0) {
00100         *r = c;
00101         return 1;
00102     }
00103     return RET_ILSEQ;
00104 }
00105 #endif /* NEED_TOMB */

```


12.278 iso8859_4.h

```
00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_4.h,v 1.3 2000/11/29 17:40:31 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-4
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_4_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0x0104, 0x0138, 0x0156, 0x00a4, 0x0128, 0x013b, 0x00a7,
00011     0x00a8, 0x0160, 0x0112, 0x0122, 0x0166, 0x00ad, 0x017d, 0x00af,
00012     /* 0xb0 */
00013     0x00b0, 0x0105, 0x02db, 0x0157, 0x00b4, 0x0129, 0x013c, 0x02c7,
00014     0x00b8, 0x0161, 0x0113, 0x0123, 0x0167, 0x014a, 0x017e, 0x014b,
00015     /* 0xc0 */
00016     0x0100, 0x00c1, 0x00c2, 0x00c3, 0x00c4, 0x00c5, 0x00c6, 0x012e,
00017     0x010c, 0x00c9, 0x0118, 0x00cb, 0x0116, 0x00cd, 0x00ce, 0x012a,
00018     /* 0xd0 */
00019     0x0110, 0x0145, 0x014c, 0x0136, 0x00d4, 0x00d5, 0x00d6, 0x00d7,
00020     0x00d8, 0x0172, 0x00da, 0x00db, 0x00dc, 0x0168, 0x016a, 0x00df,
00021     /* 0xe0 */
00022     0x0101, 0x00e1, 0x00e2, 0x00e3, 0x00e4, 0x00e5, 0x00e6, 0x012f,
00023     0x010d, 0x00e9, 0x0119, 0x00eb, 0x0117, 0x00ed, 0x00ee, 0x012b,
00024     /* 0xf0 */
00025     0x0111, 0x0146, 0x014d, 0x0137, 0x00f4, 0x00f5, 0x00f6, 0x00f7,
00026     0x00f8, 0x0173, 0x00fa, 0x00fb, 0x00fc, 0x0169, 0x016b, 0x02d9,
00027 };
00028
00029 static int
00030 iso8859_4_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c < 0xa0)
00034         *pwc = (ucs4_t) c;
00035     else
00036         *pwc = (ucs4_t) iso8859_4_2uni[c-0xa0];
00037     return 1;
00038 }
00039 #endif /* NEED_TOWC */
00040
00041 #ifndef NEED_TOMB
00042 static const unsigned char iso8859_4_page00[224] = {
00043     0xa0, 0x00, 0x00, 0x00, 0xa4, 0x00, 0x00, 0xa7, /* 0xa0-0xa7 */
00044     0xa8, 0x00, 0x00, 0x00, 0xad, 0x00, 0xaf, /* 0xa8-0xaf */
00045     0xb0, 0x00, 0x00, 0x00, 0xb4, 0x00, 0x00, /* 0xb0-0xb7 */
00046     0xb8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00047     0x00, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0x00, /* 0xc0-0xc7 */
00048     0x00, 0xc9, 0x00, 0xcb, 0x00, 0xcd, 0xce, 0x00, /* 0xc8-0xcf */
00049     0x00, 0x00, 0x00, 0x00, 0xd4, 0xd5, 0xd6, 0xd7, /* 0xd0-0xd7 */
00050     0xd8, 0x00, 0xda, 0xdb, 0xdc, 0x00, 0x00, 0xdf, /* 0xd8-0xdf */
00051     0x00, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0x00, /* 0xe0-0xe7 */
00052     0x00, 0xe9, 0x00, 0xeb, 0x00, 0xed, 0xee, 0x00, /* 0xe8-0xef */
00053     0x00, 0x00, 0x00, 0x00, 0xf4, 0xf5, 0xf6, 0xf7, /* 0xf0-0xf7 */
00054     0xf8, 0x00, 0xfa, 0xfb, 0xfc, 0x00, 0x00, 0x00, /* 0xf8-0xff */
00055     /* 0x0100 */
00056     0xc0, 0xe0, 0x00, 0x00, 0xa1, 0xb1, 0x00, 0x00, /* 0x00-0x07 */
00057     0x00, 0x00, 0x00, 0x00, 0xc8, 0xe8, 0x00, 0x00, /* 0x08-0x0f */
00058     0xd0, 0xf0, 0xaa, 0xba, 0x00, 0x00, 0xcc, 0xec, /* 0x10-0x17 */
00059     0xca, 0xea, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00060     0x00, 0x00, 0xab, 0xbb, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00061     0xa5, 0xb5, 0xcf, 0xef, 0x00, 0x00, 0xc7, 0xe7, /* 0x28-0x2f */
00062     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xd3, 0xf3, /* 0x30-0x37 */
00063     0xa2, 0x00, 0x00, 0xa6, 0xb6, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00064     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf1, 0x00, /* 0x40-0x47 */
00065     0x00, 0x00, 0xbd, 0xbf, 0xd2, 0xf2, 0x00, 0x00, /* 0x48-0x4f */
00066     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xa3, 0xb3, /* 0x50-0x57 */
00067     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00068     0xa9, 0xb9, 0x00, 0x00, 0x00, 0x00, 0xac, 0xbc, /* 0x60-0x67 */
00069     0xdd, 0xfd, 0xde, 0xfe, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00070     0x00, 0x00, 0xd9, 0xf9, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00071     0x00, 0x00, 0x00, 0x00, 0x00, 0xae, 0xbe, 0x00, /* 0x78-0x7f */
00072 };
00073 static const unsigned char iso8859_4_page02[32] = {
00074     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xb7, /* 0xc0-0xc7 */
00075     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00076     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00077     0x00, 0xff, 0x00, 0xb2, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00078 };
00079
00080 static int
00081 iso8859_4_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00082 {
00083     (void)conv; (void)n;
00084     unsigned char c = 0;
00085     if (wc < 0x00a0) {
```

```

00086     *r = wc;
00087     return 1;
00088 }
00089 else if (wc >= 0x00a0 && wc < 0x0180)
00090     c = iso8859_4_page00[wc-0x00a0];
00091 else if (wc >= 0x02c0 && wc < 0x02e0)
00092     c = iso8859_4_page02[wc-0x02c0];
00093 if (c != 0) {
00094     *r = c;
00095     return 1;
00096 }
00097 return RET_ILSEQ;
00098 }
00099 #endif /* NEED_TOMB */

```

12.279 iso8859_5.h

```

00001 /* $XFree86: xc/lib/X11/UniConv/iso8859_5.h,v 1.3 2000/11/29 17:40:32 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-5
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_5_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0x0401, 0x0402, 0x0403, 0x0404, 0x0405, 0x0406, 0x0407,
00011     0x0408, 0x0409, 0x040a, 0x040b, 0x040c, 0x00ad, 0x040e, 0x040f,
00012     /* 0xb0 */
00013     0x0410, 0x0411, 0x0412, 0x0413, 0x0414, 0x0415, 0x0416, 0x0417,
00014     0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e, 0x041f,
00015     /* 0xc0 */
00016     0x0420, 0x0421, 0x0422, 0x0423, 0x0424, 0x0425, 0x0426, 0x0427,
00017     0x0428, 0x0429, 0x042a, 0x042b, 0x042c, 0x042d, 0x042e, 0x042f,
00018     /* 0xd0 */
00019     0x0430, 0x0431, 0x0432, 0x0433, 0x0434, 0x0435, 0x0436, 0x0437,
00020     0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e, 0x043f,
00021     /* 0xe0 */
00022     0x0440, 0x0441, 0x0442, 0x0443, 0x0444, 0x0445, 0x0446, 0x0447,
00023     0x0448, 0x0449, 0x044a, 0x044b, 0x044c, 0x044d, 0x044e, 0x044f,
00024     /* 0xf0 */
00025     0x2116, 0x0451, 0x0452, 0x0453, 0x0454, 0x0455, 0x0456, 0x0457,
00026     0x0458, 0x0459, 0x045a, 0x045b, 0x045c, 0x00a7, 0x045e, 0x045f,
00027 };
00028
00029 static int
00030 iso8859_5_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c < 0xa0)
00034         *pwc = (ucs4_t) c;
00035     else
00036         *pwc = (ucs4_t) iso8859_5_2uni[c-0xa0];
00037     return 1;
00038 }
00039 #endif /* NEED_TOWC */
00040
00041 #ifndef NEED_TOMB
00042 static const unsigned char iso8859_5_page00[16] = {
00043     0xa0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfd, /* 0xa0-0xa7 */
00044     0x00, 0x00, 0x00, 0x00, 0x00, 0xad, 0x00, 0x00, /* 0xa8-0xaf */
00045 };
00046 static const unsigned char iso8859_5_page04[96] = {
00047     0x00, 0xa1, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, /* 0x00-0x07 */
00048     0xa8, 0xa9, 0xaa, 0xab, 0xac, 0xad, 0xae, 0xaf, /* 0x08-0x0f */
00049     0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5, 0xb6, 0xb7, /* 0x10-0x17 */
00050     0xb8, 0xb9, 0xba, 0xbb, 0xbc, 0xbd, 0xbe, 0xbf, /* 0x18-0x1f */
00051     0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0x20-0x27 */
00052     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0x28-0x2f */
00053     0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, /* 0x30-0x37 */
00054     0xd8, 0xd9, 0xda, 0xdb, 0xdc, 0xdd, 0xde, 0xdf, /* 0x38-0x3f */
00055     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0x40-0x47 */
00056     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0x48-0x4f */
00057     0x00, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0x50-0x57 */
00058     0xf8, 0xf9, 0xfa, 0xfb, 0xfc, 0x00, 0xfe, 0xff, /* 0x58-0x5f */
00059 };
00060
00061 static int
00062 iso8859_5_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00063 {
00064     (void)conv; (void)n;
00065     unsigned char c = 0;
00066     if (wc < 0x00a0) {
00067         *r = wc;
00068         return 1;

```

```

00069     }
00070     else if (wc >= 0x00a0 && wc < 0x00b0)
00071         c = iso8859_5_page00[wc-0x00a0];
00072     else if (wc >= 0x0400 && wc < 0x0460)
00073         c = iso8859_5_page04[wc-0x0400];
00074     else if (wc == 0x2116)
00075         c = 0xf0;
00076     if (c != 0) {
00077         *r = c;
00078         return 1;
00079     }
00080     return RET_ILSEQ;
00081 }
00082 #endif /* NEED_TOMB */

```

12.280 iso8859_6.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_6.h,v 1.3 2000/11/29 17:40:32 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-6
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_6_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0xffff, 0xffff, 0xffff, 0x00a4, 0xffff, 0xffff, 0xffff,
00011     0xffff, 0xffff, 0xffff, 0xffff, 0x060c, 0x00ad, 0xffff, 0xffff,
00012     /* 0xb0 */
00013     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00014     0xffff, 0xffff, 0xffff, 0x061b, 0xffff, 0xffff, 0xffff, 0x061f,
00015     /* 0xc0 */
00016     0xffff, 0x0621, 0x0622, 0x0623, 0x0624, 0x0625, 0x0626, 0x0627,
00017     0x0628, 0x0629, 0x062a, 0x062b, 0x062c, 0x062d, 0x062e, 0x062f,
00018     /* 0xd0 */
00019     0x0630, 0x0631, 0x0632, 0x0633, 0x0634, 0x0635, 0x0636, 0x0637,
00020     0x0638, 0x0639, 0x063a, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00021     /* 0xe0 */
00022     0x0640, 0x0641, 0x0642, 0x0643, 0x0644, 0x0645, 0x0646, 0x0647,
00023     0x0648, 0x0649, 0x064a, 0x064b, 0x064c, 0x064d, 0x064e, 0x064f,
00024     /* 0xf0 */
00025     0x0650, 0x0651, 0x0652, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00026     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00027 };
00028
00029 static int
00030 iso8859_6_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c < 0xa0) {
00034         *pwc = (ucs4_t) c;
00035         return 1;
00036     }
00037     else {
00038         unsigned short wc = iso8859_6_2uni[c-0xa0];
00039         if (wc != 0xffff) {
00040             *pwc = (ucs4_t) wc;
00041             return 1;
00042         }
00043     }
00044     return RET_ILSEQ;
00045 }
00046 #endif /* NEED_TOWC */
00047
00048 #ifndef NEED_TOMB
00049 static const unsigned char iso8859_6_page00[16] = {
00050     0xa0, 0x00, 0x00, 0x00, 0xa4, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00051     0x00, 0x00, 0x00, 0x00, 0xad, 0x00, 0x00, /* 0xa8-0xaf */
00052 };
00053 static const unsigned char iso8859_6_page06[80] = {
00054     0x00, 0x00, 0x00, 0x00, 0xac, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00055     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00056     0x00, 0x00, 0x00, 0xbb, 0x00, 0x00, 0x00, 0xbf, /* 0x18-0x1f */
00057     0x00, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0x20-0x27 */
00058     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0x28-0x2f */
00059     0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, /* 0x30-0x37 */
00060     0xd8, 0xd9, 0xda, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00061     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0x40-0x47 */
00062     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0x48-0x4f */
00063     0xf0, 0xf1, 0xf2, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00064 };
00065
00066 static int
00067 iso8859_6_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00068 {

```

```

00069     (void)conv; (void)n;
00070     unsigned char c = 0;
00071     if (wc < 0x00a0) {
00072         *r = wc;
00073         return 1;
00074     }
00075     else if (wc >= 0x00a0 && wc < 0x00b0)
00076         c = iso8859_6_page00[wc-0x00a0];
00077     else if (wc >= 0x0608 && wc < 0x0658)
00078         c = iso8859_6_page06[wc-0x0608];
00079     if (c != 0) {
00080         *r = c;
00081         return 1;
00082     }
00083     return RET_ILSEQ;
00084 }
00085 #endif /* NEED_TOMB */

```

12.281 iso8859_7.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_7.h,v 1.3 2000/11/29 17:40:32 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-7
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_7_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0x2018, 0x2019, 0x00a3, 0xffff, 0xffff, 0x00a6, 0x00a7,
00011     0x00a8, 0x00a9, 0xffff, 0x00ab, 0x00ac, 0x00ad, 0xffff, 0x2015,
00012     /* 0xb0 */
00013     0x00b0, 0x00b1, 0x00b2, 0x00b3, 0x0384, 0x0385, 0x0386, 0x00b7,
00014     0x0388, 0x0389, 0x038a, 0x00bb, 0x038c, 0x00bd, 0x038e, 0x038f,
00015     /* 0xc0 */
00016     0x0390, 0x0391, 0x0392, 0x0393, 0x0394, 0x0395, 0x0396, 0x0397,
00017     0x0398, 0x0399, 0x039a, 0x039b, 0x039c, 0x039d, 0x039e, 0x039f,
00018     /* 0xd0 */
00019     0x03a0, 0x03a1, 0xffff, 0x03a3, 0x03a4, 0x03a5, 0x03a6, 0x03a7,
00020     0x03a8, 0x03a9, 0x03aa, 0x03ab, 0x03ac, 0x03ad, 0x03ae, 0x03af,
00021     /* 0xe0 */
00022     0x03b0, 0x03b1, 0x03b2, 0x03b3, 0x03b4, 0x03b5, 0x03b6, 0x03b7,
00023     0x03b8, 0x03b9, 0x03ba, 0x03bb, 0x03bc, 0x03bd, 0x03be, 0x03bf,
00024     /* 0xf0 */
00025     0x03c0, 0x03c1, 0x03c2, 0x03c3, 0x03c4, 0x03c5, 0x03c6, 0x03c7,
00026     0x03c8, 0x03c9, 0x03ca, 0x03cb, 0x03cc, 0x03cd, 0x03ce, 0xffff,
00027 };
00028
00029 static int
00030 iso8859_7_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c < 0xa0) {
00034         *pwc = (ucs4_t) c;
00035         return 1;
00036     }
00037     else {
00038         unsigned short wc = iso8859_7_2uni[c-0xa0];
00039         if (wc != 0xffff) {
00040             *pwc = (ucs4_t) wc;
00041             return 1;
00042         }
00043     }
00044     return RET_ILSEQ;
00045 }
00046 #endif /* NEED_TOWC */
00047
00048 #ifndef NEED_TOMB
00049 static const unsigned char iso8859_7_page00[32] = {
00050     0xa0, 0x00, 0x00, 0xa3, 0x00, 0x00, 0xa6, 0xa7, /* 0xa0-0xa7 */
00051     0xa8, 0xa9, 0xab, 0xac, 0xad, 0x00, 0x00, /* 0xa8-0xaf */
00052     0xb0, 0xb1, 0xb2, 0xb3, 0x00, 0x00, 0x00, 0xb7, /* 0xb0-0xb7 */
00053     0x00, 0x00, 0x00, 0xbb, 0x00, 0xbd, 0x00, 0x00, /* 0xb8-0xbf */
00054 };
00055 static const unsigned char iso8859_7_page03[80] = {
00056     0x00, 0x00, 0x00, 0x00, 0xb4, 0xb5, 0xb6, 0x00, /* 0x80-0x87 */
00057     0xb8, 0xb9, 0xba, 0x00, 0xbc, 0x00, 0xbe, 0xbf, /* 0x88-0x8f */
00058     0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0x90-0x97 */
00059     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0x98-0x9f */
00060     0xd0, 0xd1, 0x00, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, /* 0xa0-0xa7 */
00061     0xd8, 0xd9, 0xda, 0xdb, 0xdc, 0xdd, 0xde, 0xdf, /* 0xa8-0xaf */
00062     0xe0, 0xe1, 0xea, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0xb0-0xb7 */
00063     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0xb8-0xbf */
00064     0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0xc0-0xc7 */
00065     0xf8, 0xf9, 0xfa, 0xfb, 0xfc, 0xfd, 0xfe, 0x00, /* 0xc8-0xcf */

```

```

00066 };
00067 static const unsigned char iso8859_7_page20[16] = {
00068     0x00, 0x00, 0x00, 0x00, 0x00, 0xaf, 0x00, 0x00, /* 0x10-0x17 */
00069     0xa1, 0xa2, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00070 };
00071
00072 static int
00073 iso8859_7_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00074 {
00075     (void)conv; (void)n;
00076     unsigned char c = 0;
00077     if (wc < 0x00a0) {
00078         *r = wc;
00079         return 1;
00080     }
00081     else if (wc >= 0x00a0 && wc < 0x00c0)
00082         c = iso8859_7_page00[wc-0x00a0];
00083     else if (wc >= 0x0380 && wc < 0x03d0)
00084         c = iso8859_7_page03[wc-0x0380];
00085     else if (wc >= 0x2010 && wc < 0x2020)
00086         c = iso8859_7_page20[wc-0x2010];
00087     if (c != 0) {
00088         *r = c;
00089         return 1;
00090     }
00091     return RET_ILSEQ;
00092 }
00093 #endif /* NEED_TOMB */

```

12.282 iso8859_8.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_8.h,v 1.3 2000/11/29 17:40:32 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-8
00005  */
00006
00007 #ifndef NEED_TOWC
00008 static const unsigned short iso8859_8_2uni[96] = {
00009     /* 0xa0 */
00010     0x00a0, 0xffff, 0x00a2, 0x00a3, 0x00a4, 0x00a5, 0x00a6, 0x00a7,
00011     0x00a8, 0x00a9, 0x00d7, 0x00ab, 0x00ac, 0x00ad, 0x00ae, 0x00af,
00012     /* 0xb0 */
00013     0x00b0, 0x00b1, 0x00b2, 0x00b3, 0x00b4, 0x00b5, 0x00b6, 0x00b7,
00014     0x00b8, 0x00b9, 0x00f7, 0x00bb, 0x00bc, 0x00bd, 0x00be, 0xffff,
00015     /* 0xc0 */
00016     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00017     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00018     /* 0xd0 */
00019     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00020     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x2017,
00021     /* 0xe0 */
00022     0x05d0, 0x05d1, 0x05d2, 0x05d3, 0x05d4, 0x05d5, 0x05d6, 0x05d7,
00023     0x05d8, 0x05d9, 0x05da, 0x05db, 0x05dc, 0x05dd, 0x05de, 0x05df,
00024     /* 0xf0 */
00025     0x05e0, 0x05e1, 0x05e2, 0x05e3, 0x05e4, 0x05e5, 0x05e6, 0x05e7,
00026     0x05e8, 0x05e9, 0x05ea, 0xffff, 0xffff, 0x200e, 0x200f, 0xffff,
00027 };
00028
00029 static int
00030 iso8859_8_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00031 {
00032     unsigned char c = *s;
00033     if (c >= 0xa0) {
00034         unsigned short wc = iso8859_8_2uni[c-0xa0];
00035         if (wc != 0xffff) {
00036             *pwc = (ucs4_t) wc;
00037             return 1;
00038         }
00039     }
00040     else {
00041         *pwc = (ucs4_t) c;
00042         return 1;
00043     }
00044     return RET_ILSEQ;
00045 }
00046 #endif /* NEED_TOWC */
00047
00048 #ifndef NEED_TOMB
00049 static const unsigned char iso8859_8_page00[88] = {
00050     0xa0, 0x00, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, /* 0xa0-0xa7 */
00051     0xa8, 0xa9, 0xab, 0xac, 0xad, 0xae, 0xaf, /* 0xa8-0xaf */
00052     0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5, 0xb6, 0xb7, /* 0xb0-0xb7 */
00053     0xb8, 0xb9, 0x00, 0xbb, 0xbc, 0xbd, 0xbe, 0x00, /* 0xb8-0xbf */
00054     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */

```

```

00055     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00056     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xaa, /* 0xd0-0xd7 */
00057     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00058     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe0-0xe7 */
00059     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe8-0xef */
00060     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xba, /* 0xf0-0xf7 */
00061 };
00062 static const unsigned char iso8859_8_page05[32] = {
00063     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0xd0-0xd7 */
00064     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0xd8-0xdf */
00065     0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0xe0-0xe7 */
00066     0xf8, 0xf9, 0xfa, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe8-0xef */
00067 };
00068 static const unsigned char iso8859_8_page20[16] = {
00069     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfd, 0xfe, /* 0x08-0x0f */
00070     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xdf, /* 0x10-0x17 */
00071 };
00072
00073 static int
00074 iso8859_8_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00075 {
00076     (void)conv; (void)n;
00077     unsigned char c = 0;
00078     if (wc < 0x00a0) {
00079         *r = wc;
00080         return 1;
00081     }
00082     else if (wc >= 0x00a0 && wc < 0x00f8)
00083         c = iso8859_8_page00[wc-0x00a0];
00084     else if (wc >= 0x05d0 && wc < 0x05f0)
00085         c = iso8859_8_page05[wc-0x05d0];
00086     else if (wc >= 0x2008 && wc < 0x2018)
00087         c = iso8859_8_page20[wc-0x2008];
00088     if (c != 0) {
00089         *r = c;
00090         return 1;
00091     }
00092     return RET_ILSEQ;
00093 }
00094 #endif /* NEED_TOMB */

```

12.283 iso8859_9.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_9.h,v 1.3 2000/11/29 17:40:32 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-9
00005  */
00006
00007 #ifdef NEED_TOWC
00008 static const unsigned short iso8859_9_2uni[48] = {
00009     /* 0xd0 */
00010     0x011e, 0x00d1, 0x00d2, 0x00d3, 0x00d4, 0x00d5, 0x00d6, 0x00d7,
00011     0x00d8, 0x00d9, 0x00da, 0x00db, 0x00dc, 0x0130, 0x015e, 0x00df,
00012     /* 0xe0 */
00013     0x00e0, 0x00e1, 0x00e2, 0x00e3, 0x00e4, 0x00e5, 0x00e6, 0x00e7,
00014     0x00e8, 0x00e9, 0x00ea, 0x00eb, 0x00ec, 0x00ed, 0x00ee, 0x00ef,
00015     /* 0xf0 */
00016     0x011f, 0x00f1, 0x00f2, 0x00f3, 0x00f4, 0x00f5, 0x00f6, 0x00f7,
00017     0x00f8, 0x00f9, 0x00fa, 0x00fb, 0x00fc, 0x0131, 0x015f, 0x00ff,
00018 };
00019
00020 static int
00021 iso8859_9_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00022 {
00023     unsigned char c = *s;
00024     if (c >= 0xd0)
00025         *pwc = (ucs4_t) iso8859_9_2uni[c-0xd0];
00026     else
00027         *pwc = (ucs4_t) c;
00028     return 1;
00029 }
00030 #endif /* NEED_TOWC */
00031
00032 #ifdef NEED_TOMB
00033 static const unsigned char iso8859_9_page00[48] = {
00034     0x00, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, /* 0xd0-0xd7 */
00035     0xd8, 0xd9, 0xda, 0xdb, 0xdc, 0x00, 0x00, 0xdf, /* 0xd8-0xdf */
00036     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0xe0-0xe7 */
00037     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0xe8-0xef */
00038     0x00, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0xf0-0xf7 */
00039     0xf8, 0xf9, 0xfa, 0xfb, 0xfc, 0x00, 0x00, 0xff, /* 0xf8-0xff */
00040 };
00041 static const unsigned char iso8859_9_page01[72] = {
00042     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xd0, 0xf0, /* 0x18-0x1f */

```

```
00043 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00044 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00045 0xdd, 0xfd, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00046 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00047 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00048 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00049 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00050 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xde, 0xfe, /* 0x58-0x5f */
00051 };
00052
00053 static int
00054 iso8859_9_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00055 {
00056     (void)conv; (void)n;
00057     unsigned char c = 0;
00058     if (wc < 0x00d0) {
00059         *r = wc;
00060         return 1;
00061     }
00062     else if (wc >= 0x00d0 && wc < 0x0100)
00063         c = iso8859_9_page00[wc-0x00d0];
00064     else if (wc >= 0x0118 && wc < 0x0160)
00065         c = iso8859_9_page01[wc-0x0118];
00066     if (c != 0) {
00067         *r = c;
00068         return 1;
00069     }
00070     return RET_ILSEQ;
00071 }
00072 #endif /* NEED_TOMB */
```

12.284 iso8859_9e.h

```
00001 /* $XFree86: xc/lib/X11/lcUniConv/iso8859_9e.h,v 1.3 2000/11/28 16:10:28 dawes Exp $ */
00002
00003 /*
00004  * ISO-8859-9E
00005  */
00006
00007 static const unsigned short iso8859_9e_2uni[96] = {
00008     /* 0xa0 */
00009     0x00a0, 0x017d, 0x00a2, 0x00a3, 0x20ac, 0x00a5, 0x012c, 0x00a7,
00010     0x016c, 0x00a9, 0x01e6, 0x00ab, 0x014a, 0x00ad, 0x00ae, 0x01d1,
00011     /* 0xb0 */
00012     0x00b0, 0x017e, 0x00b2, 0x00b3, 0x00b4, 0x00b5, 0x012d, 0x00b7,
00013     0x016d, 0x00b9, 0x01e7, 0x00bb, 0x014b, 0x00bd, 0x0178, 0x01d2,
00014     /* 0xc0 */
00015     0x00c0, 0x00c1, 0x00c2, 0x00c3, 0x00c4, 0x00c5, 0x018f, 0x00c7,
00016     0x00c8, 0x00c9, 0x00ca, 0x00cb, 0x00cc, 0x00cd, 0x00ce, 0x00cf,
00017     /* 0xd0 */
00018     0x011e, 0x00d1, 0x00d2, 0x00d3, 0x00d4, 0x00d5, 0x00d6, 0x00dd,
00019     0x019f, 0x00d9, 0x00da, 0x00db, 0x00dc, 0x0130, 0x015e, 0x00df,
00020     /* 0xe0 */
00021     0x00e0, 0x00e1, 0x00e2, 0x00e3, 0x00e4, 0x00e5, 0x0259, 0x00e7,
00022     0x00e8, 0x00e9, 0x00ea, 0x00eb, 0x00ec, 0x00ed, 0x00ee, 0x00ef,
00023     /* 0xf0 */
00024     0x011f, 0x00f1, 0x00f2, 0x00f3, 0x00f4, 0x00f5, 0x00f6, 0x00fd,
00025     0x0275, 0x00f9, 0x00fa, 0x00fb, 0x00fc, 0x0131, 0x015f, 0x00ff,
00026 };
00027
00028 static int
00029 iso8859_9e_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00030 {
00031     unsigned char c = *s;
00032     if (c >= 0xa0)
00033         *pwc = (ucs4_t) iso8859_9e_2uni[c-0xa0];
00034     else
00035         *pwc = (ucs4_t) c;
00036     return 1;
00037 }
00038
00039 static const unsigned char iso8859_9e_page00[96] = {
00040     0xa0, 0x00, 0xa2, 0xa3, 0x00, 0xa5, 0x00, 0xa7, /* 0xa0-0xa7 */
00041     0x00, 0xa9, 0x00, 0xab, 0x00, 0xad, 0xae, 0x00, /* 0xa8-0xaf */
00042     0xb0, 0x00, 0xb2, 0xb3, 0xb4, 0xb5, 0x00, 0xb7, /* 0xb0-0xb7 */
00043     0x00, 0xb9, 0x00, 0xbb, 0x00, 0xbd, 0x00, 0x00, /* 0xb8-0xbf */
00044     0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0x00, 0xc7, /* 0xc0-0xc7 */
00045     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0xc8-0xcf */
00046     0x00, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0x00, /* 0xd0-0xd7 */
00047     0x00, 0xd9, 0xda, 0xdb, 0xdc, 0xdd, 0x00, 0xdf, /* 0xd8-0xdf */
00048     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0x00, 0xe7, /* 0xe0-0xe7 */
00049     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0xe8-0xef */
00050     0x00, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0x00, /* 0xf0-0xf7 */
00051     0x00, 0xf9, 0xfa, 0xfb, 0xfc, 0xfd, 0x00, 0xff, /* 0xf8-0xff */
00052 };
```

```

00053 static const unsigned char iso8859_9e_page01[136] = {
00054     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xd0, 0xf0, /* 0x18-0x1f */
00055     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00056     0x00, 0x00, 0x00, 0x00, 0xa6, 0xb6, 0x00, 0x00, /* 0x28-0x2f */
00057     0xdd, 0xfd, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00058     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00059     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00060     0x00, 0x00, 0xac, 0xbc, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00061     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00062     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xde, 0xfe, /* 0x58-0x5f */
00063     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00064     0x00, 0x00, 0x00, 0x00, 0x00, 0xa8, 0xb8, 0x00, 0x00, /* 0x68-0x6f */
00065     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00066     0xbe, 0x00, 0x00, 0x00, 0x00, 0xa1, 0xb1, 0x00, /* 0x78-0x7f */
00067     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00068     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc6, /* 0x88-0x8f */
00069     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00070     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xd8, /* 0x98-0x9f */
00071 };
00072 static const unsigned char iso8859_9e_page01_d[24] = {
00073     0x00, 0xaf, 0xbf, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00074     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00075     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xaa, 0xba, /* 0xe0-0xef */
00076 };
00077
00078 static int
00079 iso8859_9e_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00080 {
00081     unsigned char c = 0;
00082     if (wc < 0x00a0) {
00083         *r = wc;
00084         return 1;
00085     }
00086     else if (wc >= 0x00a0 && wc < 0x0100)
00087         c = iso8859_9e_page00[wc-0x00a0];
00088     else if (wc >= 0x0118 && wc < 0x01a0)
00089         c = iso8859_9e_page01[wc-0x0118];
00090     else if (wc >= 0x01d0 && wc < 0x01e8)
00091         c = iso8859_9e_page01_d[wc-0x01d0];
00092     else if (wc == 0x0259)
00093         c = 0xe6;
00094     else if (wc == 0x0275)
00095         c = 0xf8;
00096     else if (wc == 0x20ac)
00097         c = 0xa4;
00098     if (c != 0) {
00099         *r = c;
00100         return 1;
00101     }
00102     return RET_ILSEQ;
00103 }

```

12.285 jisx0201.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/jisx0201.h,v 1.3 2000/11/29 17:40:33 dawes Exp $ */
00002
00003 /*
00004  * JISX0201.1976-0
00005  */
00006 #ifndef NEED_TOWC
00007
00008 static int
00009 jisx0201_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00010 {
00011     unsigned char c = *s;
00012     if (c < 0x80) {
00013         if (c == 0x5c)
00014             *pwc = (ucs4_t) 0x00a5;
00015         else if (c == 0x7e)
00016             *pwc = (ucs4_t) 0x203e;
00017         else
00018             *pwc = (ucs4_t) c;
00019         return 1;
00020     } else {
00021         if (c >= 0xa1 && c < 0xe0) {
00022             *pwc = (ucs4_t) c + 0xfec0;
00023             return 1;
00024         }
00025     }
00026     return RET_ILSEQ;
00027 }
00028 #endif /* NEED_TOWC */
00029
00030 #ifndef NEED_TOMB
00031

```



```
00032 static int
00033 jisx0201_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00034 {
00035     (void)conv; (void)n;
00036     if (wc < 0x0080 && !(wc == 0x005c || wc == 0x007e)) {
00037         *r = wc;
00038         return 1;
00039     }
00040     if (wc == 0x00a5) {
00041         *r = 0x5c;
00042         return 1;
00043     }
00044     if (wc == 0x203e) {
00045         *r = 0x7e;
00046         return 1;
00047     }
00048     if (wc >= 0xff61 && wc < 0xffa0) {
00049         *r = wc - 0xfec0;
00050         return 1;
00051     }
00052     return RET_ILSEQ;
00053 }
00054 #endif /* NEED_TOMB */
```

12.286 jisx0208.h

```
00001 /* $XFree86: xc/lib/X11/lcUniConv/jisx0208.h,v 1.6 2003/05/27 22:26:31 tsi Exp $ */
00002
00003 /*
00004  * JISX0208.1990-0
00005  */
00006 #ifdef NEED_TOWC
00007
00008 static const unsigned short jisx0208_2uni_page21[690] = {
00009     /* 0x21 */
00010     0x3000, 0x3001, 0x3002, 0xff0c, 0xff0e, 0x30fb, 0xff1a, 0xff1b,
00011     0xff1f, 0xff01, 0x309b, 0x309c, 0x00b4, 0xff40, 0x00a8, 0xff3e,
00012     0xffe3, 0xff3f, 0x30fd, 0x30fe, 0x309d, 0x309e, 0x3003, 0x4edd,
00013     0x3005, 0x3006, 0x3007, 0x30fc, 0x2015, 0x2010, 0xff0f, 0xff3c,
00014     0x301c, 0x2016, 0xff5c, 0x2026, 0x2025, 0x2018, 0x2019, 0x201c,
00015     0x201d, 0xff08, 0xff09, 0x3014, 0x3015, 0xff3b, 0xff3d, 0xff5b,
00016     0xff5d, 0x3008, 0x3009, 0x300a, 0x300b, 0x300c, 0x300d, 0x300e,
00017     0x300f, 0x3010, 0x3011, 0xff0b, 0x2212, 0x00b1, 0x00d7, 0x00f7,
00018     0xff1d, 0x2260, 0xff1c, 0xff1e, 0x2266, 0x2267, 0x221e, 0x2234,
00019     0x2642, 0x2640, 0x00b0, 0x2032, 0x2033, 0x2103, 0xffe5, 0xff04,
00020     0x00a2, 0x00a3, 0xff05, 0xff03, 0xff06, 0xff0a, 0xff20, 0x00a7,
00021     0x2606, 0x2605, 0x25cb, 0x25cf, 0x25ce, 0x25c7,
00022     /* 0x22 */
00023     0x25c6, 0x25a1, 0x25a0, 0x25b3, 0x25b2, 0x25bd, 0x25bc, 0x203b,
00024     0x3012, 0x2192, 0x2190, 0x2191, 0x2193, 0x3013, 0xffff, 0xffff,
00025     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00026     0xffff, 0x2208, 0x220b, 0x2286, 0x2287, 0x2282, 0x2283, 0x222a,
00027     0x2229, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00028     0xffff, 0x2227, 0x2228, 0x00ac, 0x21d2, 0x21d4, 0x2200, 0x2203,
00029     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00030     0xffff, 0xffff, 0xffff, 0x2220, 0x22a5, 0x2312, 0x2202, 0x2207,
00031     0x2261, 0x2252, 0x226a, 0x226b, 0x221a, 0x223d, 0x221d, 0x2235,
00032     0x222b, 0x222c, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00033     0xffff, 0x212b, 0x2030, 0x266f, 0x266d, 0x266a, 0x2020, 0x2021,
00034     0x00b6, 0xffff, 0xffff, 0xffff, 0xffff, 0x25ef,
00035     /* 0x23 */
00036     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00037     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00038     0xff11, 0xff12, 0xff13, 0xff14, 0xff15, 0xff16, 0xff17, 0xff18,
00039     0xff19, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00040     0xff21, 0xff22, 0xff23, 0xff24, 0xff25, 0xff26, 0xff27, 0xff28,
00041     0xff29, 0xff2a, 0xff2b, 0xff2c, 0xff2d, 0xff2e, 0xff2f, 0xff30,
00042     0xff31, 0xff32, 0xff33, 0xff34, 0xff35, 0xff36, 0xff37, 0xff38,
00043     0xff39, 0xff3a, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00044     0xff41, 0xff42, 0xff43, 0xff44, 0xff45, 0xff46, 0xff47, 0xff48,
00045     0xff49, 0xff4a, 0xff4b, 0xff4c, 0xff4d, 0xff4e, 0xff4f, 0xff50,
00046     0xff51, 0xff52, 0xff53, 0xff54, 0xff55, 0xff56, 0xff57, 0xff58,
00047     0xff59, 0xff5a, 0xffff, 0xffff, 0xffff, 0xffff,
00048     /* 0x24 */
00049     0x3041, 0x3042, 0x3043, 0x3044, 0x3045, 0x3046, 0x3047, 0x3048,
00050     0x3049, 0x304a, 0x304b, 0x304c, 0x304d, 0x304e, 0x304f, 0x3050,
00051     0x3051, 0x3052, 0x3053, 0x3054, 0x3055, 0x3056, 0x3057, 0x3058,
00052     0x3059, 0x305a, 0x305b, 0x305c, 0x305d, 0x305e, 0x305f, 0x3060,
00053     0x3061, 0x3062, 0x3063, 0x3064, 0x3065, 0x3066, 0x3067, 0x3068,
00054     0x3069, 0x306a, 0x306b, 0x306c, 0x306d, 0x306e, 0x306f, 0x3070,
00055     0x3071, 0x3072, 0x3073, 0x3074, 0x3075, 0x3076, 0x3077, 0x3078,
00056     0x3079, 0x307a, 0x307b, 0x307c, 0x307d, 0x307e, 0x307f, 0x3080,
00057     0x3081, 0x3082, 0x3083, 0x3084, 0x3085, 0x3086, 0x3087, 0x3088,
00058     0x3089, 0x308a, 0x308b, 0x308c, 0x308d, 0x308e, 0x308f, 0x3090,
00059     0x3091, 0x3092, 0x3093, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
```

```

00060 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00061 /* 0x25 */
00062 0x30a1, 0x30a2, 0x30a3, 0x30a4, 0x30a5, 0x30a6, 0x30a7, 0x30a8,
00063 0x30a9, 0x30aa, 0x30ab, 0x30ac, 0x30ad, 0x30ae, 0x30af, 0x30b0,
00064 0x30b1, 0x30b2, 0x30b3, 0x30b4, 0x30b5, 0x30b6, 0x30b7, 0x30b8,
00065 0x30b9, 0x30ba, 0x30bb, 0x30bc, 0x30bd, 0x30be, 0x30bf, 0x30c0,
00066 0x30c1, 0x30c2, 0x30c3, 0x30c4, 0x30c5, 0x30c6, 0x30c7, 0x30c8,
00067 0x30c9, 0x30ca, 0x30cb, 0x30cc, 0x30cd, 0x30ce, 0x30cf, 0x30d0,
00068 0x30d1, 0x30d2, 0x30d3, 0x30d4, 0x30d5, 0x30d6, 0x30d7, 0x30d8,
00069 0x30d9, 0x30da, 0x30db, 0x30dc, 0x30dd, 0x30de, 0x30df, 0x30e0,
00070 0x30e1, 0x30e2, 0x30e3, 0x30e4, 0x30e5, 0x30e6, 0x30e7, 0x30e8,
00071 0x30e9, 0x30ea, 0x30eb, 0x30ec, 0x30ed, 0x30ee, 0x30ef, 0x30f0,
00072 0x30f1, 0x30f2, 0x30f3, 0x30f4, 0x30f5, 0x30f6, 0xffffd, 0xffffd,
00073 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00074 /* 0x26 */
00075 0x0391, 0x0392, 0x0393, 0x0394, 0x0395, 0x0396, 0x0397, 0x0398,
00076 0x0399, 0x039a, 0x039b, 0x039c, 0x039d, 0x039e, 0x039f, 0x03a0,
00077 0x03a1, 0x03a3, 0x03a4, 0x03a5, 0x03a6, 0x03a7, 0x03a8, 0x03a9,
00078 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00079 0x03b1, 0x03b2, 0x03b3, 0x03b4, 0x03b5, 0x03b6, 0x03b7, 0x03b8,
00080 0x03b9, 0x03ba, 0x03bb, 0x03bc, 0x03bd, 0x03be, 0x03bf, 0x03c0,
00081 0x03c1, 0x03c3, 0x03c4, 0x03c5, 0x03c6, 0x03c7, 0x03c8, 0x03c9,
00082 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00083 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00084 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00085 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00086 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00087 /* 0x27 */
00088 0x0410, 0x0411, 0x0412, 0x0413, 0x0414, 0x0415, 0x0401, 0x0416,
00089 0x0417, 0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e,
00090 0x041f, 0x0420, 0x0421, 0x0422, 0x0423, 0x0424, 0x0425, 0x0426,
00091 0x0427, 0x0428, 0x0429, 0x042a, 0x042b, 0x042c, 0x042d, 0x042e,
00092 0x042f, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00093 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00094 0x0430, 0x0431, 0x0432, 0x0433, 0x0434, 0x0435, 0x0436, 0x0437,
00095 0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e, 0x043f,
00096 0x0440, 0x0441, 0x0442, 0x0443, 0x0444, 0x0445, 0x0446, 0x0447,
00097 0x0448, 0x0449, 0x044a, 0x044b, 0x044c, 0x044d, 0x044e, 0x044f,
00098 0x0450, 0x0451, 0x0452, 0x0453, 0x0454, 0x0455, 0x0456, 0x0457,
00099 0x0458, 0x0459, 0x045a, 0x045b, 0x045c, 0x045d, 0x045e, 0x045f,
00100 /* 0x28 */
00101 0x2500, 0x2502, 0x250c, 0x2510, 0x2518, 0x2514, 0x251c, 0x252c,
00102 0x2524, 0x2534, 0x253c, 0x2501, 0x2503, 0x250f, 0x2513, 0x251b,
00103 0x2517, 0x2523, 0x2533, 0x252b, 0x253b, 0x254b, 0x2520, 0x252f,
00104 0x2528, 0x2537, 0x253f, 0x251d, 0x2530, 0x2525, 0x2538, 0x2542,
00105 };
00106 static const unsigned short jisx0208_2uni_page30[6398] = {
00107 /* 0x30 */
00108 0x4e9c, 0x5516, 0x5a03, 0x963f, 0x54c0, 0x611b, 0x6328, 0x59f6,
00109 0x9022, 0x8475, 0x831c, 0x7a50, 0x60aa, 0x63e1, 0x6e25, 0x65ed,
00110 0x8466, 0x82a6, 0x9bf5, 0x6893, 0x5727, 0x65a1, 0x6271, 0x5b9b,
00111 0x59d0, 0x867b, 0x98f4, 0x7d62, 0x7dbe, 0x9b8e, 0x6216, 0x7c9f,
00112 0x88b7, 0x5b89, 0x5eb5, 0x6309, 0x6697, 0x6848, 0x95c7, 0x978d,
00113 0x674f, 0x4ee5, 0x4f0a, 0x4f4d, 0x4f9d, 0x5049, 0x56f2, 0x5937,
00114 0x59d4, 0x5a01, 0x5c09, 0x60df, 0x610f, 0x6170, 0x6613, 0x6905,
00115 0x70ba, 0x754f, 0x7570, 0x79fb, 0x7dad, 0x7def, 0x80c3, 0x840e,
00116 0x8863, 0x8b02, 0x9055, 0x907a, 0x533b, 0x4e95, 0x4ea5, 0x57df,
00117 0x80b2, 0x90c1, 0x78ef, 0x4e00, 0x58f1, 0x6ea2, 0x9038, 0x7a32,
00118 0x8328, 0x828b, 0x9c2f, 0x5141, 0x5370, 0x54bd, 0x54e1, 0x56e0,
00119 0x59fb, 0x5f15, 0x98f2, 0x6deb, 0x80e4, 0x852d,
00120 /* 0x31 */
00121 0x9662, 0x9670, 0x96a0, 0x97fb, 0x540b, 0x53f3, 0x5b87, 0x70cf,
00122 0x7fbd, 0x8fc2, 0x96e8, 0x536f, 0x9d5c, 0x7aba, 0x4e11, 0x7893,
00123 0x81fc, 0x6e26, 0x5618, 0x5504, 0x6b1d, 0x851a, 0x9c3b, 0x59e5,
00124 0x53a9, 0x6d66, 0x74dc, 0x958f, 0x5642, 0x4e91, 0x904b, 0x96f2,
00125 0x834f, 0x990c, 0x53e1, 0x55b6, 0x5b30, 0x5f71, 0x6620, 0x66f3,
00126 0x6804, 0x6c38, 0x6cf3, 0x6d29, 0x745b, 0x76c8, 0x7a4e, 0x9834,
00127 0x82f1, 0x885b, 0x8a60, 0x92ed, 0x6db2, 0x75ab, 0x76ca, 0x99c5,
00128 0x60a6, 0x8b01, 0x8d8a, 0x95b2, 0x698e, 0x53ad, 0x5186, 0x5712,
00129 0x5830, 0x5944, 0x5bb4, 0x5ef6, 0x6028, 0x63a9, 0x63f4, 0x6cbf,
00130 0x6f14, 0x708e, 0x7114, 0x7159, 0x71d5, 0x733f, 0x7e01, 0x8276,
00131 0x82d1, 0x8597, 0x9060, 0x925b, 0x9d1b, 0x5869, 0x65bc, 0x6c5a,
00132 0x7525, 0x5f19, 0x592e, 0x5965, 0x5f80, 0x5fdc,
00133 /* 0x32 */
00134 0x62bc, 0x65fa, 0x6a2a, 0x6b27, 0x6bb4, 0x738b, 0x7fc1, 0x8956,
00135 0x9d2c, 0x9d0e, 0x9ec4, 0x5ca1, 0x6c96, 0x837b, 0x5104, 0x5c4b,
00136 0x61b6, 0x81c6, 0x6876, 0x7261, 0x4e59, 0x4ffa, 0x5378, 0x6069,
00137 0x6e29, 0x7a4f, 0x97f3, 0x4e0b, 0x5316, 0x4eee, 0x4f55, 0x4f3d,
00138 0x4fa1, 0x4f73, 0x52a0, 0x53ef, 0x5609, 0x590f, 0x5ac1, 0x5bb6,
00139 0x5be1, 0x79d1, 0x6687, 0x679c, 0x67b6, 0x6b4c, 0x6cb3, 0x70b6,
00140 0x73c2, 0x798d, 0x79be, 0x7a3c, 0x7b87, 0x82b1, 0x82db, 0x8304,
00141 0x8377, 0x83ef, 0x83d3, 0x8766, 0x8ab2, 0x5629, 0x8ca8, 0x8fe6,
00142 0x904e, 0x971e, 0x868a, 0x4fc4, 0x5ce8, 0x6211, 0x7259, 0x753b,
00143 0x81e5, 0x82bd, 0x86fe, 0x8cc0, 0x96c5, 0x9913, 0x99d5, 0x4ecb,
00144 0x4f1a, 0x89e3, 0x56de, 0x584a, 0x58ca, 0x5efb, 0x5feb, 0x602a,
00145 0x6094, 0x6062, 0x61d0, 0x6212, 0x62d0, 0x6539,
00146 /* 0x33 */

```

```
00147 0x9b41, 0x6666, 0x68b0, 0x6d77, 0x7070, 0x754c, 0x7686, 0x7d75,
00148 0x82a5, 0x87f9, 0x958b, 0x968e, 0x8c9d, 0x51f1, 0x52be, 0x5916,
00149 0x54b3, 0x5bb3, 0x5bb3, 0x5d16, 0x6168, 0x6982, 0x6daf, 0x788d, 0x84bc,
00150 0x8857, 0x8a72, 0x93a7, 0x9ab8, 0x6d6c, 0x99a8, 0x86d9, 0x57a3,
00151 0x67ff, 0x86ce, 0x920e, 0x5283, 0x5687, 0x5404, 0x5ed3, 0x62e1,
00152 0x64b9, 0x683c, 0x6838, 0x6bbb, 0x7372, 0x78ba, 0x7a6b, 0x899a,
00153 0x89d2, 0x8d6b, 0x8f03, 0x90ed, 0x95a3, 0x9694, 0x9769, 0x5b66,
00154 0x5cb3, 0x697d, 0x984d, 0x984e, 0x639b, 0x7b20, 0x6a2b, 0x6a7f,
00155 0x68b6, 0x9c0d, 0x6f5f, 0x5272, 0x559d, 0x6070, 0x62ec, 0x6d3b,
00156 0x6e07, 0x6ed1, 0x845b, 0x8910, 0x8f44, 0x4e14, 0x9c39, 0x53f6,
00157 0x691b, 0x6a3a, 0x9784, 0x682a, 0x515c, 0x7ac3, 0x84b2, 0x91dc,
00158 0x938c, 0x565b, 0x9d28, 0x6822, 0x8305, 0x8431,
00159 /* 0x34 */
00160 0x7ca5, 0x5208, 0x82c5, 0x74e6, 0x4e7e, 0x4f83, 0x51a0, 0x5bd2,
00161 0x520a, 0x52d8, 0x52e7, 0x5dfb, 0x559a, 0x582a, 0x59e6, 0x5b8c,
00162 0x5b98, 0x5bdb, 0x5e72, 0x5e79, 0x60a3, 0x611f, 0x6163, 0x61be,
00163 0x63db, 0x6562, 0x67d1, 0x6853, 0x68fa, 0x6b3e, 0x6b53, 0x6c57,
00164 0x6f22, 0x6f97, 0x6f45, 0x74b0, 0x7518, 0x76e3, 0x770b, 0x7aff,
00165 0x7ba1, 0x7c21, 0x7de9, 0x7f36, 0x7ff0, 0x809d, 0x8266, 0x839e,
00166 0x89b3, 0x8acc, 0x8cab, 0x9084, 0x9451, 0x9593, 0x9591, 0x95a2,
00167 0x9665, 0x97d3, 0x9928, 0x8218, 0x4e38, 0x542b, 0x5cb8, 0x5dcc,
00168 0x73a9, 0x764c, 0x773c, 0x5ca9, 0x7feb, 0x8d0b, 0x96c1, 0x9811,
00169 0x9854, 0x9858, 0x4f01, 0x4f0e, 0x5371, 0x559c, 0x5668, 0x57fa,
00170 0x5947, 0x5b09, 0x5bc4, 0x5c90, 0x5e0c, 0x5e7e, 0x5fcc, 0x63ee,
00171 0x673a, 0x65d7, 0x65e2, 0x671f, 0x68cb, 0x68c4,
00172 /* 0x35 */
00173 0x6a5f, 0x5e30, 0x6bc5, 0x6c17, 0x6c7d, 0x757f, 0x7948, 0x5b63,
00174 0x7a00, 0x7d00, 0x5fbd, 0x898f, 0x8a18, 0x8cb4, 0x8d77, 0x8ecc,
00175 0x8f1d, 0x98e2, 0x9a0e, 0x9b3c, 0x4e80, 0x507d, 0x5100, 0x5993,
00176 0x5b9c, 0x622f, 0x6280, 0x64ec, 0x6b3a, 0x72a0, 0x7591, 0x7947,
00177 0x7fa9, 0x87fb, 0x8abc, 0x8b70, 0x63ac, 0x83ca, 0x97a0, 0x5409,
00178 0x5403, 0x55ab, 0x6854, 0x6a58, 0x8a70, 0x7827, 0x6775, 0x9ecd,
00179 0x5374, 0x5ba2, 0x811a, 0x8650, 0x9006, 0x4e18, 0x4e45, 0x4ec7,
00180 0x4f11, 0x53ca, 0x5438, 0x5bae, 0x5f13, 0x6025, 0x6551, 0x673d,
00181 0x6c42, 0x6c72, 0x6ce3, 0x7078, 0x7403, 0x7a76, 0x7aae, 0x7b08,
00182 0x7d1a, 0x7cfe, 0x7d66, 0x65e7, 0x725b, 0x53bb, 0x5c45, 0x5de8,
00183 0x62d2, 0x62e0, 0x6319, 0x6e20, 0x865a, 0x8a31, 0x8ddd, 0x92f8,
00184 0x6f01, 0x79a6, 0x9b5a, 0x4ea8, 0x4eab, 0x4eac,
00185 /* 0x36 */
00186 0x4f9b, 0x4fa0, 0x50d1, 0x5147, 0x7af6, 0x5171, 0x51f6, 0x5354,
00187 0x5321, 0x537f, 0x53eb, 0x55ac, 0x5883, 0x5ce1, 0x5f37, 0x5f4a,
00188 0x602f, 0x6050, 0x606d, 0x631f, 0x6559, 0x6a4b, 0x6cc1, 0x72c2,
00189 0x72ed, 0x77ef, 0x80f8, 0x8105, 0x8208, 0x854e, 0x90f7, 0x93e1,
00190 0x97ff, 0x9957, 0x9a5a, 0x4ef0, 0x51dd, 0x5c2d, 0x6681, 0x696d,
00191 0x5c40, 0x66f2, 0x6975, 0x7389, 0x6850, 0x7c81, 0x50c5, 0x52e4,
00192 0x5747, 0x5dfe, 0x9326, 0x65a4, 0x6b23, 0x6b3d, 0x7434, 0x7981,
00193 0x79bd, 0x7b4b, 0x7dca, 0x82b9, 0x83cc, 0x887f, 0x895f, 0x8b39,
00194 0x8fd1, 0x91d1, 0x541f, 0x9280, 0x4e5d, 0x5036, 0x53e5, 0x533a,
00195 0x72d7, 0x7396, 0x77e9, 0x82e6, 0x8eaf, 0x99c6, 0x99c8, 0x99d2,
00196 0x5177, 0x611a, 0x865e, 0x55b0, 0x7a7a, 0x5076, 0x5bd3, 0x9047,
00197 0x9685, 0x4e32, 0x6adb, 0x91e7, 0x5c51, 0x5c48,
00198 /* 0x37 */
00199 0x6398, 0x7a9f, 0x6c93, 0x9774, 0x8f61, 0x7aaa, 0x718a, 0x9688,
00200 0x7c82, 0x6817, 0x7e70, 0x6851, 0x936c, 0x52f2, 0x541b, 0x85ab,
00201 0x8a13, 0x7fa4, 0x8ecd, 0x90e1, 0x5366, 0x8888, 0x7941, 0x4fc2,
00202 0x50be, 0x5211, 0x5144, 0x5553, 0x572d, 0x73ea, 0x578b, 0x5951,
00203 0x5f62, 0x5f84, 0x6075, 0x6176, 0x6167, 0x61a9, 0x63b2, 0x643a,
00204 0x656c, 0x666f, 0x6842, 0x6e13, 0x7566, 0x7a3d, 0x7cfe, 0x7d4c,
00205 0x7d99, 0x7e4b, 0x7f6b, 0x830e, 0x834a, 0x86cd, 0x8a08, 0x8a63,
00206 0x8b66, 0x8efd, 0x981a, 0x9d8f, 0x82b8, 0x8fce, 0x9be8, 0x5287,
00207 0x621f, 0x6483, 0x6fc0, 0x9699, 0x6841, 0x5091, 0x6b20, 0x6c7a,
00208 0x6f54, 0x7a74, 0x7d50, 0x8840, 0x8a23, 0x6708, 0x4ef6, 0x5039,
00209 0x5026, 0x5065, 0x517c, 0x5238, 0x5263, 0x55a7, 0x570f, 0x5805,
00210 0x5acc, 0x5efa, 0x61b2, 0x61f8, 0x62f3, 0x6372,
00211 /* 0x38 */
00212 0x691c, 0x6a29, 0x727d, 0x72ac, 0x732e, 0x7814, 0x786f, 0x7d79,
00213 0x770c, 0x80a9, 0x898b, 0x8b19, 0x8ce2, 0x8ed2, 0x9063, 0x9375,
00214 0x967a, 0x9855, 0x9a13, 0x9e78, 0x5143, 0x539f, 0x53b3, 0x5e7b,
00215 0x5f26, 0x6e1b, 0x6e90, 0x7384, 0x73fe, 0x7d43, 0x8237, 0x8a00,
00216 0x8afa, 0x9650, 0x4e4e, 0x500b, 0x53e4, 0x547c, 0x56fa, 0x59d1,
00217 0x5b64, 0x5df1, 0x5eab, 0x5f27, 0x6238, 0x6545, 0x67af, 0x6e56,
00218 0x72d0, 0x7cca, 0x88b4, 0x80a1, 0x80e1, 0x83f0, 0x864e, 0x8a87,
00219 0x8de8, 0x9237, 0x96c7, 0x9867, 0x9f13, 0x4e94, 0x4e92, 0x4f0d,
00220 0x5348, 0x5449, 0x543e, 0x5a2f, 0x5f8c, 0x5fa1, 0x609f, 0x68a7,
00221 0x6a8e, 0x745a, 0x7881, 0x8a9e, 0x8aa4, 0x8b77, 0x9190, 0x4e5e,
00222 0x9bc9, 0x4ea4, 0x4f7c, 0x4faf, 0x5019, 0x5016, 0x5149, 0x516c,
00223 0x529f, 0x52b9, 0x52fe, 0x539a, 0x53e3, 0x5411,
00224 /* 0x39 */
00225 0x540e, 0x5589, 0x5751, 0x57a2, 0x597d, 0x5b54, 0x5b5d, 0x5b8f,
00226 0x5de5, 0x5de7, 0x5df7, 0x5e78, 0x5e83, 0x5e9a, 0x5eb7, 0x5f18,
00227 0x6052, 0x614c, 0x6297, 0x62d8, 0x63a7, 0x653b, 0x6602, 0x6643,
00228 0x66f4, 0x676d, 0x6821, 0x6897, 0x69cb, 0x6c5f, 0x6d2a, 0x6d69,
00229 0x6e2f, 0x6e9d, 0x7532, 0x7687, 0x786c, 0x7a3f, 0x7ce0, 0x7d05,
00230 0x7d18, 0x7d5e, 0x7db1, 0x8015, 0x8003, 0x80af, 0x80b1, 0x8154,
00231 0x818f, 0x822a, 0x8352, 0x884c, 0x8861, 0x8b1b, 0x8ca2, 0x8cfc,
00232 0x90ca, 0x9175, 0x9271, 0x783f, 0x92fc, 0x95a4, 0x964d, 0x9805,
00233 0x9999, 0x9ad8, 0x9d3b, 0x525b, 0x52ab, 0x53f7, 0x5408, 0x58d5,
```

```

00234 0x62f7, 0x6fe0, 0x8c6a, 0x8f5f, 0x9eb9, 0x514b, 0x523b, 0x544a,
00235 0x56fd, 0x7a40, 0x9177, 0x9d60, 0x9ed2, 0x7344, 0x6f09, 0x8170,
00236 0x7511, 0x5ffd, 0x60da, 0x9aa8, 0x72db, 0x8fbc,
00237 /* 0x3a */
00238 0x6b64, 0x9803, 0x4eca, 0x56f0, 0x5764, 0x58be, 0x5a5a, 0x6068,
00239 0x61c7, 0x660f, 0x6606, 0x6839, 0x68b1, 0x6df7, 0x75d5, 0x7d3a,
00240 0x826e, 0x9b42, 0x4e9b, 0x4f50, 0x53c9, 0x5506, 0x5d6f, 0x5de6,
00241 0x5dee, 0x67fb, 0x6c99, 0x7473, 0x7802, 0x8a50, 0x9396, 0x88df,
00242 0x5750, 0x5ea7, 0x632b, 0x50b5, 0x50ac, 0x518d, 0x6700, 0x54c9,
00243 0x585e, 0x59bb, 0x5bb0, 0x5f69, 0x624d, 0x63a1, 0x683d, 0x6b73,
00244 0x6e08, 0x707d, 0x91c7, 0x7280, 0x7815, 0x7826, 0x796d, 0x658e,
00245 0x7d30, 0x83dc, 0x88c1, 0x8f09, 0x969b, 0x5264, 0x5728, 0x6750,
00246 0x7f6a, 0x8ca1, 0x51b4, 0x5742, 0x962a, 0x583a, 0x698a, 0x80b4,
00247 0x54b2, 0x5d0e, 0x57fc, 0x7895, 0x9dfa, 0x4f5c, 0x524a, 0x548b,
00248 0x643e, 0x6628, 0x6714, 0x67f5, 0x7a84, 0x7b56, 0x7d22, 0x932f,
00249 0x685c, 0x9bad, 0x7b39, 0x5319, 0x518a, 0x5237,
00250 /* 0x3b */
00251 0x5bdf, 0x62f6, 0x64ae, 0x64e6, 0x672d, 0x6bba, 0x85a9, 0x96d1,
00252 0x7690, 0x9bd6, 0x634c, 0x9306, 0x9bab, 0x76bf, 0x6652, 0x4e09,
00253 0x5098, 0x53c2, 0x5c71, 0x60e8, 0x6492, 0x6563, 0x685f, 0x71e6,
00254 0x73ca, 0x7523, 0x7b97, 0x7e82, 0x8695, 0x8b83, 0x8cdb, 0x9178,
00255 0x9910, 0x65ac, 0x66ab, 0x6b8b, 0x4ed5, 0x4ed4, 0x4f3a, 0x4f7f,
00256 0x523a, 0x53f8, 0x53f2, 0x55e3, 0x56db, 0x58eb, 0x59cb, 0x59c9,
00257 0x59ff, 0x5b50, 0x5c4d, 0x5e02, 0x5e2b, 0x5fd7, 0x601d, 0x6307,
00258 0x652f, 0x5b5c, 0x65af, 0x65bd, 0x65e8, 0x679d, 0x6b62, 0x6b7b,
00259 0x6c0f, 0x7345, 0x7949, 0x79c1, 0x7cf8, 0x7d19, 0x7d2b, 0x80a2,
00260 0x8102, 0x81f3, 0x8996, 0x8a5e, 0x8a69, 0x8a66, 0x8a8c, 0x8aee,
00261 0x8cc7, 0x8cdc, 0x96cc, 0x98fc, 0x6b6f, 0x4e8b, 0x4f3c, 0x4f8d,
00262 0x5150, 0x5b57, 0x5bfa, 0x6148, 0x6301, 0x6642,
00263 /* 0x3c */
00264 0x6b21, 0x6ecb, 0x6cbb, 0x723e, 0x74bd, 0x75d4, 0x78c1, 0x793a,
00265 0x800c, 0x8033, 0x81ea, 0x8494, 0x8f9e, 0x6c50, 0x9e7f, 0x5f0f,
00266 0x8b58, 0x9d2b, 0x7afa, 0x8ef8, 0x5b8d, 0x96eb, 0x4e03, 0x53f1,
00267 0x57f7, 0x5931, 0x5ac9, 0x5ba4, 0x6089, 0x6e7f, 0x6f06, 0x75be,
00268 0x8cea, 0x5b9f, 0x8500, 0x7be0, 0x5072, 0x67f4, 0x829d, 0x5c61,
00269 0x854a, 0x7e1e, 0x820e, 0x5199, 0x5c04, 0x6368, 0x8d66, 0x659c,
00270 0x716e, 0x793e, 0x7d17, 0x8005, 0x8b1d, 0x8eca, 0x906e, 0x86c7,
00271 0x90aa, 0x501f, 0x52fa, 0x5c3a, 0x6753, 0x707c, 0x7235, 0x914c,
00272 0x91c8, 0x932b, 0x82e5, 0x5bc2, 0x5f31, 0x60f9, 0x4e3b, 0x53d6,
00273 0x5b88, 0x624b, 0x6731, 0x6b8a, 0x72e9, 0x73e0, 0x7a2e, 0x816b,
00274 0x8da3, 0x9152, 0x9996, 0x5112, 0x53d7, 0x546a, 0x5bff, 0x6388,
00275 0x6a39, 0x7dac, 0x9700, 0x56da, 0x53ce, 0x5468,
00276 /* 0x3d */
00277 0x5b97, 0x5c31, 0x5dde, 0x4fee, 0x6101, 0x62fe, 0x6d32, 0x79c0,
00278 0x79cb, 0x7d42, 0x7e4d, 0x7fd2, 0x81ed, 0x821f, 0x8490, 0x8846,
00279 0x8972, 0x8b90, 0x8e74, 0x8f2f, 0x9031, 0x914b, 0x916c, 0x96c6,
00280 0x919c, 0x4ec0, 0x4f4f, 0x5145, 0x5341, 0x5f93, 0x620e, 0x67d4,
00281 0x6c41, 0x6e0b, 0x7363, 0x7e26, 0x91cd, 0x9283, 0x53d4, 0x5919,
00282 0x5bbf, 0x6dd1, 0x795d, 0x7e2e, 0x7c9b, 0x587e, 0x719f, 0x51fa,
00283 0x8853, 0x8ff0, 0x4fca, 0x5cfb, 0x6625, 0x77ac, 0x7ae3, 0x821c,
00284 0x99ff, 0x51c6, 0x5faa, 0x65ec, 0x696f, 0x6b89, 0x6df3, 0x6e96,
00285 0x6f64, 0x76fe, 0x7d14, 0x5de1, 0x9075, 0x9187, 0x9806, 0x51e6,
00286 0x521d, 0x6240, 0x6691, 0x66d9, 0x6e1a, 0x5eb6, 0x7dd2, 0x7f72,
00287 0x66f8, 0x85af, 0x85f7, 0x8af8, 0x52a9, 0x53d9, 0x5973, 0x5e8f,
00288 0x5f90, 0x6055, 0x92e4, 0x9664, 0x50b7, 0x511f,
00289 /* 0x3e */
00290 0x52dd, 0x5320, 0x5347, 0x53ec, 0x54e8, 0x5546, 0x5531, 0x5617,
00291 0x5968, 0x59be, 0x5a3c, 0x5bb5, 0x5c06, 0x5c0f, 0x5c11, 0x5c1a,
00292 0x5e84, 0x5e8a, 0x5ee0, 0x5f70, 0x627f, 0x6284, 0x62db, 0x638c,
00293 0x6377, 0x6607, 0x660c, 0x662d, 0x6676, 0x677e, 0x68a2, 0x6a1f,
00294 0x6a35, 0x6cbc, 0x6d88, 0x6e09, 0x6e58, 0x713c, 0x7126, 0x7167,
00295 0x75c7, 0x7701, 0x785d, 0x7901, 0x7965, 0x79f0, 0x7ae0, 0x7b11,
00296 0x7ca7, 0x7d39, 0x8096, 0x83d6, 0x848b, 0x8549, 0x885d, 0x88f3,
00297 0x8a1f, 0x8a3c, 0x8a54, 0x8a73, 0x8c61, 0x8cde, 0x91a4, 0x9266,
00298 0x937e, 0x9418, 0x969c, 0x9798, 0x4e0a, 0x4e08, 0x4e1e, 0x4e57,
00299 0x5197, 0x5270, 0x57ce, 0x5834, 0x58cc, 0x5b22, 0x5e38, 0x60c5,
00300 0x64fe, 0x6761, 0x6756, 0x6d44, 0x72b6, 0x7573, 0x7a63, 0x84b8,
00301 0x8b72, 0x91b8, 0x9320, 0x5631, 0x57f4, 0x98fe,
00302 /* 0x3f */
00303 0x62ed, 0x690d, 0x6b96, 0x71ed, 0x7e54, 0x8077, 0x8272, 0x89e6,
00304 0x98df, 0x8755, 0x8fb1, 0x5c3b, 0x4f38, 0x4fe1, 0x4fb5, 0x5507,
00305 0x5a20, 0x5bdd, 0x5be9, 0x5fc3, 0x614e, 0x632f, 0x65b0, 0x664b,
00306 0x68ee, 0x699b, 0x6d78, 0x6df1, 0x7533, 0x75b9, 0x771f, 0x795e,
00307 0x79e6, 0x7d33, 0x81e3, 0x82af, 0x85aa, 0x89aa, 0x8a3a, 0x8eab,
00308 0x8f9b, 0x9032, 0x91dd, 0x9707, 0x4eba, 0x4ec1, 0x5203, 0x5875,
00309 0x58ec, 0x5c0b, 0x751a, 0x5c3d, 0x814e, 0x8a0a, 0x8fc5, 0x9663,
00310 0x976d, 0x7b25, 0x8acf, 0x9808, 0x9162, 0x56f3, 0x53a8, 0x9017,
00311 0x5439, 0x5782, 0x5e25, 0x63a8, 0x6c34, 0x708a, 0x7761, 0x7c8b,
00312 0x7fe0, 0x8870, 0x9042, 0x9154, 0x9310, 0x9318, 0x968f, 0x745e,
00313 0x9ac4, 0x5d07, 0x5d69, 0x6570, 0x67a2, 0x8da8, 0x96db, 0x636e,
00314 0x6749, 0x6919, 0x83c5, 0x9817, 0x96c0, 0x88fe,
00315 /* 0x40 */
00316 0x6f84, 0x647a, 0x5bf8, 0x4e16, 0x702c, 0x755d, 0x662f, 0x51c4,
00317 0x5236, 0x52e2, 0x59d3, 0x5f81, 0x6027, 0x6210, 0x653f, 0x6574,
00318 0x661f, 0x6674, 0x68f2, 0x6816, 0x6b63, 0x6e05, 0x7272, 0x751f,
00319 0x76db, 0x7cbe, 0x8056, 0x58f0, 0x88fd, 0x897f, 0x8aa0, 0x8a93,
00320 0x8acb, 0x901d, 0x9192, 0x9752, 0x9759, 0x6589, 0x7a0e, 0x8106,

```

```
00321 0x96bb, 0x5e2d, 0x60dc, 0x621a, 0x65a5, 0x6614, 0x6790, 0x77f3,
00322 0x7a4d, 0x7c4d, 0x7e3e, 0x810a, 0x8cac, 0x8d64, 0x8de1, 0x8e5f,
00323 0x78a9, 0x5207, 0x62d9, 0x63a5, 0x6442, 0x6298, 0x8a2d, 0x7a83,
00324 0x7bc0, 0x8aac, 0x96ea, 0x7d76, 0x820c, 0x8749, 0x4ed9, 0x5148,
00325 0x5343, 0x5360, 0x5ba3, 0x5c02, 0x5c16, 0x5ddd, 0x6226, 0x6247,
00326 0x64b0, 0x6813, 0x6834, 0x6cc9, 0x6d45, 0x6d17, 0x67d3, 0x6f5c,
00327 0x714e, 0x717d, 0x65cb, 0x7a7f, 0x7bad, 0x7dda,
00328 /* 0x41 */
00329 0x7e4a, 0x7fa8, 0x817a, 0x821b, 0x8239, 0x85a6, 0x8a6e, 0x8cce,
00330 0x8df5, 0x9078, 0x9077, 0x92ad, 0x9291, 0x9583, 0x9bae, 0x524d,
00331 0x5584, 0x6f38, 0x7136, 0x5168, 0x7985, 0x7e55, 0x81b3, 0x7cce,
00332 0x564c, 0x5851, 0x5ca8, 0x63aa, 0x66fe, 0x66fd, 0x695a, 0x72d9,
00333 0x758f, 0x758e, 0x790e, 0x7956, 0x79df, 0x7c97, 0x7d20, 0x7d44,
00334 0x8607, 0x8a34, 0x963b, 0x9061, 0x9f20, 0x50e7, 0x5275, 0x53cc,
00335 0x53e2, 0x5009, 0x55aa, 0x58ee, 0x594f, 0x723d, 0x5b8b, 0x5c64,
00336 0x531d, 0x60e3, 0x60f3, 0x635c, 0x6383, 0x633f, 0x63bb, 0x64cd,
00337 0x65e9, 0x66f9, 0x5de3, 0x69cd, 0x69fd, 0x6f15, 0x71e5, 0x4e89,
00338 0x75e9, 0x76f8, 0x7a93, 0x7cdf, 0x7dcf, 0x7d9c, 0x8061, 0x8349,
00339 0x8358, 0x846c, 0x84bc, 0x85fb, 0x88c5, 0x8d70, 0x9001, 0x906d,
00340 0x9397, 0x971c, 0x9a12, 0x50cf, 0x5897, 0x618e,
00341 /* 0x42 */
00342 0x81d3, 0x8535, 0x8d08, 0x9020, 0x4fc3, 0x5074, 0x5247, 0x5373,
00343 0x606f, 0x6349, 0x675f, 0x6e2c, 0x8db3, 0x901f, 0x4fd7, 0x5c5e,
00344 0x8cca, 0x65cf, 0x7d9a, 0x5352, 0x8896, 0x5176, 0x63c3, 0x5b58,
00345 0x5b6b, 0x5c0a, 0x640d, 0x6751, 0x905c, 0x4ed6, 0x591a, 0x592a,
00346 0x6c70, 0x8a51, 0x553e, 0x5815, 0x59a5, 0x60f0, 0x6253, 0x67c1,
00347 0x8235, 0x6955, 0x9640, 0x99c4, 0x9a28, 0x4f53, 0x5806, 0x5bfe,
00348 0x8010, 0x5cb1, 0x5e2f, 0x5f85, 0x6020, 0x614b, 0x6234, 0x66ff,
00349 0x6cf0, 0x6ede, 0x80ce, 0x817f, 0x82d4, 0x888b, 0x8cb8, 0x9000,
00350 0x902e, 0x968a, 0x9edb, 0x9bdb, 0x4ee3, 0x53f0, 0x5927, 0x7b2c,
00351 0x918d, 0x984c, 0x9df9, 0x6edd, 0x7027, 0x5353, 0x5544, 0x5b85,
00352 0x6258, 0x629e, 0x62d3, 0x6ca2, 0x6fef, 0x7422, 0x8a17, 0x9438,
00353 0x6fc1, 0x8afe, 0x8338, 0x51e7, 0x86f8, 0x53ea,
00354 /* 0x43 */
00355 0x53e9, 0x4f46, 0x9054, 0x8fb0, 0x596a, 0x8131, 0x5dfd, 0x7aea,
00356 0x8fbf, 0x68da, 0x8c37, 0x72f8, 0x9c48, 0x6a3d, 0x8ab0, 0x4e39,
00357 0x5358, 0x5606, 0x5766, 0x62c5, 0x63a2, 0x65e6, 0x6b4e, 0x6de1,
00358 0x6e5b, 0x70ad, 0x77ed, 0x7aef, 0x7baa, 0x7dbb, 0x803d, 0x80c6,
00359 0x86cb, 0x8a95, 0x935b, 0x56e3, 0x58c7, 0x5f3e, 0x65ad, 0x6696,
00360 0x6a80, 0x6bb5, 0x7537, 0x8ac7, 0x5024, 0x77e5, 0x5730, 0x5f1b,
00361 0x6065, 0x667a, 0x6c60, 0x75f4, 0x7a1a, 0x7f6e, 0x81f4, 0x8718,
00362 0x9045, 0x99b3, 0x7bc9, 0x755c, 0x7af9, 0x7b51, 0x84c4, 0x9010,
00363 0x79e9, 0x7a92, 0x8336, 0x5ae1, 0x7740, 0x4e2d, 0x4ef2, 0x5b99,
00364 0x5fe0, 0x62bd, 0x663c, 0x67f1, 0x6ce8, 0x866b, 0x8877, 0x8a3b,
00365 0x914e, 0x92f3, 0x99d0, 0x6a17, 0x7026, 0x732a, 0x82e7, 0x8457,
00366 0x8caf, 0x4e01, 0x5146, 0x51cb, 0x558b, 0x5bf5,
00367 /* 0x44 */
00368 0x5e16, 0x5e33, 0x5e81, 0x5f14, 0x5f35, 0x5f6b, 0x5fb4, 0x61f2,
00369 0x6311, 0x66a2, 0x671d, 0x6f6e, 0x7252, 0x753a, 0x773a, 0x8074,
00370 0x8139, 0x8178, 0x8776, 0x8abf, 0x8adc, 0x8d85, 0x8df3, 0x929a,
00371 0x9577, 0x9802, 0x9ce5, 0x52c5, 0x6357, 0x76f4, 0x6715, 0x6c88,
00372 0x73cd, 0x8cc3, 0x93ae, 0x9673, 0x6d25, 0x589c, 0x690e, 0x69cc,
00373 0x8ffd, 0x939a, 0x75db, 0x901a, 0x585a, 0x6802, 0x63b4, 0x69fb,
00374 0x4f43, 0x6f2c, 0x67d8, 0x8fbb, 0x8526, 0x7db4, 0x9354, 0x693f,
00375 0x6f70, 0x576a, 0x58f7, 0x5b2c, 0x7d2c, 0x722a, 0x540a, 0x91e3,
00376 0x9db4, 0x4ead, 0x4f4e, 0x505c, 0x5075, 0x5243, 0x8c9e, 0x5448,
00377 0x5824, 0x5b9a, 0x5e1d, 0x5e95, 0x5ead, 0x5ef7, 0x5f1f, 0x608c,
00378 0x62b5, 0x633a, 0x63d0, 0x68af, 0x6c40, 0x7887, 0x798e, 0x7a0b,
00379 0x7de0, 0x8247, 0x8a02, 0x8ae6, 0x8e44, 0x9013,
00380 /* 0x45 */
00381 0x90b8, 0x912d, 0x91d8, 0x9f0e, 0x6ce5, 0x6458, 0x64e2, 0x6575,
00382 0x6ef4, 0x7684, 0x7b1b, 0x9069, 0x93d1, 0x6eba, 0x54f2, 0x5fb9,
00383 0x64a4, 0x8f4d, 0x8fed, 0x9244, 0x5178, 0x586b, 0x5929, 0x5c55,
00384 0x5e97, 0x6dfb, 0x7e8f, 0x751c, 0x8cbc, 0x8ee2, 0x985b, 0x70b9,
00385 0x4f1d, 0x6bbf, 0x6fb1, 0x7530, 0x96fb, 0x514e, 0x5410, 0x5835,
00386 0x5857, 0x59ac, 0x5c60, 0x5f92, 0x6597, 0x675c, 0x6e21, 0x767b,
00387 0x83df, 0x8ced, 0x9014, 0x90fd, 0x934d, 0x7825, 0x783a, 0x52aa,
00388 0x5ea6, 0x571f, 0x5974, 0x6012, 0x5012, 0x515a, 0x51ac, 0x51cd,
00389 0x5200, 0x5510, 0x5854, 0x5858, 0x5957, 0x5b95, 0x5cf6, 0x5d8b,
00390 0x60bc, 0x6295, 0x642d, 0x6771, 0x6843, 0x68bc, 0x68df, 0x76d7,
00391 0x6dd8, 0x6e6f, 0x6d9b, 0x706f, 0x71c8, 0x5f53, 0x75d8, 0x7977,
00392 0x7b49, 0x7b54, 0x7b52, 0x7cd6, 0x7d71, 0x5230,
00393 /* 0x46 */
00394 0x8463, 0x8569, 0x85e4, 0x8a0e, 0x8b04, 0x8c46, 0x8e0f, 0x9003,
00395 0x900f, 0x9419, 0x9676, 0x982d, 0x9a30, 0x95d8, 0x50cd, 0x52d5,
00396 0x540c, 0x5802, 0x5c0e, 0x61a7, 0x649e, 0x6d1e, 0x77b3, 0x7ae5,
00397 0x80f4, 0x8404, 0x9053, 0x9285, 0x5ce0, 0x9d07, 0x533f, 0x5f97,
00398 0x5fb3, 0x6d9c, 0x7279, 0x7763, 0x79bf, 0x7be4, 0x6bd2, 0x72ec,
00399 0x8aad, 0x6803, 0x6a61, 0x51f8, 0x7a81, 0x6934, 0x5c4a, 0x9cf6,
00400 0x82eb, 0x5bc5, 0x9149, 0x701e, 0x5678, 0x5c6f, 0x60c7, 0x6566,
00401 0x6c8c, 0x8c5a, 0x9041, 0x9813, 0x5451, 0x66c7, 0x920d, 0x5948,
00402 0x90a3, 0x5185, 0x4e4d, 0x51ea, 0x8599, 0x8b0e, 0x7058, 0x637a,
00403 0x934b, 0x6962, 0x99b4, 0x7e04, 0x7577, 0x5357, 0x6960, 0x8edf,
00404 0x96e3, 0x6c5d, 0x4e8c, 0x5c3c, 0x5f10, 0x8fe9, 0x5302, 0x8cd1,
00405 0x8089, 0x8679, 0x5eff, 0x65e5, 0x4e73, 0x5165,
00406 /* 0x47 */
00407 0x5982, 0x5c3f, 0x97ee, 0x4efb, 0x598a, 0x5fcd, 0x8a8d, 0x6fe1,
```

```

00408 0x79b0, 0x7962, 0x5be7, 0x8471, 0x732b, 0x71b1, 0x5e74, 0x5ff5,
00409 0x637b, 0x649a, 0x71c3, 0x7c98, 0x4e43, 0x5efc, 0x4e4b, 0x57dc,
00410 0x56a2, 0x60a9, 0x6fc3, 0x7d0d, 0x80fd, 0x8133, 0x81bf, 0x8fb2,
00411 0x8997, 0x86a4, 0x5df4, 0x628a, 0x64ad, 0x8987, 0x6777, 0x6ce2,
00412 0x6d3e, 0x7436, 0x7834, 0x5a46, 0x7f75, 0x82ad, 0x99ac, 0x4ff3,
00413 0x5ec3, 0x62dd, 0x6392, 0x6557, 0x676f, 0x76c3, 0x724c, 0x80cc,
00414 0x80ba, 0x8f29, 0x914d, 0x500d, 0x57f9, 0x5a92, 0x6885, 0x6973,
00415 0x7164, 0x72fd, 0x8cb7, 0x58f2, 0x8ce0, 0x966a, 0x9019, 0x877f,
00416 0x79e4, 0x77e7, 0x8429, 0x4f2f, 0x5265, 0x535a, 0x62cd, 0x67cf,
00417 0x6cca, 0x767d, 0x7b94, 0x7c95, 0x8236, 0x8584, 0x8feb, 0x66dd,
00418 0x6f20, 0x7206, 0x7e1b, 0x83ab, 0x99c1, 0x9ea6,
00419 /* 0x48 */
00420 0x51fd, 0x7bb1, 0x7872, 0x7bb8, 0x8087, 0x7b48, 0x6ae8, 0x5e61,
00421 0x808c, 0x7551, 0x7560, 0x516b, 0x9262, 0x6e8c, 0x767a, 0x9197,
00422 0x9aea, 0x4f10, 0x7f70, 0x629c, 0x7b4f, 0x95a5, 0x9ce9, 0x567a,
00423 0x5859, 0x86e4, 0x96bc, 0x4f34, 0x5224, 0x534a, 0x53cd, 0x53db,
00424 0x5e06, 0x642c, 0x6591, 0x677f, 0x6c3e, 0x6c4e, 0x7248, 0x72af,
00425 0x73ed, 0x7554, 0x7e41, 0x822c, 0x85e9, 0x8ca9, 0x7bc4, 0x91c6,
00426 0x7169, 0x9812, 0x98ef, 0x633d, 0x6669, 0x756a, 0x76e4, 0x78d0,
00427 0x8543, 0x86ee, 0x532a, 0x5351, 0x5426, 0x5983, 0x5e87, 0x5f7c,
00428 0x60b2, 0x6249, 0x6279, 0x62ab, 0x6590, 0x6bd4, 0x6ccc, 0x75b2,
00429 0x76ae, 0x7891, 0x79d8, 0x7dcf, 0x7f77, 0x80a5, 0x88ab, 0x8ab9,
00430 0x8cbb, 0x907f, 0x975e, 0x98db, 0x6a0b, 0x7c38, 0x5099, 0x5c3e,
00431 0x5fae, 0x6787, 0x6bd8, 0x7435, 0x7709, 0x7f8e,
00432 /* 0x49 */
00433 0x9f3b, 0x67ca, 0x7a17, 0x5339, 0x758b, 0x9aed, 0x5f66, 0x819d,
00434 0x83f1, 0x8098, 0x5f3c, 0x5fc5, 0x7562, 0x7b46, 0x903c, 0x6867,
00435 0x59eb, 0x5a9b, 0x7d10, 0x767e, 0x8b2c, 0x4ff5, 0x5f6a, 0x6a19,
00436 0x6c37, 0x6f02, 0x74e2, 0x7968, 0x8868, 0x8a55, 0x8c79, 0x5edf,
00437 0x63cf, 0x75c5, 0x79d2, 0x82d7, 0x9328, 0x92f2, 0x849c, 0x86ed,
00438 0x9c2d, 0x54c1, 0x5f6c, 0x658c, 0x6d5c, 0x7015, 0x8ca7, 0x8cd3,
00439 0x983b, 0x654f, 0x74f6, 0x4e0d, 0x4ed8, 0x57e0, 0x592b, 0x5a66,
00440 0x5bcc, 0x51a8, 0x5e03, 0x5e9c, 0x6016, 0x6276, 0x6577, 0x65a7,
00441 0x666e, 0x6d6e, 0x7236, 0x7b26, 0x8150, 0x819a, 0x8299, 0x8b5c,
00442 0x8ca0, 0x8ce6, 0x8d74, 0x961c, 0x9644, 0x4fae, 0x64ab, 0x6b66,
00443 0x821e, 0x8461, 0x856a, 0x90e8, 0x5c01, 0x6953, 0x98a8, 0x847a,
00444 0x8557, 0x4f0f, 0x526f, 0x5fa9, 0x5e45, 0x670d,
00445 /* 0x4a */
00446 0x798f, 0x8179, 0x8907, 0x8986, 0x6df5, 0x5f17, 0x6255, 0x6cb8,
00447 0x4ecf, 0x7269, 0x9b92, 0x5206, 0x543b, 0x5674, 0x58b3, 0x61a4,
00448 0x626e, 0x711a, 0x596e, 0x7c89, 0x7cde, 0x7dlb, 0x96f0, 0x6587,
00449 0x805e, 0x4e19, 0x4f75, 0x5175, 0x5840, 0x5e63, 0x5e73, 0x5f0a,
00450 0x67c4, 0x4e26, 0x853d, 0x9589, 0x965b, 0x7c73, 0x9801, 0x50fb,
00451 0x58c1, 0x7656, 0x78a7, 0x5225, 0x77a5, 0x8511, 0x7b86, 0x504f,
00452 0x5909, 0x7247, 0x7bc7, 0x7de8, 0x8fba, 0x8fd4, 0x904d, 0x4fbf,
00453 0x52c9, 0x5a29, 0x5f01, 0x97ad, 0x4fdd, 0x8217, 0x92ea, 0x5703,
00454 0x6355, 0x6b69, 0x752b, 0x88dc, 0x8f14, 0x7a42, 0x52df, 0x5893,
00455 0x6155, 0x620a, 0x66ae, 0x6bcd, 0x7c3f, 0x83e9, 0x5023, 0x4ff8,
00456 0x5305, 0x5446, 0x5831, 0x5949, 0x5b9d, 0x5cf0, 0x5cef, 0x5d29,
00457 0x5e96, 0x62b1, 0x6367, 0x653e, 0x65b9, 0x670b,
00458 /* 0x4b */
00459 0x6cd5, 0x6ce1, 0x70f9, 0x7832, 0x7e2b, 0x80de, 0x82b3, 0x840c,
00460 0x84ec, 0x8702, 0x8912, 0x8a2a, 0x8c4a, 0x90a6, 0x92d2, 0x98fd,
00461 0x9cf3, 0x9d6c, 0x4e4f, 0x4ea1, 0x508d, 0x5256, 0x574a, 0x59a8,
00462 0x5e3d, 0x5fd8, 0x5fd9, 0x623f, 0x66b4, 0x671b, 0x67d0, 0x68d2,
00463 0x5192, 0x7d21, 0x80aa, 0x81a8, 0x8b00, 0x8c8c, 0x8cbf, 0x927e,
00464 0x9632, 0x5420, 0x982c, 0x5317, 0x50d5, 0x535c, 0x58a8, 0x64b2,
00465 0x6734, 0x7267, 0x7766, 0x7a46, 0x91e6, 0x52c3, 0x6ca1, 0x6b86,
00466 0x5800, 0x5e4c, 0x5954, 0x672c, 0x7ffb, 0x51e1, 0x76c6, 0x6469,
00467 0x78e8, 0x9b54, 0x9ebb, 0x57cb, 0x59b9, 0x6627, 0x679a, 0x6bce,
00468 0x54e9, 0x69d9, 0x5e55, 0x819c, 0x6795, 0x9baa, 0x67fe, 0x9c52,
00469 0x685d, 0x4ea6, 0x4fe3, 0x53c8, 0x62b9, 0x672b, 0x6cab, 0x8fc4,
00470 0x4fad, 0x7e6d, 0x9ebf, 0x4e07, 0x6162, 0x6e80,
00471 /* 0x4c */
00472 0x6f2b, 0x8513, 0x5473, 0x672a, 0x9b45, 0x5df3, 0x7b95, 0x5cac,
00473 0x5bc6, 0x871c, 0x6e4a, 0x84d1, 0x7a14, 0x8108, 0x5999, 0x7c8d,
00474 0x6c11, 0x7720, 0x52d9, 0x5922, 0x7121, 0x725f, 0x77db, 0x9727,
00475 0x9d61, 0x690b, 0x5a7f, 0x5a18, 0x51a5, 0x540d, 0x547d, 0x660e,
00476 0x76df, 0x8ff7, 0x9298, 0x9cf4, 0x59ea, 0x725d, 0x6ec5, 0x514d,
00477 0x68c9, 0x7dbf, 0x7dec, 0x9762, 0x9eba, 0x6478, 0x6a21, 0x8302,
00478 0x5984, 0x5b5f, 0x6bdb, 0x731b, 0x76f2, 0x7db2, 0x8017, 0x8499,
00479 0x5132, 0x6728, 0x9ed9, 0x76ee, 0x6762, 0x52ff, 0x9905, 0x5c24,
00480 0x623b, 0x7c7e, 0x8cb0, 0x554f, 0x60b6, 0x7d0b, 0x9580, 0x5301,
00481 0x4e5f, 0x51b6, 0x591c, 0x723a, 0x8036, 0x91ce, 0x5f25, 0x77e2,
00482 0x5384, 0x5f79, 0x7d04, 0x85ac, 0x8a33, 0x8e8d, 0x9756, 0x67f3,
00483 0x85ae, 0x9453, 0x6109, 0x6108, 0x6cb9, 0x7652,
00484 /* 0x4d */
00485 0x8aed, 0x8f38, 0x552f, 0x4f51, 0x512a, 0x52c7, 0x53cb, 0x5ba5,
00486 0x5e7d, 0x60a0, 0x6182, 0x63d6, 0x6709, 0x67da, 0x6e67, 0x6d8c,
00487 0x7336, 0x7337, 0x7531, 0x7950, 0x88d5, 0x8a98, 0x904a, 0x9091,
00488 0x90f5, 0x96c4, 0x878d, 0x5915, 0x4e88, 0x4f59, 0x4e0e, 0x8a89,
00489 0x8f3f, 0x9810, 0x50ad, 0x5e7c, 0x5996, 0x5bb9, 0x5eb8, 0x63da,
00490 0x63fa, 0x64c1, 0x66dc, 0x694a, 0x69d8, 0x6d0b, 0x6eb6, 0x7194,
00491 0x7528, 0x7aaf, 0x7f8a, 0x8000, 0x8449, 0x84c9, 0x8981, 0x8b21,
00492 0x8e0a, 0x9065, 0x967d, 0x990a, 0x617e, 0x6291, 0x6b32, 0x6c83,
00493 0x6d74, 0x7fcc, 0x7ffc, 0x6dc0, 0x7f85, 0x87ba, 0x88f8, 0x6765,
00494 0x83b1, 0x983c, 0x96f7, 0x6d1b, 0x7d61, 0x843d, 0x916a, 0x4e71,

```

```
00495 0x5375, 0x5d50, 0x6b04, 0x6feb, 0x85cd, 0x862d, 0x89a7, 0x5229,
00496 0x540f, 0x5c65, 0x674e, 0x68a8, 0x7406, 0x7483,
00497 /* 0x4e */
00498 0x75e2, 0x88cf, 0x88e1, 0x91cc, 0x96e2, 0x9678, 0x5f8b, 0x7387,
00499 0x7acb, 0x844e, 0x63a0, 0x7565, 0x5289, 0x6d41, 0x6e9c, 0x7409,
00500 0x7559, 0x786b, 0x7c92, 0x9686, 0x7adc, 0x9f8d, 0x4fb6, 0x616e,
00501 0x65c5, 0x865c, 0x4e86, 0x4eae, 0x50da, 0x4e21, 0x51cc, 0x5bee,
00502 0x6599, 0x6881, 0x6dbc, 0x731f, 0x7642, 0x77ad, 0x7a1c, 0x7ce7,
00503 0x826f, 0x8ad2, 0x907c, 0x91cf, 0x9675, 0x9818, 0x529b, 0x7dd1,
00504 0x502b, 0x5398, 0x6797, 0x6dcf, 0x71d0, 0x7433, 0x81e8, 0x8f2a,
00505 0x96a3, 0x9c57, 0x9e9f, 0x7460, 0x5841, 0x6d99, 0x7d2f, 0x985e,
00506 0x4ee4, 0x4f36, 0x4f8b, 0x51b7, 0x52b1, 0x5dba, 0x601c, 0x73b2,
00507 0x793c, 0x82d3, 0x9234, 0x96b7, 0x96f6, 0x970a, 0x9e97, 0x9f62,
00508 0x66a6, 0x6b74, 0x5217, 0x52a3, 0x70c8, 0x88c2, 0x5ec9, 0x604b,
00509 0x6190, 0x6f23, 0x7149, 0x7c3e, 0x7df4, 0x806f,
00510 /* 0x4f */
00511 0x84ee, 0x9023, 0x932c, 0x5442, 0x9b6f, 0x6ad3, 0x7089, 0x8cc2,
00512 0x8def, 0x9732, 0x52b4, 0x5a41, 0x5eca, 0x5f04, 0x6717, 0x697c,
00513 0x6994, 0x6d6a, 0x6f0f, 0x7262, 0x72fc, 0x7bed, 0x8001, 0x807e,
00514 0x874b, 0x90ce, 0x516d, 0x9e93, 0x7984, 0x808b, 0x9332, 0x8ad6,
00515 0x502d, 0x548c, 0x8a71, 0x6b6a, 0x8cc4, 0x8107, 0x60d1, 0x67a0,
00516 0x9df2, 0x4e99, 0x4e98, 0x9c10, 0x8a6b, 0x85c1, 0x8568, 0x6900,
00517 0x6e7e, 0x7897, 0x8155, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00518 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00519 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00520 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00521 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00522 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00523 /* 0x50 */
00524 0x5f0c, 0x4e10, 0x4e15, 0x4e2a, 0x4e31, 0x4e36, 0x4e3c, 0x4e3f,
00525 0x4e42, 0x4e56, 0x4e58, 0x4e82, 0x4e85, 0x8c6b, 0x4e8a, 0x8212,
00526 0x5f0d, 0x4e8e, 0x4e9e, 0x4e9f, 0x4ea0, 0x4ea2, 0x4eb0, 0x4eb3,
00527 0x4eb6, 0x4ece, 0x4ecd, 0x4ec4, 0x4ec6, 0x4ec2, 0x4ed7, 0x4ede,
00528 0x4eed, 0x4edf, 0x4ef7, 0x4f09, 0x4f5a, 0x4f30, 0x4f5b, 0x4f5d,
00529 0x4f57, 0x4f47, 0x4f76, 0x4f88, 0x4f8f, 0x4f98, 0x4f7b, 0x4f69,
00530 0x4f70, 0x4f91, 0x4f91, 0x4f6f, 0x4f86, 0x4f96, 0x5118, 0x4fd4, 0x4fdf,
00531 0x4fce, 0x4fd8, 0x4fdb, 0x4fd1, 0x4fda, 0x4fd0, 0x4fe4, 0x4fe5,
00532 0x501a, 0x5028, 0x5014, 0x502a, 0x5025, 0x5005, 0x4f1c, 0x4ff6,
00533 0x5021, 0x5029, 0x502c, 0x4ffe, 0x4fef, 0x5011, 0x5006, 0x5043,
00534 0x5047, 0x6703, 0x5055, 0x5050, 0x5048, 0x505a, 0x5056, 0x506c,
00535 0x5078, 0x5080, 0x509a, 0x5085, 0x50b4, 0x50b2,
00536 /* 0x51 */
00537 0x50c9, 0x50ca, 0x50b3, 0x50c2, 0x50d6, 0x50de, 0x50e5, 0x50ed,
00538 0x50e3, 0x50ee, 0x50f9, 0x50f5, 0x5109, 0x5101, 0x5102, 0x5116,
00539 0x5115, 0x5114, 0x511a, 0x5121, 0x513a, 0x5137, 0x513c, 0x513b,
00540 0x513f, 0x5140, 0x5152, 0x514c, 0x5154, 0x5162, 0x7af8, 0x5169,
00541 0x516a, 0x516e, 0x5180, 0x5182, 0x56d8, 0x518c, 0x5189, 0x518f,
00542 0x5191, 0x5193, 0x5195, 0x5196, 0x51a4, 0x51a6, 0x51a2, 0x51a9,
00543 0x51aa, 0x51ab, 0x51b3, 0x51b1, 0x51b2, 0x51b0, 0x51b5, 0x51bd,
00544 0x51c5, 0x51c9, 0x51db, 0x51e0, 0x8655, 0x51e9, 0x51ed, 0x51f0,
00545 0x51f5, 0x51fe, 0x5204, 0x520b, 0x5214, 0x520e, 0x5227, 0x522a,
00546 0x522e, 0x5233, 0x5239, 0x524f, 0x5244, 0x524b, 0x524c, 0x525e,
00547 0x5254, 0x526a, 0x5274, 0x5269, 0x5273, 0x527f, 0x527d, 0x528d,
00548 0x5294, 0x5292, 0x5291, 0x5288, 0x5291, 0x8fa8,
00549 /* 0x52 */
00550 0x8fa7, 0x52ac, 0x52ad, 0x52bc, 0x52b5, 0x52c1, 0x52cd, 0x52d7,
00551 0x52de, 0x52e3, 0x52e6, 0x98ed, 0x52e0, 0x52f3, 0x52f5, 0x52f8,
00552 0x52f9, 0x5306, 0x5308, 0x7538, 0x530d, 0x5310, 0x530f, 0x5315,
00553 0x531a, 0x5323, 0x532f, 0x5331, 0x5333, 0x5338, 0x5340, 0x5346,
00554 0x5345, 0x4e17, 0x5349, 0x534d, 0x51d6, 0x535e, 0x5369, 0x536e,
00555 0x5918, 0x537b, 0x5377, 0x5382, 0x5396, 0x53a0, 0x53a6, 0x53a5,
00556 0x53ae, 0x53b0, 0x53b6, 0x53c3, 0x7c12, 0x96d9, 0x53df, 0x66fc,
00557 0x71ee, 0x53ee, 0x53ed, 0x53fa, 0x5401, 0x543d, 0x5440, 0x5440,
00558 0x542c, 0x542d, 0x543c, 0x542e, 0x5436, 0x5429, 0x541d, 0x544e,
00559 0x548f, 0x5475, 0x548e, 0x545f, 0x5471, 0x5477, 0x5470, 0x5492,
00560 0x547b, 0x5480, 0x5476, 0x5484, 0x5490, 0x5486, 0x54c7, 0x54a2,
00561 0x54b8, 0x54a5, 0x54ac, 0x54c4, 0x54c8, 0x54a8,
00562 /* 0x53 */
00563 0x54ab, 0x54c2, 0x54a4, 0x54be, 0x54bc, 0x54d8, 0x54e5, 0x54e6,
00564 0x550f, 0x5514, 0x54fd, 0x54ee, 0x54ed, 0x54fa, 0x54e2, 0x5539,
00565 0x5540, 0x5563, 0x554c, 0x552e, 0x555c, 0x5545, 0x5556, 0x5557,
00566 0x5538, 0x5533, 0x555d, 0x5599, 0x5580, 0x54af, 0x558a, 0x559f,
00567 0x557b, 0x557e, 0x5598, 0x559e, 0x55ae, 0x557c, 0x5583, 0x55a9,
00568 0x5587, 0x55a8, 0x55da, 0x55c5, 0x55df, 0x55c4, 0x55dc, 0x55e4,
00569 0x55d4, 0x5614, 0x55f7, 0x5616, 0x55fe, 0x55fd, 0x561b, 0x55f9,
00570 0x564e, 0x5650, 0x71df, 0x5634, 0x5636, 0x5632, 0x5638, 0x566b,
00571 0x5664, 0x562f, 0x566c, 0x566a, 0x5686, 0x5680, 0x568a, 0x56a0,
00572 0x5694, 0x568f, 0x56a5, 0x56ae, 0x56b6, 0x56b4, 0x56c2, 0x56bc,
00573 0x56c1, 0x56c3, 0x56c0, 0x56c8, 0x56ce, 0x56d1, 0x56d3, 0x56d7,
00574 0x56ee, 0x56f9, 0x5700, 0x56ff, 0x5704, 0x5709,
00575 /* 0x54 */
00576 0x5708, 0x570b, 0x570d, 0x5713, 0x5718, 0x5716, 0x55c7, 0x571c,
00577 0x5726, 0x5737, 0x5738, 0x574e, 0x573b, 0x5740, 0x574f, 0x5769,
00578 0x57c0, 0x5788, 0x5761, 0x577f, 0x5789, 0x5793, 0x57a0, 0x57b3,
00579 0x57a4, 0x57aa, 0x57b0, 0x57c3, 0x57c6, 0x57d4, 0x57d2, 0x57d3,
00580 0x580a, 0x57d6, 0x57e3, 0x580b, 0x5819, 0x581d, 0x5872, 0x5821,
00581 0x5862, 0x584b, 0x5870, 0x6bc0, 0x5852, 0x583d, 0x5879, 0x5885,
```



```

00582 0x58b9, 0x589f, 0x58ab, 0x58ba, 0x58de, 0x58bb, 0x58b8, 0x58ae,
00583 0x58c5, 0x58d3, 0x58d1, 0x58d7, 0x58d9, 0x58d8, 0x58e5, 0x58dc,
00584 0x58e4, 0x58df, 0x58ef, 0x58fa, 0x58f9, 0x58fb, 0x58fc, 0x58fd,
00585 0x5902, 0x590a, 0x5910, 0x591b, 0x68a6, 0x5925, 0x592c, 0x592d,
00586 0x5932, 0x5938, 0x593e, 0x7ad2, 0x5955, 0x5950, 0x594e, 0x595a,
00587 0x5958, 0x5962, 0x5960, 0x5967, 0x596c, 0x5969,
00588 /* 0x55 */
00589 0x5978, 0x5981, 0x599d, 0x4f5e, 0x4fab, 0x59a3, 0x59b2, 0x59c6,
00590 0x59e8, 0x59dc, 0x598d, 0x59d9, 0x59da, 0x5a25, 0x5a1f, 0x5a11,
00591 0x5a1c, 0x5a09, 0x5a1a, 0x5a40, 0x5a6c, 0x5a49, 0x5a35, 0x5a36,
00592 0x5a62, 0x5a6a, 0x5a9a, 0x5abc, 0x5abe, 0x5acb, 0x5ac2, 0x5abd,
00593 0x5ae3, 0x5ad7, 0x5ae6, 0x5ae9, 0x5ad6, 0x5afa, 0x5afb, 0x5b0c,
00594 0x5b0b, 0x5b16, 0x5b32, 0x5ad0, 0x5b2a, 0x5b36, 0x5b3e, 0x5b43,
00595 0x5b45, 0x5b40, 0x5b51, 0x5b55, 0x5b5a, 0x5b5b, 0x5b65, 0x5b69,
00596 0x5b70, 0x5b73, 0x5b75, 0x5b78, 0x6588, 0x5b7a, 0x5b80, 0x5b83,
00597 0x5ba6, 0x5bb8, 0x5bc3, 0x5bc7, 0x5bc9, 0x5bd4, 0x5bd0, 0x5be4,
00598 0x5be6, 0x5be2, 0x5bde, 0x5be5, 0x5beb, 0x5bf0, 0x5bf6, 0x5bf3,
00599 0x5c05, 0x5c07, 0x5c08, 0x5c0d, 0x5c13, 0x5c20, 0x5c22, 0x5c28,
00600 0x5c38, 0x5c39, 0x5c41, 0x5c46, 0x5c4e, 0x5c53,
00601 /* 0x56 */
00602 0x5c50, 0x5c4f, 0x5b71, 0x5c6c, 0x5c6e, 0x4e62, 0x5c76, 0x5c79,
00603 0x5c8c, 0x5c91, 0x5c94, 0x599b, 0x5cab, 0x5cbb, 0x5cb6, 0x5cbc,
00604 0x5cb7, 0x5cc5, 0x5cbe, 0x5cc7, 0x5cd9, 0x5ce9, 0x5cfd, 0x5cfa,
00605 0x5ced, 0x5d8c, 0x5cea, 0x5d0b, 0x5d15, 0x5d17, 0x5d5c, 0x5d1f,
00606 0x5d1b, 0x5d11, 0x5d14, 0x5d22, 0x5d1a, 0x5d19, 0x5d18, 0x5d4c,
00607 0x5d52, 0x5d4e, 0x5d4b, 0x5d6c, 0x5d73, 0x5d76, 0x5d87, 0x5d84,
00608 0x5d82, 0x5da2, 0x5d9d, 0x5dac, 0x5dae, 0x5dbd, 0x5d90, 0x5db7,
00609 0x5dbc, 0x5dc9, 0x5dcd, 0x5dd3, 0x5dd2, 0x5dd6, 0x5ddb, 0x5deb,
00610 0x5df2, 0x5df5, 0x5e0b, 0x5e1a, 0x5e19, 0x5e11, 0x5e1b, 0x5e36,
00611 0x5e37, 0x5e44, 0x5e43, 0x5e40, 0x5e4e, 0x5e57, 0x5e54, 0x5e5f,
00612 0x5e62, 0x5e64, 0x5e47, 0x5e75, 0x5e76, 0x5e7a, 0x9ebc, 0x5e7f,
00613 0x5ea0, 0x5ec1, 0x5ec2, 0x5ec8, 0x5ed0, 0x5ecf,
00614 /* 0x57 */
00615 0x5ed6, 0x5ee3, 0x5edd, 0x5eda, 0x5edb, 0x5ee2, 0x5ee1, 0x5ee8,
00616 0x5ee9, 0x5eec, 0x5ef1, 0x5ef3, 0x5ef0, 0x5ef4, 0x5ef8, 0x5efe,
00617 0x5f03, 0x5f09, 0x5f5d, 0x5f5c, 0x5f0b, 0x5f11, 0x5f16, 0x5f29,
00618 0x5f2d, 0x5f38, 0x5f41, 0x5f48, 0x5f4c, 0x5f4e, 0x5f2f, 0x5f51,
00619 0x5f56, 0x5f57, 0x5f59, 0x5f61, 0x5f6d, 0x5f73, 0x5f77, 0x5f83,
00620 0x5f82, 0x5f7f, 0x5f8a, 0x5f88, 0x5f91, 0x5f87, 0x5f9e, 0x5f99,
00621 0x5f98, 0x5fa0, 0x5fa8, 0x5fad, 0x5fbc, 0x5fd6, 0x5ffb, 0x5fe4,
00622 0x5ff8, 0x5ff1, 0x5fdd, 0x60b3, 0x5fff, 0x6021, 0x6060, 0x6019,
00623 0x6010, 0x6029, 0x600e, 0x6031, 0x601b, 0x6015, 0x602b, 0x6026,
00624 0x600f, 0x603a, 0x605a, 0x6041, 0x606a, 0x6077, 0x605f, 0x604a,
00625 0x6046, 0x604d, 0x6063, 0x6043, 0x6064, 0x6042, 0x606c, 0x606b,
00626 0x6059, 0x6081, 0x608d, 0x60e7, 0x6083, 0x609a,
00627 /* 0x58 */
00628 0x6084, 0x609b, 0x6096, 0x6097, 0x6092, 0x60a7, 0x608b, 0x60e1,
00629 0x60b8, 0x60e0, 0x60d3, 0x60b4, 0x5ff0, 0x60bd, 0x60c6, 0x60b5,
00630 0x60d8, 0x614d, 0x6115, 0x6106, 0x60f6, 0x60f7, 0x6100, 0x60f4,
00631 0x60fa, 0x6103, 0x6121, 0x60fb, 0x60f1, 0x610d, 0x610e, 0x6147,
00632 0x613e, 0x6128, 0x6127, 0x614a, 0x613f, 0x613c, 0x612c, 0x6134,
00633 0x613d, 0x6142, 0x6144, 0x6173, 0x6177, 0x6158, 0x6159, 0x615a,
00634 0x616b, 0x6174, 0x616f, 0x6165, 0x6171, 0x615f, 0x615d, 0x6153,
00635 0x6175, 0x6199, 0x6196, 0x6187, 0x61ac, 0x6194, 0x619a, 0x618a,
00636 0x6191, 0x61ab, 0x61ae, 0x61cc, 0x61ca, 0x61c9, 0x61f7, 0x61c8,
00637 0x61c3, 0x61c6, 0x61ba, 0x61cb, 0x7f79, 0x61cd, 0x61e6, 0x61e3,
00638 0x61f6, 0x61fa, 0x61ff, 0x61fd, 0x61fc, 0x61fe, 0x6200,
00639 0x6208, 0x6209, 0x620d, 0x620c, 0x6214, 0x621b,
00640 /* 0x59 */
00641 0x621e, 0x6221, 0x622a, 0x622e, 0x6230, 0x6232, 0x6233, 0x6241,
00642 0x624e, 0x625e, 0x6263, 0x625b, 0x6260, 0x6268, 0x627c, 0x6282,
00643 0x6289, 0x627e, 0x6292, 0x6293, 0x6296, 0x62d4, 0x6283, 0x6294,
00644 0x62d7, 0x62d1, 0x62bb, 0x62cf, 0x62ff, 0x62c6, 0x64d4, 0x62c8,
00645 0x62dc, 0x62cc, 0x62ca, 0x62c2, 0x62c7, 0x629b, 0x62c9, 0x630c,
00646 0x62ee, 0x62f1, 0x6327, 0x6302, 0x6308, 0x62ef, 0x62f5, 0x6350,
00647 0x633e, 0x634d, 0x641c, 0x634f, 0x6396, 0x638e, 0x6380, 0x63ab,
00648 0x6376, 0x63a3, 0x638f, 0x6389, 0x639f, 0x63b5, 0x636b, 0x6369,
00649 0x63be, 0x63e9, 0x63c0, 0x63c6, 0x63e3, 0x63c9, 0x63d2, 0x63f6,
00650 0x63c4, 0x6416, 0x6434, 0x6406, 0x6413, 0x6426, 0x6436, 0x651d,
00651 0x6417, 0x6428, 0x640f, 0x6467, 0x646f, 0x6476, 0x644e, 0x652a,
00652 0x6495, 0x6493, 0x64a5, 0x64a9, 0x6488, 0x64bc,
00653 /* 0x5a */
00654 0x64da, 0x64d2, 0x64c5, 0x64c7, 0x64bb, 0x64d8, 0x64c2, 0x64f1,
00655 0x64e7, 0x8209, 0x64e0, 0x64e1, 0x62ac, 0x64e3, 0x64ef, 0x652c,
00656 0x64f6, 0x64f4, 0x64f2, 0x64fa, 0x6500, 0x64fd, 0x6518, 0x651c,
00657 0x6505, 0x6524, 0x6523, 0x652b, 0x6534, 0x6535, 0x6537, 0x6536,
00658 0x6538, 0x754b, 0x6548, 0x6556, 0x6555, 0x654d, 0x6558, 0x655e,
00659 0x655d, 0x6572, 0x6578, 0x6582, 0x6583, 0x8b8a, 0x659b, 0x659f,
00660 0x65ab, 0x65b7, 0x65c3, 0x65c6, 0x65c1, 0x65c4, 0x65cc, 0x65d2,
00661 0x65db, 0x65d9, 0x65e0, 0x65e1, 0x65f1, 0x6772, 0x660a, 0x6603,
00662 0x65fb, 0x6773, 0x6635, 0x6636, 0x6634, 0x661c, 0x664f, 0x6644,
00663 0x6649, 0x6641, 0x665e, 0x665d, 0x6664, 0x6667, 0x6668, 0x665f,
00664 0x6662, 0x6670, 0x6683, 0x6688, 0x668e, 0x6689, 0x6684, 0x6698,
00665 0x669d, 0x66c1, 0x66b9, 0x66c9, 0x66be, 0x66bc,
00666 /* 0x5b */
00667 0x66c4, 0x66b8, 0x66d6, 0x66da, 0x66e0, 0x663f, 0x66e6, 0x66e9,
00668 0x66f0, 0x66f5, 0x66f7, 0x670f, 0x6716, 0x671e, 0x6726, 0x6727,

```



```
00669 0x9738, 0x672e, 0x673f, 0x6736, 0x6741, 0x6738, 0x6737, 0x6746,
00670 0x675e, 0x6760, 0x6759, 0x6763, 0x6764, 0x6789, 0x6770, 0x67a9,
00671 0x677c, 0x676a, 0x678c, 0x678b, 0x67a6, 0x67a1, 0x6785, 0x67b7,
00672 0x67ef, 0x67b4, 0x67ec, 0x67b3, 0x67e9, 0x67b8, 0x67e4, 0x67de,
00673 0x67dd, 0x67e2, 0x67ee, 0x67b9, 0x67ce, 0x67c6, 0x67e7, 0x6a9c,
00674 0x681e, 0x6846, 0x6829, 0x6840, 0x684d, 0x6832, 0x684e, 0x68b3,
00675 0x682b, 0x6859, 0x6863, 0x6877, 0x687f, 0x689f, 0x688f, 0x68ad,
00676 0x6894, 0x689d, 0x689b, 0x6883, 0x6aae, 0x68b9, 0x6874, 0x68b5,
00677 0x68a0, 0x68ba, 0x690f, 0x688d, 0x687e, 0x6901, 0x68ca, 0x6908,
00678 0x68d8, 0x6922, 0x6926, 0x68e1, 0x690c, 0x68cd,
00679 /* 0x5c */
00680 0x68d4, 0x68e7, 0x68d5, 0x6936, 0x6912, 0x6904, 0x68d7, 0x68e3,
00681 0x6925, 0x68f9, 0x68e0, 0x68ef, 0x6928, 0x692a, 0x691a, 0x6923,
00682 0x6921, 0x68c6, 0x6979, 0x6977, 0x695c, 0x6978, 0x696b, 0x6954,
00683 0x697e, 0x696e, 0x6939, 0x6974, 0x693d, 0x6959, 0x6930, 0x6961,
00684 0x695e, 0x695d, 0x6981, 0x696a, 0x69b2, 0x69ae, 0x69d0, 0x69bf,
00685 0x69c1, 0x69d3, 0x69be, 0x69ce, 0x5be8, 0x69ca, 0x69dd, 0x69bb,
00686 0x69c3, 0x69a7, 0x6a2e, 0x6991, 0x69a0, 0x699c, 0x6995, 0x69b4,
00687 0x69de, 0x69e8, 0x6a02, 0x6a1b, 0x69ff, 0x6b0a, 0x69f9, 0x69f2,
00688 0x69e7, 0x6a05, 0x69b1, 0x6a1e, 0x69ed, 0x6a14, 0x69eb, 0x6a0a,
00689 0x6a12, 0x6ac1, 0x6a23, 0x6a13, 0x6a44, 0x6a0c, 0x6a72, 0x6a36,
00690 0x6a78, 0x6a47, 0x6a62, 0x6a59, 0x6a66, 0x6a48, 0x6a38, 0x6a22,
00691 0x6a90, 0x6a8d, 0x6a80, 0x6a84, 0x6aa2, 0x6aa3,
00692 /* 0x5d */
00693 0x6a97, 0x8617, 0x6abb, 0x6ac3, 0x6ac2, 0x6ab8, 0x6ab3, 0x6aac,
00694 0x6ade, 0x6ad1, 0x6adf, 0x6aaa, 0x6ada, 0x6aea, 0x6afb, 0x6b05,
00695 0x8616, 0x6afa, 0x6b12, 0x6b16, 0x9b31, 0x6b1f, 0x6b38, 0x6b37,
00696 0x76dc, 0x6b39, 0x98ee, 0x6b47, 0x6b43, 0x6b49, 0x6b50, 0x6b59,
00697 0x6b54, 0x6b5b, 0x6b5f, 0x6b61, 0x6b78, 0x6b79, 0x6b7e, 0x6b80,
00698 0x6b84, 0x6b83, 0x6b8d, 0x6b98, 0x6b95, 0x6b9e, 0x6ba4, 0x6baa,
00699 0x6bab, 0x6baf, 0x6bb2, 0x6bb1, 0x6bb3, 0x6bb7, 0x6bbc, 0x6bc6,
00700 0x6bcb, 0x6bd3, 0x6bdf, 0x6bec, 0x6beb, 0x6bf3, 0x6bef, 0x9ebe,
00701 0x6c08, 0x6c13, 0x6c14, 0x6c1b, 0x6c24, 0x6c23, 0x6c5e, 0x6c55,
00702 0x6c62, 0x6c6a, 0x6c82, 0x6c8d, 0x6c9a, 0x6c81, 0x6c9b, 0x6c7e,
00703 0x6c68, 0x6c73, 0x6c92, 0x6c90, 0x6cc4, 0x6cf1, 0x6cd3, 0x6cbd,
00704 0x6cd7, 0x6cc5, 0x6cdd, 0x6cae, 0x6cb1, 0x6cbe,
00705 /* 0x5e */
00706 0x6cba, 0x6cdb, 0x6cef, 0x6cd9, 0x6cea, 0x6dlf, 0x884d, 0x6d36,
00707 0x6d2b, 0x6d3d, 0x6d38, 0x6d19, 0x6d35, 0x6d33, 0x6d12, 0x6d0c,
00708 0x6d63, 0x6d93, 0x6d64, 0x6d5a, 0x6d79, 0x6d59, 0x6d8e, 0x6d95,
00709 0x6fe4, 0x6d85, 0x6df9, 0x6e15, 0x6e0a, 0x6db5, 0x6dc7, 0x6de6,
00710 0x6db8, 0x6dc6, 0x6dec, 0x6dde, 0x6dcc, 0x6de8, 0x6dd2, 0x6dc5,
00711 0x6dfa, 0x6dd9, 0x6de4, 0x6dd5, 0x6dea, 0x6dee, 0x6e2d, 0x6e6e,
00712 0x6e2e, 0x6e19, 0x6e72, 0x6e5f, 0x6e3e, 0x6e23, 0x6e6b, 0x6e2b,
00713 0x6e76, 0x6e4d, 0x6e1f, 0x6e43, 0x6e3a, 0x6e4e, 0x6e24, 0x6eff,
00714 0x6e1d, 0x6e38, 0x6e82, 0x6eaa, 0x6e98, 0x6ec9, 0x6eb7, 0x6ed3,
00715 0x6ebd, 0x6eaf, 0x6ec4, 0x6eb2, 0x6ed4, 0x6ed5, 0x6e8f, 0x6ea5,
00716 0x6ec2, 0x6e9f, 0x6f41, 0x6f11, 0x704c, 0x6eec, 0x6ef8, 0x6efe,
00717 0x6f3f, 0x6ef2, 0x6f31, 0x6eef, 0x6f32, 0x6ecc,
00718 /* 0x5f */
00719 0x6f3e, 0x6f13, 0x6ef7, 0x6f86, 0x6f7a, 0x6f78, 0x6f81, 0x6f80,
00720 0x6f6f, 0x6f5b, 0x6ff3, 0x6f6d, 0x6f82, 0x6f7c, 0x6f58, 0x6f8e,
00721 0x6f91, 0x6fc2, 0x6f66, 0x6fb3, 0x6fa3, 0x6fa1, 0x6fa4, 0x6fb9,
00722 0x6fc6, 0x6faa, 0x6fdf, 0x6fd5, 0x6fec, 0x6fd4, 0x6fd8, 0x6ff1,
00723 0x6fee, 0x6fdb, 0x7009, 0x700b, 0x6ffa, 0x7011, 0x7001, 0x700f,
00724 0x6ffe, 0x701b, 0x701a, 0x6f74, 0x701d, 0x7018, 0x701f, 0x7030,
00725 0x703e, 0x7032, 0x7051, 0x7063, 0x7099, 0x7092, 0x70a1, 0x70f1,
00726 0x70ac, 0x70b8, 0x70b3, 0x70ae, 0x70df, 0x70cb, 0x70dd, 0x70d9,
00727 0x7109, 0x70fd, 0x711c, 0x7119, 0x7165, 0x7155, 0x7188, 0x7166,
00728 0x7162, 0x714c, 0x7156, 0x716c, 0x718f, 0x71fb, 0x7184, 0x7195,
00729 0x71a8, 0x71ac, 0x71d7, 0x71b9, 0x71be, 0x71d2, 0x71c9, 0x71d4,
00730 0x71ce, 0x71e0, 0x71ec, 0x71e7, 0x71f5, 0x71fc,
00731 /* 0x60 */
00732 0x71f9, 0x71ff, 0x720d, 0x7210, 0x721b, 0x7228, 0x722d, 0x722c,
00733 0x7230, 0x7232, 0x723b, 0x723c, 0x723f, 0x7240, 0x7246, 0x724b,
00734 0x7258, 0x7274, 0x727e, 0x7282, 0x7281, 0x7287, 0x7292, 0x7296,
00735 0x72a2, 0x72a7, 0x72b9, 0x72b2, 0x72c3, 0x72c6, 0x72c4, 0x72ce,
00736 0x72d2, 0x72e2, 0x72e0, 0x72e1, 0x72f9, 0x72f7, 0x500f, 0x7317,
00737 0x730a, 0x731c, 0x7316, 0x731d, 0x7334, 0x732f, 0x7329, 0x7325,
00738 0x733e, 0x734e, 0x734f, 0x9ed8, 0x7357, 0x736a, 0x7368, 0x7370,
00739 0x7378, 0x7375, 0x737b, 0x737a, 0x73c8, 0x73b3, 0x73ce, 0x73bb,
00740 0x73c0, 0x73e5, 0x73ee, 0x73de, 0x74a2, 0x7405, 0x746f, 0x7425,
00741 0x73f8, 0x7432, 0x743a, 0x7455, 0x743f, 0x745f, 0x7459, 0x7441,
00742 0x745c, 0x7469, 0x7470, 0x7463, 0x746a, 0x7476, 0x747e, 0x748b,
00743 0x749e, 0x74a7, 0x74ca, 0x74cf, 0x74d4, 0x73f1,
00744 /* 0x61 */
00745 0x74e0, 0x74e3, 0x74e7, 0x74e9, 0x74ee, 0x74f2, 0x74f0, 0x74f1,
00746 0x74f8, 0x74f7, 0x7504, 0x7503, 0x7505, 0x750c, 0x750e, 0x750d,
00747 0x7515, 0x7513, 0x751e, 0x7526, 0x752c, 0x753c, 0x7544, 0x754d,
00748 0x754a, 0x7549, 0x754b, 0x7546, 0x755a, 0x7569, 0x7564, 0x7567,
00749 0x756b, 0x756d, 0x7578, 0x7576, 0x7586, 0x7587, 0x7574, 0x758a,
00750 0x7589, 0x7582, 0x7594, 0x759a, 0x759d, 0x75a5, 0x75a3, 0x75c2,
00751 0x75b3, 0x75c3, 0x75c3, 0x75b5, 0x75bd, 0x75b8, 0x75bc, 0x75b1, 0x75cd,
00752 0x75ca, 0x75d2, 0x75d9, 0x75e3, 0x75de, 0x75fe, 0x75ff, 0x75fc,
00753 0x7601, 0x75f0, 0x75fa, 0x75f2, 0x75f3, 0x760b, 0x760d, 0x7609,
00754 0x761f, 0x7627, 0x7620, 0x7621, 0x7622, 0x7624, 0x7634, 0x7630,
00755 0x763b, 0x7647, 0x7648, 0x7646, 0x765c, 0x7658, 0x7661, 0x7662,
```

```

00756 0x7668, 0x7669, 0x766a, 0x7667, 0x766c, 0x7670,
00757 /* 0x62 */
00758 0x7672, 0x7676, 0x7678, 0x767c, 0x7680, 0x7683, 0x7688, 0x768b,
00759 0x768e, 0x7696, 0x7693, 0x7699, 0x769a, 0x76b0, 0x76b4, 0x76b8,
00760 0x76b9, 0x76ba, 0x76c2, 0x76cd, 0x76d6, 0x76d2, 0x76de, 0x76e1,
00761 0x76e5, 0x76e7, 0x76ea, 0x862f, 0x76fb, 0x7708, 0x7707, 0x7704,
00762 0x7729, 0x7724, 0x771e, 0x7725, 0x7726, 0x771b, 0x7737, 0x7738,
00763 0x7747, 0x775a, 0x7768, 0x776b, 0x775b, 0x7765, 0x777f, 0x777e,
00764 0x7779, 0x778e, 0x778b, 0x7791, 0x77a0, 0x779e, 0x77b0, 0x77b6,
00765 0x77b9, 0x77bf, 0x77bc, 0x77bd, 0x77bb, 0x77c7, 0x77cd, 0x77d7,
00766 0x77da, 0x77dc, 0x77e3, 0x77ee, 0x77fc, 0x780c, 0x7812, 0x7926,
00767 0x7820, 0x792a, 0x7845, 0x788e, 0x7874, 0x7886, 0x787c, 0x789a,
00768 0x788c, 0x78a3, 0x78b5, 0x78aa, 0x78af, 0x78d1, 0x78c6, 0x78cb,
00769 0x78d4, 0x78be, 0x78bc, 0x78c5, 0x78ca, 0x78ec,
00770 /* 0x63 */
00771 0x78e7, 0x78da, 0x78fd, 0x78f4, 0x7907, 0x7912, 0x7911, 0x7919,
00772 0x792c, 0x792b, 0x7940, 0x7960, 0x7957, 0x795f, 0x795a, 0x7955,
00773 0x7953, 0x797a, 0x797f, 0x798a, 0x799d, 0x79a7, 0x9f4b, 0x79aa,
00774 0x79ae, 0x79b3, 0x79b9, 0x79ba, 0x79c9, 0x79d5, 0x79e7, 0x79ec,
00775 0x79e1, 0x79e3, 0x7a08, 0x7a0d, 0x7a18, 0x7a19, 0x7a20, 0x7a1f,
00776 0x7980, 0x7a31, 0x7a3b, 0x7a3e, 0x7a37, 0x7a43, 0x7a57, 0x7a49,
00777 0x7a61, 0x7a62, 0x7a69, 0x9f9d, 0x7a70, 0x7a79, 0x7a7d, 0x7a88,
00778 0x7a97, 0x7a95, 0x7a98, 0x7a96, 0x7aa9, 0x7ac8, 0x7ab0, 0x7ab6,
00779 0x7ac5, 0x7ac4, 0x7abf, 0x9083, 0x7ac7, 0x7aca, 0x7acd, 0x7acf,
00780 0x7ad5, 0x7ad3, 0x7ad9, 0x7ada, 0x7add, 0x7ae1, 0x7ae2, 0x7ae6,
00781 0x7aed, 0x7af0, 0x7b02, 0x7b0f, 0x7b0a, 0x7b06, 0x7b33, 0x7b18,
00782 0x7b19, 0x7b1e, 0x7b35, 0x7b28, 0x7b36, 0x7b50,
00783 /* 0x64 */
00784 0x7b7a, 0x7b04, 0x7b4d, 0x7b0b, 0x7b4c, 0x7b45, 0x7b75, 0x7b65,
00785 0x7b74, 0x7b67, 0x7b70, 0x7b71, 0x7b6c, 0x7b6e, 0x7b9d, 0x7b98,
00786 0x7b9f, 0x7b8d, 0x7b9c, 0x7b9a, 0x7b8b, 0x7b92, 0x7b8f, 0x7b5d,
00787 0x7b99, 0x7bcb, 0x7bc1, 0x7bcc, 0x7bcf, 0x7bb4, 0x7bc6, 0x7bdd,
00788 0x7be9, 0x7c11, 0x7c14, 0x7be6, 0x7be5, 0x7c60, 0x7c00, 0x7c07,
00789 0x7c13, 0x7bf3, 0x7bf7, 0x7c17, 0x7c0d, 0x7bf6, 0x7c23, 0x7c27,
00790 0x7c2a, 0x7c1f, 0x7c37, 0x7c2b, 0x7c3d, 0x7c4c, 0x7c43, 0x7c54,
00791 0x7c4f, 0x7c40, 0x7c50, 0x7c58, 0x7c5f, 0x7c64, 0x7c56, 0x7c65,
00792 0x7c6c, 0x7c75, 0x7c83, 0x7c90, 0x7ca4, 0x7cad, 0x7ca2, 0x7cab,
00793 0x7ca1, 0x7ca8, 0x7cb3, 0x7cb2, 0x7cb1, 0x7cae, 0x7cb9, 0x7cbd,
00794 0x7cc0, 0x7cc5, 0x7cc2, 0x7cd8, 0x7cd2, 0x7cdc, 0x7ce2, 0x9b3b,
00795 0x7cef, 0x7cf2, 0x7cf4, 0x7cf6, 0x7cfa, 0x7d06,
00796 /* 0x65 */
00797 0x7d02, 0x7d1c, 0x7d15, 0x7d0a, 0x7d45, 0x7d4b, 0x7d2e, 0x7d32,
00798 0x7d3f, 0x7d35, 0x7d46, 0x7d73, 0x7d56, 0x7d4e, 0x7d72, 0x7d68,
00799 0x7d6e, 0x7d4f, 0x7d63, 0x7d93, 0x7d89, 0x7d5b, 0x7d8f, 0x7d7d,
00800 0x7d9b, 0x7dba, 0x7dae, 0x7da3, 0x7db5, 0x7dc7, 0x7dbd, 0x7dab,
00801 0x7e3d, 0x7da2, 0x7daf, 0x7ddc, 0x7db8, 0x7d9f, 0x7db0, 0x7dd8,
00802 0x7ddd, 0x7de4, 0x7dde, 0x7dfb, 0x7df2, 0x7de1, 0x7e05, 0x7e0a,
00803 0x7e23, 0x7e21, 0x7e12, 0x7e1f, 0x7e09, 0x7e0b, 0x7e22,
00804 0x7e46, 0x7e66, 0x7e3b, 0x7e35, 0x7e39, 0x7e43, 0x7e37, 0x7e32,
00805 0x7e3a, 0x7e67, 0x7e5d, 0x7e56, 0x7e5e, 0x7e59, 0x7e5a, 0x7e79,
00806 0x7e6a, 0x7ef9, 0x7e7c, 0x7e7b, 0x7e83, 0x7dd5, 0x7e7d, 0x8fae,
00807 0x7e7f, 0x7e88, 0x7e89, 0x7e8c, 0x7e92, 0x7e90, 0x7e93, 0x7e94,
00808 0x7e96, 0x7e8e, 0x7e9b, 0x7e9c, 0x7f38, 0x7f3a,
00809 /* 0x66 */
00810 0x7f45, 0x7f4c, 0x7f4d, 0x7f4e, 0x7f50, 0x7f51, 0x7f55, 0x7f54,
00811 0x7f58, 0x7f5f, 0x7f60, 0x7f68, 0x7f69, 0x7f67, 0x7f78, 0x7f82,
00812 0x7f86, 0x7f83, 0x7f88, 0x7f87, 0x7f8c, 0x7f94, 0x7f9e, 0x7f9d,
00813 0x7f9a, 0x7fa3, 0x7faf, 0x7fb2, 0x7fb9, 0x7fae, 0x7fb6, 0x7fb8,
00814 0x8b71, 0x7fc5, 0x7fc6, 0x7fca, 0x7fd5, 0x7fd4, 0x7fe1, 0x7fe6,
00815 0x7fe9, 0x7ff3, 0x7ff9, 0x98dc, 0x8006, 0x8004, 0x800b, 0x8012,
00816 0x8018, 0x8019, 0x801c, 0x8021, 0x8028, 0x803f, 0x803b, 0x804a,
00817 0x8046, 0x8052, 0x8058, 0x805a, 0x805f, 0x8062, 0x8068, 0x8073,
00818 0x8072, 0x807d, 0x8076, 0x8079, 0x807f, 0x8084, 0x8086,
00819 0x8085, 0x809b, 0x8093, 0x809a, 0x80ad, 0x5190, 0x80ac, 0x80db,
00820 0x80e5, 0x80d9, 0x80dd, 0x80c4, 0x80da, 0x80d6, 0x8109, 0x80ef,
00821 0x80f1, 0x811b, 0x8129, 0x8123, 0x812f, 0x814b,
00822 /* 0x67 */
00823 0x968b, 0x8146, 0x813e, 0x8153, 0x8151, 0x80fc, 0x8171, 0x816e,
00824 0x8165, 0x8166, 0x8174, 0x8183, 0x8188, 0x818a, 0x8180, 0x8182,
00825 0x81a0, 0x8195, 0x81a4, 0x81a3, 0x815f, 0x8193, 0x81a9, 0x81b0,
00826 0x81b5, 0x81be, 0x81b8, 0x81bd, 0x81c0, 0x81c2, 0x81ba, 0x81c9,
00827 0x81cd, 0x81d1, 0x81d9, 0x81d8, 0x81c8, 0x81da, 0x81df, 0x81e0,
00828 0x81e7, 0x81fa, 0x81fb, 0x81fe, 0x8201, 0x8202, 0x8205, 0x8207,
00829 0x820a, 0x820d, 0x8210, 0x8216, 0x8229, 0x822b, 0x8238, 0x8233,
00830 0x8240, 0x8259, 0x8258, 0x825d, 0x825a, 0x825f, 0x8264, 0x8262,
00831 0x8268, 0x826a, 0x826b, 0x822e, 0x8271, 0x8277, 0x8278, 0x827e,
00832 0x828d, 0x8292, 0x82ab, 0x829f, 0x82bb, 0x82ac, 0x82e1, 0x82e3,
00833 0x82df, 0x82d2, 0x82f4, 0x82f3, 0x82fa, 0x8393, 0x8303, 0x82fb,
00834 0x82f9, 0x82de, 0x8306, 0x82dc, 0x8309, 0x82d9,
00835 /* 0x68 */
00836 0x8335, 0x8334, 0x8316, 0x8332, 0x8331, 0x8340, 0x8339, 0x8350,
00837 0x8345, 0x832f, 0x832b, 0x8317, 0x8318, 0x8385, 0x839a, 0x83aa,
00838 0x839f, 0x83a2, 0x8396, 0x8323, 0x838e, 0x8387, 0x838a, 0x837c,
00839 0x83b5, 0x8373, 0x8375, 0x83a0, 0x8389, 0x83a8, 0x83f4, 0x8413,
00840 0x83eb, 0x83ce, 0x83fd, 0x8403, 0x83d8, 0x840b, 0x83c1, 0x83f7,
00841 0x8407, 0x83e0, 0x83f2, 0x840d, 0x8422, 0x8420, 0x83bd, 0x8438,
00842 0x8506, 0x83fb, 0x846d, 0x842a, 0x843c, 0x855a, 0x8484, 0x8477,

```

```
00843 0x846b, 0x84ad, 0x846e, 0x8482, 0x8469, 0x8446, 0x842c, 0x846f,
00844 0x8479, 0x8433, 0x84ca, 0x8462, 0x84b9, 0x84bf, 0x849f, 0x84d9,
00845 0x84cd, 0x84bb, 0x84da, 0x84d0, 0x84c1, 0x84c6, 0x84d6, 0x84a1,
00846 0x8521, 0x84ff, 0x84f4, 0x8517, 0x8518, 0x852c, 0x851f, 0x8515,
00847 0x8514, 0x84fc, 0x8540, 0x8563, 0x8558, 0x8548,
00848 /* 0x69 */
00849 0x8541, 0x8602, 0x854b, 0x8555, 0x8580, 0x85a4, 0x8588, 0x8591,
00850 0x858a, 0x85a8, 0x856d, 0x8594, 0x859b, 0x85ea, 0x8587, 0x859c,
00851 0x8577, 0x857e, 0x8590, 0x85c9, 0x85ba, 0x85cf, 0x85b9, 0x85d0,
00852 0x85d5, 0x85dd, 0x85e5, 0x85dc, 0x85f9, 0x860a, 0x8613, 0x860b,
00853 0x85fe, 0x85fa, 0x8606, 0x8622, 0x861a, 0x8630, 0x863f, 0x864d,
00854 0x4e55, 0x8654, 0x865f, 0x8667, 0x8671, 0x8693, 0x86a3, 0x86a9,
00855 0x86aa, 0x868b, 0x868c, 0x86b6, 0x86af, 0x86c4, 0x86c6, 0x86b0,
00856 0x86c9, 0x8823, 0x86ab, 0x86d4, 0x86de, 0x86e9, 0x86ec, 0x86df,
00857 0x86db, 0x86ef, 0x8712, 0x8706, 0x8708, 0x8700, 0x8703, 0x86fb,
00858 0x8711, 0x8709, 0x870d, 0x86f9, 0x870a, 0x8734, 0x873f, 0x8737,
00859 0x873b, 0x8725, 0x8729, 0x871a, 0x8760, 0x875f, 0x8778, 0x874c,
00860 0x874e, 0x8774, 0x8757, 0x8768, 0x876e, 0x8759,
00861 /* 0x6a */
00862 0x8753, 0x8763, 0x876a, 0x8805, 0x87a2, 0x879f, 0x8782, 0x87af,
00863 0x87cb, 0x87bd, 0x87c0, 0x87d0, 0x96d6, 0x87ab, 0x87c4, 0x87b3,
00864 0x87c7, 0x87c6, 0x87bb, 0x87ef, 0x87f2, 0x87e0, 0x880f, 0x880d,
00865 0x87fe, 0x87f6, 0x87f7, 0x880e, 0x87d2, 0x8811, 0x8816, 0x8815,
00866 0x8822, 0x8821, 0x8831, 0x8836, 0x8839, 0x8827, 0x883b, 0x8844,
00867 0x8842, 0x8852, 0x8859, 0x885e, 0x8862, 0x886b, 0x8881, 0x887e,
00868 0x889e, 0x8875, 0x887d, 0x88b5, 0x8872, 0x8882, 0x8897, 0x8892,
00869 0x88ae, 0x8899, 0x88a2, 0x888d, 0x88a4, 0x88b0, 0x88bf, 0x88b1,
00870 0x88c3, 0x88c4, 0x88d4, 0x88d8, 0x88d9, 0x88dd, 0x88f9, 0x8902,
00871 0x88fc, 0x88f4, 0x88e8, 0x88f2, 0x8904, 0x890c, 0x890a, 0x8913,
00872 0x8943, 0x891e, 0x8925, 0x892a, 0x892b, 0x8941, 0x8944, 0x893b,
00873 0x8936, 0x8938, 0x894c, 0x891d, 0x8960, 0x895e,
00874 /* 0x6b */
00875 0x8966, 0x8964, 0x896d, 0x896a, 0x896f, 0x8974, 0x8977, 0x897e,
00876 0x8983, 0x8988, 0x898a, 0x8993, 0x8998, 0x89a1, 0x89a9, 0x89a6,
00877 0x89ac, 0x89af, 0x89b2, 0x89ba, 0x89bd, 0x89bf, 0x89c0, 0x89da,
00878 0x89dc, 0x89dd, 0x89e7, 0x89f4, 0x89f8, 0x8a03, 0x8a16, 0x8a10,
00879 0x8a0c, 0x8a1b, 0x8a1d, 0x8a25, 0x8a36, 0x8a41, 0x8a5b, 0x8a52,
00880 0x8a46, 0x8a48, 0x8a7c, 0x8a6d, 0x8a6c, 0x8a62, 0x8a85, 0x8a82,
00881 0x8a84, 0x8aa8, 0x8aa1, 0x8aa9, 0x8aa6, 0x8aa9, 0x8aa3,
00882 0x8ac4, 0x8acd, 0x8ac2, 0x8ada, 0x8aeb, 0x8af3, 0x8ae7, 0x8ae4,
00883 0x8af1, 0x8b14, 0x8ae0, 0x8ae2, 0x8af7, 0x8ade, 0x8adb, 0x8b0c,
00884 0x8b07, 0x8b1a, 0x8ae1, 0x8b16, 0x8b10, 0x8b17, 0x8b20, 0x8b33,
00885 0x97ab, 0x8b26, 0x8b2b, 0x8b3e, 0x8b28, 0x8b41, 0x8b4c, 0x8b4f,
00886 0x8b4e, 0x8b49, 0x8b56, 0x8b5b, 0x8b5a, 0x8b6b,
00887 /* 0x6c */
00888 0x8b5f, 0x8b6c, 0x8b6f, 0x8b74, 0x8b7d, 0x8b80, 0x8b8c, 0x8b8e,
00889 0x8b92, 0x8b93, 0x8b96, 0x8b99, 0x8b9a, 0x8c3a, 0x8c41, 0x8c3f,
00890 0x8c48, 0x8c4c, 0x8c4e, 0x8c50, 0x8c55, 0x8c62, 0x8c6c, 0x8c78,
00891 0x8c7a, 0x8c82, 0x8c89, 0x8c85, 0x8c8a, 0x8c8d, 0x8c8e, 0x8c94,
00892 0x8c7c, 0x8c98, 0x621d, 0x8cad, 0x8caa, 0x8cbd, 0x8cb2, 0x8cb3,
00893 0x8cae, 0x8cb6, 0x8cc8, 0x8cc1, 0x8cce4, 0x8ce3, 0x8cda, 0x8cdf,
00894 0x8cfa, 0x8cfb, 0x8d04, 0x8d05, 0x8d0a, 0x8d07, 0x8d0f, 0x8d0d,
00895 0x8d10, 0x9f4e, 0x8d13, 0x8ccd, 0x8d14, 0x8d16, 0x8d67, 0x8d6d,
00896 0x8d71, 0x8d73, 0x8d81, 0x8d99, 0x8dc2, 0x8dbe, 0x8dba, 0x8dcf,
00897 0x8dda, 0x8dd6, 0x8dcc, 0x8ddb, 0x8dcb, 0x8dea, 0x8deb, 0x8ddf,
00898 0x8de3, 0x8dfc, 0x8e08, 0x8e09, 0x8dff, 0x8e1d, 0x8e1e, 0x8e10,
00899 0x8e1f, 0x8e42, 0x8e0f, 0x8e35, 0x8e30, 0x8e34, 0x8e4a,
00900 /* 0x6d */
00901 0x8e47, 0x8e49, 0x8e4c, 0x8e50, 0x8e48, 0x8e59, 0x8e64, 0x8e60,
00902 0x8e2a, 0x8e63, 0x8e55, 0x8e76, 0x8e72, 0x8e7c, 0x8e81, 0x8e87,
00903 0x8e85, 0x8e84, 0x8e8b, 0x8e8a, 0x8e93, 0x8e91, 0x8e94, 0x8e99,
00904 0x8eaa, 0x8eal, 0x8eac, 0x8eb0, 0x8ec6, 0x8eb1, 0x8ebe, 0x8ec5,
00905 0x8ec8, 0x8ecb, 0x8edb, 0x8ee3, 0x8efc, 0x8efb, 0x8eeb, 0x8efe,
00906 0x8f0a, 0x8f05, 0x8f15, 0x8f12, 0x8f19, 0x8f13, 0x8f1c, 0x8f1f,
00907 0x8f1b, 0x8f0c, 0x8f26, 0x8f33, 0x8f3b, 0x8f39, 0x8f45, 0x8f42,
00908 0x8f3e, 0x8f4c, 0x8f49, 0x8f46, 0x8f4e, 0x8f57, 0x8f5c, 0x8f62,
00909 0x8f63, 0x8f64, 0x8f9c, 0x8f9f, 0x8fa3, 0x8fad, 0x8faf, 0x8fb7,
00910 0x8fda, 0x8fe5, 0x8fe2, 0x8fea, 0x8fef, 0x9087, 0x8ff4, 0x9005,
00911 0x8ff9, 0x8ffa, 0x9011, 0x9015, 0x9021, 0x900d, 0x901e, 0x9016,
00912 0x900b, 0x9027, 0x9036, 0x9035, 0x9039, 0x8ff8,
00913 /* 0x6e */
00914 0x904f, 0x9050, 0x9051, 0x9052, 0x900e, 0x9049, 0x903e, 0x9056,
00915 0x9058, 0x905e, 0x9068, 0x906f, 0x9076, 0x96a8, 0x9072, 0x9082,
00916 0x907d, 0x9081, 0x9080, 0x908a, 0x9089, 0x908f, 0x90a8, 0x90af,
00917 0x90b1, 0x90b5, 0x90e2, 0x90e4, 0x6248, 0x90db, 0x9102, 0x9112,
00918 0x9119, 0x9132, 0x9130, 0x914a, 0x9156, 0x9158, 0x9163, 0x9165,
00919 0x9169, 0x9173, 0x9172, 0x917c, 0x918b, 0x9189, 0x9182, 0x91a2, 0x91ab,
00920 0x91af, 0x91aa, 0x91b5, 0x91b4, 0x91ba, 0x91c0, 0x91c1, 0x91c9,
00921 0x91cb, 0x91d0, 0x91d6, 0x91df, 0x91e1, 0x91db, 0x91fc, 0x91f5,
00922 0x91f6, 0x921e, 0x921f, 0x9214, 0x922c, 0x9215, 0x9211, 0x922e,
00923 0x9257, 0x9245, 0x9249, 0x9264, 0x9248, 0x9295, 0x923f, 0x924b,
00924 0x9250, 0x929c, 0x9296, 0x9293, 0x929b, 0x925a, 0x92cf, 0x92b9,
00925 0x92b7, 0x92e9, 0x930f, 0x92fa, 0x9344, 0x932e,
00926 /* 0x6f */
00927 0x9319, 0x9322, 0x931a, 0x9323, 0x933a, 0x9335, 0x933b, 0x933c,
00928 0x9360, 0x937c, 0x936e, 0x9356, 0x93b0, 0x93ac, 0x93ad, 0x9394,
00929 0x93b9, 0x93d6, 0x93d7, 0x93e8, 0x93e5, 0x93d8, 0x93c3, 0x93dd,
```

```

00930 0x93d0, 0x93c8, 0x93e4, 0x941a, 0x9414, 0x9413, 0x9403, 0x9407,
00931 0x9410, 0x9436, 0x942b, 0x9435, 0x9421, 0x943a, 0x9441, 0x9452,
00932 0x9444, 0x945b, 0x9460, 0x9462, 0x945e, 0x946a, 0x9229, 0x9470,
00933 0x9475, 0x9477, 0x947d, 0x945a, 0x947c, 0x947e, 0x9481, 0x947f,
00934 0x9582, 0x9587, 0x958a, 0x9594, 0x9596, 0x9598, 0x9599, 0x95a0,
00935 0x95a8, 0x95a7, 0x95ad, 0x95bc, 0x95bb, 0x95b9, 0x95be, 0x95ca,
00936 0x6ff6, 0x95c3, 0x95cd, 0x95cc, 0x95d5, 0x95d4, 0x95d6, 0x95dc,
00937 0x95e1, 0x95e5, 0x95e2, 0x9621, 0x9628, 0x962e, 0x962f, 0x9642,
00938 0x964c, 0x964f, 0x964b, 0x9677, 0x965c, 0x965e,
00939 /* 0x70 */
00940 0x965d, 0x965f, 0x9666, 0x9672, 0x966c, 0x968d, 0x9698, 0x9695,
00941 0x9697, 0x96aa, 0x96a7, 0x96b1, 0x96b2, 0x96b0, 0x96b4, 0x96b6,
00942 0x96b8, 0x96b9, 0x96ce, 0x96cb, 0x96c9, 0x96cd, 0x894d, 0x96dc,
00943 0x970d, 0x96d5, 0x96f9, 0x9704, 0x9706, 0x9708, 0x9713, 0x970e,
00944 0x9711, 0x970f, 0x9716, 0x9719, 0x9724, 0x972a, 0x9730, 0x9739,
00945 0x973d, 0x973e, 0x9744, 0x9746, 0x9748, 0x9742, 0x9749, 0x975c,
00946 0x9760, 0x9764, 0x9766, 0x9768, 0x52d2, 0x976b, 0x9771, 0x9779,
00947 0x9785, 0x977c, 0x9781, 0x977a, 0x9786, 0x978b, 0x978f, 0x9790,
00948 0x979c, 0x97a8, 0x97a6, 0x97a3, 0x97b3, 0x97b4, 0x97c3, 0x97c6,
00949 0x97c8, 0x97cb, 0x97dc, 0x97ed, 0x97f4, 0x97f2, 0x7adf, 0x97f6,
00950 0x97f5, 0x980f, 0x980c, 0x9838, 0x9824, 0x9821, 0x9837, 0x983d,
00951 0x9846, 0x984f, 0x984b, 0x986b, 0x986f, 0x9870,
00952 /* 0x71 */
00953 0x9871, 0x9874, 0x9873, 0x98aa, 0x98af, 0x98b1, 0x98b6, 0x98c4,
00954 0x98c3, 0x98c6, 0x98e9, 0x98eb, 0x9903, 0x9909, 0x9912, 0x9914,
00955 0x9918, 0x9921, 0x991d, 0x991e, 0x9924, 0x9920, 0x992c, 0x992e,
00956 0x993d, 0x993b, 0x9942, 0x9949, 0x9945, 0x9950, 0x994b, 0x9951,
00957 0x9952, 0x994c, 0x9955, 0x9997, 0x9998, 0x99a5, 0x99ad, 0x99ae,
00958 0x99bc, 0x99df, 0x99db, 0x99dd, 0x99d8, 0x99d1, 0x99ed, 0x99ee,
00959 0x99f1, 0x99f2, 0x99fb, 0x99f8, 0x9a01, 0x9a0f, 0x9a05, 0x99e2,
00960 0x9a19, 0x9a2b, 0x9a37, 0x9a45, 0x9a42, 0x9a40, 0x9a43, 0x9a3e,
00961 0x9a55, 0x9a4d, 0x9a5b, 0x9a57, 0x9a5f, 0x9a62, 0x9a65, 0x9a64,
00962 0x9a69, 0x9a6b, 0x9a6a, 0x9aad, 0x9ab0, 0x9abc, 0x9ac0, 0x9acf,
00963 0x9ad1, 0x9ad3, 0x9ad4, 0x9ade, 0x9adf, 0x9ae2, 0x9ae3, 0x9ae6,
00964 0x9aef, 0x9aeb, 0x9aee, 0x9af4, 0x9af1, 0x9af7,
00965 /* 0x72 */
00966 0x9afb, 0x9b06, 0x9b18, 0x9b1a, 0x9b1f, 0x9b22, 0x9b23, 0x9b25,
00967 0x9b27, 0x9b28, 0x9b29, 0x9b2a, 0x9b2e, 0x9b2f, 0x9b32, 0x9b44,
00968 0x9b43, 0x9b4b, 0x9b4d, 0x9b4e, 0x9b51, 0x9b58, 0x9b74, 0x9b93,
00969 0x9b83, 0x9b91, 0x9b96, 0x9b97, 0x9b9f, 0x9ba0, 0x9ba8, 0x9bb4,
00970 0x9bc0, 0x9bca, 0x9bb9, 0x9bc6, 0x9bcf, 0x9bd1, 0x9bd2, 0x9be3,
00971 0x9be2, 0x9be4, 0x9bd4, 0x9be1, 0x9c3a, 0x9bf2, 0x9bf1, 0x9bf0,
00972 0x9c15, 0x9c14, 0x9c09, 0x9c13, 0x9c0c, 0x9c06, 0x9c08, 0x9c12,
00973 0x9c0a, 0x9c04, 0x9c2e, 0x9c1b, 0x9c25, 0x9c24, 0x9c21, 0x9c30,
00974 0x9c47, 0x9c32, 0x9c46, 0x9c3e, 0x9c5a, 0x9c60, 0x9c67, 0x9c76,
00975 0x9c78, 0x9ce7, 0x9cec, 0x9cf0, 0x9d09, 0x9d08, 0x9ceb, 0x9d03,
00976 0x9d06, 0x9d2a, 0x9d26, 0x9daf, 0x9d23, 0x9d1f, 0x9d44, 0x9d15,
00977 0x9d12, 0x9d41, 0x9d3f, 0x9d3e, 0x9d46, 0x9d48,
00978 /* 0x73 */
00979 0x9d5d, 0x9d5e, 0x9d64, 0x9d51, 0x9d50, 0x9d59, 0x9d72, 0x9d89,
00980 0x9d87, 0x9dab, 0x9d6f, 0x9d7a, 0x9d9a, 0x9da4, 0x9da9, 0x9db2,
00981 0x9dc4, 0x9dc1, 0x9dbb, 0x9db8, 0x9dba, 0x9dc6, 0x9dcf, 0x9dc2,
00982 0x9dd9, 0x9dd3, 0x9df8, 0x9de6, 0x9ded, 0x9def, 0x9dfd, 0x9e1a,
00983 0x9e1b, 0x9e1e, 0x9e75, 0x9e79, 0x9e7d, 0x9e81, 0x9e88, 0x9e8b,
00984 0x9e8c, 0x9e92, 0x9e95, 0x9e91, 0x9e9d, 0x9ea5, 0x9ea9, 0x9eb8,
00985 0x9eaa, 0x9ead, 0x9761, 0x9ecc, 0x9ece, 0x9ecf, 0x9ed0, 0x9ed4,
00986 0x9edc, 0x9ede, 0x9edd, 0x9ee0, 0x9ee5, 0x9ee8, 0x9ee9, 0x9ef4,
00987 0x9ef6, 0x9ef7, 0x9ef9, 0x9efb, 0x9efc, 0x9efd, 0x9f07, 0x9f08,
00988 0x76b7, 0x9f15, 0x9f21, 0x9f2c, 0x9f3e, 0x9f4a, 0x9f52, 0x9f54,
00989 0x9f63, 0x9f5f, 0x9f60, 0x9f61, 0x9f66, 0x9f67, 0x9f6c, 0x9f6a,
00990 0x9f77, 0x9f72, 0x9f76, 0x9f95, 0x9f9c, 0x9fa0,
00991 /* 0x74 */
00992 0x582f, 0x69c7, 0x9059, 0x7464, 0x51dc, 0x7199,
00993 };
00994
00995 static int
00996 jisx0208_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00997 {
00998     unsigned char c1 = (s[0] & 0x7f);
00999     if ((c1 >= 0x21 && c1 <= 0x28) || (c1 >= 0x30 && c1 <= 0x74)) {
01000         if (n >= 2) {
01001             unsigned char c2 = (s[1] & 0x7f);
01002             if (c2 >= 0x21 && c2 < 0x7f) {
01003                 unsigned int i = 94 * (c1 - 0x21) + (c2 - 0x21);
01004                 unsigned short wc = 0xffff;
01005                 if (i < 1410) {
01006                     if (i < 690)
01007                         wc = jisx0208_2uni_page21[i];
01008                     else {
01009                         if (i < 7808)
01010                             wc = jisx0208_2uni_page30[i-1410];
01011                     }
01012                     if (wc != 0xffff) {
01013                         *pwc = (ucs4_t) wc;
01014                         return 2;
01015                     }
01016                 }

```

```
01017     return RET_ILSEQ;
01018 }
01019     return RET_TOOFEW(0);
01020 }
01021     return RET_ILSEQ;
01022 }
01023 #endif /* NEED_TOWC */
01024
01025 #ifdef NEED_TOMB
01026 static const unsigned short jisx0208_2charset[6879] = {
01027     0x2140, 0x2171, 0x2172, 0x2178, 0x212f, 0x224c, 0x216b, 0x215e,
01028     0x212d, 0x2279, 0x215f, 0x2160, 0x2621, 0x2622, 0x2623, 0x2624,
01029     0x2625, 0x2626, 0x2627, 0x2628, 0x2629, 0x262a, 0x262b, 0x262c,
01030     0x262d, 0x262e, 0x262f, 0x2630, 0x2631, 0x2632, 0x2633, 0x2634,
01031     0x2635, 0x2636, 0x2637, 0x2638, 0x2641, 0x2642, 0x2643, 0x2644,
01032     0x2645, 0x2646, 0x2647, 0x2648, 0x2649, 0x264a, 0x264b, 0x264c,
01033     0x264d, 0x264e, 0x264f, 0x2650, 0x2651, 0x2652, 0x2653, 0x2654,
01034     0x2655, 0x2656, 0x2657, 0x2658, 0x2727, 0x2721, 0x2722, 0x2723,
01035     0x2724, 0x2725, 0x2726, 0x2728, 0x2729, 0x272a, 0x272b, 0x272c,
01036     0x272d, 0x272e, 0x272f, 0x2730, 0x2731, 0x2732, 0x2733, 0x2734,
01037     0x2735, 0x2736, 0x2737, 0x2738, 0x2739, 0x273a, 0x273b, 0x273c,
01038     0x273d, 0x273e, 0x273f, 0x2740, 0x2741, 0x2751, 0x2752, 0x2753,
01039     0x2754, 0x2755, 0x2756, 0x2758, 0x2759, 0x275a, 0x275b, 0x275c,
01040     0x275d, 0x275e, 0x275f, 0x2760, 0x2761, 0x2762, 0x2763, 0x2764,
01041     0x2765, 0x2766, 0x2767, 0x2768, 0x2769, 0x276a, 0x276b, 0x276c,
01042     0x276d, 0x276e, 0x276f, 0x2770, 0x2771, 0x2757, 0x213e, 0x213d,
01043     0x2142, 0x2146, 0x2147, 0x2148, 0x2149, 0x2277, 0x2278, 0x2145,
01044     0x2144, 0x2273, 0x216c, 0x216d, 0x2228, 0x216e, 0x2272, 0x222b,
01045     0x222c, 0x222a, 0x222d, 0x224d, 0x224e, 0x224f, 0x225f, 0x2250,
01046     0x2260, 0x223a, 0x223b, 0x215d, 0x2265, 0x2267, 0x2167, 0x222c,
01047     0x224a, 0x224b, 0x2241, 0x2240, 0x2269, 0x226a, 0x2168, 0x2268,
01048     0x2266, 0x2262, 0x2162, 0x2261, 0x2165, 0x2166, 0x2263, 0x2264,
01049     0x223e, 0x223f, 0x223c, 0x223d, 0x225d, 0x225e, 0x2821, 0x282c,
01050     0x2822, 0x282d, 0x2823, 0x282e, 0x2824, 0x282f, 0x2826, 0x2831,
01051     0x2825, 0x2830, 0x2827, 0x283c, 0x2837, 0x2832, 0x2829, 0x283e,
01052     0x2839, 0x2834, 0x2828, 0x2838, 0x283d, 0x2833, 0x282a, 0x283a,
01053     0x283f, 0x2835, 0x282b, 0x283b, 0x2840, 0x2836, 0x2223, 0x2222,
01054     0x2225, 0x2224, 0x2227, 0x2226, 0x2221, 0x217e, 0x217b, 0x217d,
01055     0x217c, 0x227e, 0x217a, 0x2179, 0x216a, 0x2169, 0x2276, 0x2275,
01056     0x2274, 0x2121, 0x2122, 0x2123, 0x2137, 0x2139, 0x213a, 0x213b,
01057     0x2152, 0x2153, 0x2154, 0x2155, 0x2156, 0x2157, 0x2158, 0x2159,
01058     0x215a, 0x215b, 0x2229, 0x222e, 0x214c, 0x214d, 0x2141, 0x2421,
01059     0x2422, 0x2423, 0x2424, 0x2425, 0x2426, 0x2427, 0x2428, 0x2429,
01060     0x242a, 0x242b, 0x242c, 0x242d, 0x242e, 0x242f, 0x2430, 0x2431,
01061     0x2432, 0x2433, 0x2434, 0x2435, 0x2436, 0x2437, 0x2438, 0x2439,
01062     0x243a, 0x243b, 0x243c, 0x243d, 0x243e, 0x243f, 0x2440, 0x2441,
01063     0x2442, 0x2443, 0x2444, 0x2445, 0x2446, 0x2447, 0x2448, 0x2449,
01064     0x244a, 0x244b, 0x244c, 0x244d, 0x244e, 0x244f, 0x2450, 0x2451,
01065     0x2452, 0x2453, 0x2454, 0x2455, 0x2456, 0x2457, 0x2458, 0x2459,
01066     0x245a, 0x245b, 0x245c, 0x245d, 0x245e, 0x245f, 0x2460, 0x2461,
01067     0x2462, 0x2463, 0x2464, 0x2465, 0x2466, 0x2467, 0x2468, 0x2469,
01068     0x246a, 0x246b, 0x246c, 0x246d, 0x246e, 0x246f, 0x2470, 0x2471,
01069     0x2472, 0x2473, 0x212b, 0x212c, 0x2135, 0x2136, 0x2521, 0x2522,
01070     0x2523, 0x2524, 0x2525, 0x2526, 0x2527, 0x2528, 0x2529, 0x252a,
01071     0x252b, 0x252c, 0x252d, 0x252e, 0x252f, 0x2530, 0x2531, 0x2532,
01072     0x2533, 0x2534, 0x2535, 0x2536, 0x2537, 0x2538, 0x2539, 0x253a,
01073     0x253b, 0x253c, 0x253d, 0x253e, 0x253f, 0x2540, 0x2541, 0x2542,
01074     0x2543, 0x2544, 0x2545, 0x2546, 0x2547, 0x2548, 0x2549, 0x254a,
01075     0x254b, 0x254c, 0x254d, 0x254e, 0x254f, 0x2550, 0x2551, 0x2552,
01076     0x2553, 0x2554, 0x2555, 0x2556, 0x2557, 0x2558, 0x2559, 0x255a,
01077     0x255b, 0x255c, 0x255d, 0x255e, 0x255f, 0x2560, 0x2561, 0x2562,
01078     0x2563, 0x2564, 0x2565, 0x2566, 0x2567, 0x2568, 0x2569, 0x256a,
01079     0x256b, 0x256c, 0x256d, 0x256e, 0x256f, 0x2570, 0x2571, 0x2572,
01080     0x2573, 0x2574, 0x2575, 0x2576, 0x2126, 0x213c, 0x2133, 0x2134,
01081     0x306c, 0x437a, 0x3c37, 0x4b7c, 0x3e66, 0x3b30, 0x3e65, 0x323c,
01082     0x4954, 0x4d3f, 0x5022, 0x312f, 0x336e, 0x5023, 0x4024, 0x5242,
01083     0x3556, 0x4a3a, 0x3e67, 0x4e3e, 0x4a42, 0x5024, 0x4366, 0x5025,
01084     0x367a, 0x5026, 0x345d, 0x4330, 0x3c67, 0x5027, 0x5028, 0x5029,
01085     0x4735, 0x3557, 0x4737, 0x4663, 0x3843, 0x4b33, 0x6949, 0x502a,
01086     0x3e68, 0x502b, 0x3235, 0x3665, 0x3870, 0x4c69, 0x5626, 0x4d70,
01087     0x467d, 0x3425, 0x3535, 0x502c, 0x502d, 0x4e3b, 0x4d3d, 0x4168,
01088     0x502f, 0x3b76, 0x4673, 0x5032, 0x313e, 0x385f, 0x385e, 0x3066,
01089     0x4f4b, 0x4f4a, 0x3a33, 0x3021, 0x5033, 0x5034, 0x5035, 0x4b34,
01090     0x5036, 0x3872, 0x3067, 0x4b72, 0x357c, 0x357d, 0x357e, 0x4462,
01091     0x4e3c, 0x5037, 0x5038, 0x5039, 0x3f4d, 0x3d3a, 0x3f4e, 0x503e,
01092     0x503c, 0x503d, 0x3558, 0x3a23, 0x3270, 0x503b, 0x503a, 0x4a29,
01093     0x3b46, 0x3b45, 0x423e, 0x503f, 0x4955, 0x4067, 0x2138, 0x5040,
01094     0x5042, 0x4265, 0x4e61, 0x304a, 0x5041, 0x323e, 0x3644, 0x4367,
01095     0x376f, 0x5043, 0x4724, 0x346b, 0x5044, 0x304b, 0x3860, 0x346c,
01096     0x497a, 0x4832, 0x3559, 0x3271, 0x5067, 0x4541, 0x476c, 0x5046,
01097     0x483c, 0x4e62, 0x3f2d, 0x3b47, 0x3b77, 0x3240, 0x4451, 0x4322,
01098     0x504a, 0x304c, 0x4463, 0x3d3b, 0x3a34, 0x4d24, 0x424e, 0x323f,
01099     0x5049, 0x4d3e, 0x5045, 0x5047, 0x3a6e, 0x5048, 0x5524, 0x5050,
01100     0x5053, 0x5051, 0x3242, 0x4a3b, 0x504b, 0x504f, 0x3873, 0x3b48,
01101     0x3426, 0x5054, 0x504c, 0x4e63, 0x3b78, 0x504d, 0x5052, 0x5055,
01102     0x504e, 0x3621, 0x304d, 0x3622, 0x3241, 0x5525, 0x4b79, 0x496e,
01103     0x3874, 0x3f2f, 0x4e37, 0x4a58, 0x3738, 0x4225, 0x3264, 0x3d53,
```

```
01104 0x5059, 0x505e, 0x505c, 0x5057, 0x422f, 0x505a, 0x505d, 0x505b,
01105 0x4a5d, 0x5058, 0x3f2e, 0x4b73, 0x505f, 0x5060, 0x3d24, 0x506d,
01106 0x4750, 0x4936, 0x5068, 0x4a70, 0x3236, 0x506c, 0x5066, 0x506f,
01107 0x4152, 0x3844, 0x475c, 0x6047, 0x506e, 0x455d, 0x5063, 0x3876,
01108 0x3875, 0x5061, 0x3c5a, 0x5069, 0x4a6f, 0x434d, 0x5065, 0x3771,
01109 0x5062, 0x506a, 0x5064, 0x4e51, 0x506b, 0x4f41, 0x3666, 0x3770,
01110 0x5070, 0x5071, 0x5075, 0x304e, 0x4a50, 0x5074, 0x5073, 0x5077,
01111 0x5076, 0x4464, 0x3772, 0x5078, 0x3c45, 0x4226, 0x4465, 0x3676,
01112 0x5079, 0x3536, 0x507a, 0x507c, 0x4b35, 0x3766, 0x3b31, 0x4877,
01113 0x507b, 0x3a45, 0x4d43, 0x507e, 0x5123, 0x507d, 0x3a44, 0x3d7d,
01114 0x3739, 0x5124, 0x364f, 0x5121, 0x5122, 0x462f, 0x417c, 0x3623,
01115 0x4b4d, 0x5125, 0x4e3d, 0x5126, 0x5129, 0x5127, 0x414e, 0x5128,
01116 0x512a, 0x512c, 0x512b, 0x4a48, 0x3537, 0x512e, 0x512f, 0x322f,
01117 0x512d, 0x3c74, 0x5132, 0x5131, 0x5130, 0x5056, 0x5133, 0x3d7e,
01118 0x5134, 0x4d25, 0x4c59, 0x5136, 0x5135, 0x5138, 0x5137, 0x5139,
01119 0x513a, 0x3074, 0x3835, 0x373b, 0x3d3c, 0x437b, 0x3624, 0x4068,
01120 0x3877, 0x396e, 0x513c, 0x4c48, 0x4546, 0x3b79, 0x513b, 0x513d,
01121 0x455e, 0x3375, 0x513e, 0x467e, 0x4134, 0x5140, 0x5141, 0x482c,
01122 0x3878, 0x4f3b, 0x5142, 0x3626, 0x4a3c, 0x4236, 0x3671, 0x4535,
01123 0x3773, 0x5143, 0x5144, 0x4662, 0x315f, 0x5147, 0x3a7d, 0x5146,
01124 0x3a46, 0x5148, 0x666e, 0x5149, 0x4b41, 0x514a, 0x514b, 0x514c,
01125 0x3e69, 0x3c4c, 0x3427, 0x514f, 0x514d, 0x4c3d, 0x514e, 0x495a,
01126 0x5150, 0x5151, 0x5152, 0x455f, 0x5156, 0x5154, 0x5155, 0x5153,
01127 0x3a63, 0x5157, 0x4c6a, 0x4e64, 0x5158, 0x4028, 0x5159, 0x3d5a,
01128 0x515a, 0x437c, 0x4e3f, 0x4560, 0x5245, 0x515b, 0x7425, 0x3645,
01129 0x515c, 0x4b5e, 0x3d68, 0x427c, 0x515e, 0x4664, 0x515f, 0x5160,
01130 0x332e, 0x5161, 0x3627, 0x464c, 0x317a, 0x3d50, 0x4821, 0x5162,
01131 0x4561, 0x3f4f, 0x5163, 0x4a2c, 0x405a, 0x3422, 0x3429, 0x5164,
01132 0x5166, 0x373a, 0x5165, 0x4e73, 0x3d69, 0x483d, 0x4a4c, 0x5167,
01133 0x4d78, 0x5168, 0x5169, 0x457e, 0x516a, 0x4029, 0x3a7e, 0x3774,
01134 0x516b, 0x3b49, 0x396f, 0x4466, 0x516d, 0x4227, 0x3a6f, 0x516e,
01135 0x516f, 0x4130, 0x516c, 0x5171, 0x4b36, 0x3964, 0x5170, 0x3775,
01136 0x3a5e, 0x476d, 0x5174, 0x5172, 0x497b, 0x3e6a, 0x517b, 0x3364,
01137 0x5175, 0x5173, 0x414f, 0x5177, 0x5176, 0x3344, 0x3760, 0x517c,
01138 0x4e2d, 0x5178, 0x517d, 0x517a, 0x5179, 0x4e4f, 0x3879, 0x3243,
01139 0x4e74, 0x5243, 0x4558, 0x3965, 0x5222, 0x5223, 0x4e65, 0x4f2b,
01140 0x5225, 0x387a, 0x5224, 0x332f, 0x5226, 0x4b56, 0x443c, 0x4d26,
01141 0x4a59, 0x5227, 0x7055, 0x4630, 0x5228, 0x342a, 0x4c33, 0x3e21,
01142 0x5229, 0x4a67, 0x522d, 0x402a, 0x522a, 0x3650, 0x522b, 0x342b,
01143 0x372e, 0x522e, 0x522f, 0x5230, 0x5231, 0x3c5b, 0x387b, 0x4c5e,
01144 0x4c68, 0x4677, 0x4a71, 0x5232, 0x5233, 0x5235, 0x5237, 0x5236,
01145 0x5238, 0x323d, 0x4b4c, 0x3a7c, 0x5239, 0x4159, 0x3e22, 0x3629,
01146 0x523a, 0x485b, 0x523b, 0x523c, 0x523d, 0x523e, 0x4924, 0x3668,
01147 0x3065, 0x463f, 0x523f, 0x3d3d, 0x4069, 0x5241, 0x5240, 0x3e23,
01148 0x3861, 0x5243, 0x483e, 0x5244, 0x485c, 0x4234, 0x426e, 0x3628,
01149 0x466e, 0x4331, 0x476e, 0x4b4e, 0x5246, 0x406a, 0x3735, 0x5247,
01150 0x5248, 0x312c, 0x3075, 0x346d, 0x4228, 0x3551, 0x4d71, 0x524b,
01151 0x3237, 0x524a, 0x362a, 0x524c, 0x4c71, 0x524d, 0x4e52, 0x387c,
01152 0x3836, 0x524e, 0x5250, 0x524f, 0x3f5f, 0x3139, 0x315e, 0x5251,
01153 0x5252, 0x3837, 0x5253, 0x356e, 0x3b32, 0x5254, 0x4b74, 0x3a35,
01154 0x355a, 0x4d27, 0x4150, 0x483f, 0x3c7d, 0x3d47, 0x3c68, 0x3c75,
01155 0x3d76, 0x4840, 0x5257, 0x3143, 0x4151, 0x387d, 0x3845, 0x3667,
01156 0x525b, 0x4321, 0x427e, 0x362b, 0x3e24, 0x525c, 0x525a, 0x3244,
01157 0x4266, 0x3c38, 0x3b4b, 0x3126, 0x3370, 0x3966, 0x3b4a, 0x525d,
01158 0x525e, 0x3549, 0x3346, 0x3967, 0x3548, 0x445f, 0x3125, 0x4631,
01159 0x4c3e, 0x3921, 0x4d79, 0x4547, 0x387e, 0x372f, 0x5267, 0x3663,
01160 0x4b4a, 0x485d, 0x5266, 0x345e, 0x5261, 0x5262, 0x5264, 0x5265,
01161 0x355b, 0x3f61, 0x4a2d, 0x5263, 0x525f, 0x3863, 0x5260, 0x4f24,
01162 0x4a72, 0x4468, 0x3862, 0x3970, 0x5268, 0x465d, 0x526c, 0x3c7e,
01163 0x3c76, 0x526f, 0x526d, 0x4c23, 0x526a, 0x5273, 0x526e, 0x5271,
01164 0x3846, 0x4c3f, 0x5272, 0x5274, 0x5276, 0x3a70, 0x4f42, 0x526b,
01165 0x5269, 0x5275, 0x5270, 0x5278, 0x5323, 0x527a, 0x527e, 0x5321,
01166 0x527b, 0x533e, 0x3a69, 0x3331, 0x5279, 0x5325, 0x3076, 0x5324,
01167 0x3025, 0x494a, 0x5322, 0x527c, 0x5277, 0x527d, 0x3a48, 0x5326,
01168 0x3077, 0x532f, 0x5327, 0x5328, 0x3e25, 0x4b69, 0x532d, 0x532c,
01169 0x452f, 0x532e, 0x532b, 0x3134, 0x3a36, 0x3f30, 0x5329, 0x4562,
01170 0x532a, 0x3022, 0x5334, 0x4d23, 0x3e27, 0x533a, 0x5339, 0x5330,
01171 0x4243, 0x5331, 0x426f, 0x5336, 0x3e26, 0x5333, 0x4c64, 0x373c,
01172 0x5337, 0x5338, 0x5335, 0x533b, 0x5332, 0x5341, 0x5346, 0x5342,
01173 0x533d, 0x5347, 0x4131, 0x5349, 0x3922, 0x533f, 0x437d, 0x5343,
01174 0x533c, 0x342d, 0x346e, 0x3365, 0x5344, 0x5340, 0x3776, 0x534a,
01175 0x5348, 0x4153, 0x354a, 0x362c, 0x5345, 0x3674, 0x3144, 0x534e,
01176 0x534c, 0x5427, 0x5351, 0x534b, 0x534f, 0x534d, 0x3b4c, 0x5350,
01177 0x5353, 0x5358, 0x5356, 0x5355, 0x4332, 0x3245, 0x5352, 0x5354,
01178 0x3e28, 0x3133, 0x5357, 0x325e, 0x5362, 0x3e7c, 0x535e, 0x535c,
01179 0x535d, 0x535f, 0x313d, 0x4139, 0x5359, 0x535a, 0x337a, 0x5361,
01180 0x346f, 0x5364, 0x5360, 0x5363, 0x4a2e, 0x4655, 0x4838, 0x5366,
01181 0x5365, 0x3345, 0x5367, 0x536a, 0x5369, 0x5368, 0x4739, 0x536b,
01182 0x536c, 0x536e, 0x536d, 0x5370, 0x5373, 0x5371, 0x536f, 0x5372,
01183 0x5374, 0x5375, 0x5376, 0x5377, 0x5378, 0x5145, 0x3c7c, 0x3b4d,
01184 0x3273, 0x3078, 0x4344, 0x5379, 0x3a24, 0x304f, 0x3f5e, 0x537a,
01185 0x3847, 0x3971, 0x537c, 0x537b, 0x4a60, 0x537d, 0x5421, 0x537e,
01186 0x5422, 0x5423, 0x3777, 0x3160, 0x5424, 0x5426, 0x5425, 0x5428,
01187 0x455a, 0x5429, 0x3035, 0x3a5f, 0x373d, 0x434f, 0x542a, 0x542b,
01188 0x542d, 0x542e, 0x3a64, 0x3651, 0x4b37, 0x542c, 0x542f, 0x3a41,
01189 0x3923, 0x5433, 0x3a25, 0x4333, 0x5430, 0x445a, 0x5434, 0x3f62,
01190 0x5432, 0x5435, 0x373f, 0x5436, 0x5437, 0x3924, 0x3340, 0x5439,
```

```
01191 0x543a, 0x543b, 0x5438, 0x5431, 0x543c, 0x543d, 0x4b64, 0x3e6b,
01192 0x543f, 0x5440, 0x543e, 0x5442, 0x4738, 0x3068, 0x4956, 0x5443,
01193 0x3e7d, 0x3c39, 0x475d, 0x3470, 0x3a6b, 0x4b59, 0x4632, 0x3778,
01194 0x424f, 0x5441, 0x5444, 0x4244, 0x5445, 0x5446, 0x5448, 0x4469,
01195 0x342e, 0x7421, 0x3161, 0x4a73, 0x3e6c, 0x4548, 0x3a66, 0x544e,
01196 0x4a3d, 0x4e5d, 0x3274, 0x544a, 0x413a, 0x544d, 0x4563, 0x4549,
01197 0x4564, 0x4839, 0x444d, 0x3a49, 0x5449, 0x3176, 0x4536, 0x544b,
01198 0x5447, 0x3f50, 0x544f, 0x3d4e, 0x362d, 0x5450, 0x4a68, 0x417d,
01199 0x4446, 0x5452, 0x4b4f, 0x5453, 0x5458, 0x4a2f, 0x5457, 0x5451,
01200 0x5454, 0x5456, 0x3a26, 0x4a49, 0x5459, 0x4345, 0x3275, 0x3e6d,
01201 0x545b, 0x545a, 0x3968, 0x545c, 0x545e, 0x545d, 0x5460, 0x5455,
01202 0x5462, 0x5461, 0x545f, 0x3b4e, 0x3f51, 0x4154, 0x5463, 0x403c,
01203 0x306d, 0x4764, 0x445b, 0x5465, 0x5464, 0x5466, 0x5467, 0x5468,
01204 0x5469, 0x4a51, 0x546a, 0x3246, 0x546b, 0x4d3c, 0x3330, 0x5249,
01205 0x3d48, 0x423f, 0x546c, 0x4c6b, 0x4c34, 0x546e, 0x4267, 0x4537,
01206 0x4240, 0x4957, 0x546f, 0x5470, 0x317b, 0x3c3a, 0x5471, 0x3050,
01207 0x5472, 0x5473, 0x3162, 0x3471, 0x4660, 0x4a74, 0x5477, 0x4155,
01208 0x5476, 0x3740, 0x4b5b, 0x5475, 0x4565, 0x5479, 0x5478, 0x547b,
01209 0x547a, 0x317c, 0x547c, 0x3e29, 0x547e, 0x4325, 0x547d, 0x4a33,
01210 0x3d77, 0x455b, 0x5521, 0x3925, 0x5522, 0x4721, 0x485e, 0x4c51,
01211 0x4725, 0x552b, 0x3538, 0x4d45, 0x4c2f, 0x562c, 0x5523, 0x5526,
01212 0x4245, 0x4b38, 0x454a, 0x5527, 0x4b65, 0x3a4a, 0x3e2a, 0x5528,
01213 0x3b50, 0x3b4f, 0x3039, 0x3848, 0x402b, 0x3051, 0x552c, 0x552d,
01214 0x552a, 0x3138, 0x342f, 0x5529, 0x4c45, 0x4931, 0x3028, 0x3079,
01215 0x3b51, 0x3052, 0x3023, 0x5532, 0x5530, 0x4c3c, 0x5533, 0x5531,
01216 0x552f, 0x3f31, 0x552e, 0x4a5a, 0x3864, 0x5537, 0x5538, 0x3e2b,
01217 0x5534, 0x4f2c, 0x474c, 0x5536, 0x3a27, 0x5539, 0x4958, 0x553a,
01218 0x5535, 0x4c3b, 0x475e, 0x553b, 0x4932, 0x553c, 0x5540, 0x553d,
01219 0x3247, 0x553f, 0x3c3b, 0x553e, 0x3779, 0x554c, 0x5545, 0x5542,
01220 0x4364, 0x5541, 0x5543, 0x5544, 0x5546, 0x5547, 0x3472, 0x5549,
01221 0x5548, 0x554a, 0x3e6e, 0x554d, 0x445c, 0x3145, 0x554b, 0x554e,
01222 0x554f, 0x5552, 0x5550, 0x5551, 0x3b52, 0x5553, 0x3926, 0x5554,
01223 0x3b7a, 0x4238, 0x4633, 0x5555, 0x5556, 0x3b5a, 0x3927, 0x4c52, 0x3528,
01224 0x3849, 0x5557, 0x3358, 0x5558, 0x4239, 0x5559, 0x5623, 0x555a,
01225 0x555b, 0x555c, 0x555e, 0x555f, 0x5560, 0x4270, 0x3127, 0x3c69,
01226 0x3042, 0x4157, 0x3430, 0x3c35, 0x3928, 0x4566, 0x3d21, 0x3431,
01227 0x4368, 0x446a, 0x3038, 0x3539, 0x4a75, 0x3c42, 0x3552, 0x406b,
01228 0x3c3c, 0x4d28, 0x5561, 0x355c, 0x3a4b, 0x3332, 0x3163, 0x3e2c,
01229 0x3248, 0x5562, 0x4d46, 0x3d49, 0x3c64, 0x5563, 0x3473, 0x4652,
01230 0x4c29, 0x5564, 0x5565, 0x4959, 0x5567, 0x3428, 0x3677, 0x5566,
01231 0x3432, 0x3f32, 0x556b, 0x3b21, 0x3249, 0x556a, 0x5568, 0x556c,
01232 0x5569, 0x472b, 0x5c4d, 0x3f33, 0x556d, 0x4e40, 0x556e, 0x5570,
01233 0x437e, 0x556f, 0x4023, 0x3b7b, 0x4250, 0x3c77, 0x4975, 0x406c,
01234 0x3c4d, 0x5571, 0x3e2d, 0x5572, 0x5573, 0x3053, 0x423a, 0x3f52,
01235 0x5574, 0x4633, 0x3e2e, 0x3e2f, 0x5575, 0x406d, 0x3e30, 0x5576,
01236 0x5577, 0x4c60, 0x5578, 0x3646, 0x3d22, 0x5579, 0x557a, 0x3c5c,
01237 0x3f2c, 0x4674, 0x3f54, 0x4878, 0x4722, 0x3649, 0x557b, 0x356f,
01238 0x557c, 0x367e, 0x464f, 0x3230, 0x3b53, 0x557d, 0x5622, 0x5621,
01239 0x367d, 0x557e, 0x4538, 0x4230, 0x454b, 0x3c48, 0x4158, 0x4d7a,
01240 0x5624, 0x5625, 0x4656, 0x3b33, 0x5627, 0x5628, 0x5629, 0x3474,
01241 0x562a, 0x562b, 0x562c, 0x413b, 0x3464, 0x562d, 0x4c28, 0x4252,
01242 0x3359, 0x562f, 0x5631, 0x345f, 0x562e, 0x5630, 0x5633, 0x5632,
01243 0x5634, 0x5635, 0x463d, 0x362e, 0x3265, 0x5636, 0x563b, 0x5639,
01244 0x4a77, 0x4a76, 0x4567, 0x5638, 0x3d54, 0x5637, 0x3f72, 0x563c,
01245 0x3a6a, 0x5642, 0x5643, 0x563d, 0x3333, 0x563e, 0x5647, 0x5646,
01246 0x5645, 0x5641, 0x5640, 0x5644, 0x4a78, 0x564b, 0x5648, 0x564a,
01247 0x4d72, 0x5649, 0x563f, 0x3f73, 0x564c, 0x3a37, 0x564d, 0x564e,
01248 0x5651, 0x5650, 0x564f, 0x4568, 0x563a, 0x5657, 0x5653, 0x5652,
01249 0x5654, 0x5655, 0x5658, 0x4e66, 0x5659, 0x5656, 0x565a, 0x3460,
01250 0x565b, 0x565d, 0x565c, 0x565e, 0x565f, 0x406e, 0x3d23, 0x3d64,
01251 0x4163, 0x3929, 0x3a38, 0x392a, 0x3570, 0x5660, 0x3a39, 0x384a,
01252 0x5661, 0x4c26, 0x4743, 0x5662, 0x392b, 0x342c, 0x4327, 0x3652,
01253 0x3b54, 0x495b, 0x4841, 0x5663, 0x3475, 0x5666, 0x4421, 0x5665,
01254 0x5664, 0x5667, 0x446b, 0x3f63, 0x3b55, 0x404a, 0x4253, 0x3522,
01255 0x4422, 0x5668, 0x5669, 0x3e6f, 0x4b39, 0x566c, 0x566b, 0x566a,
01256 0x497d, 0x5673, 0x4b5a, 0x566d, 0x566f, 0x4b6b, 0x566e, 0x5670,
01257 0x4828, 0x5671, 0x4a3e, 0x5672, 0x3433, 0x4a3f, 0x472f, 0x5674,
01258 0x5675, 0x392c, 0x3434, 0x5676, 0x3838, 0x4d44, 0x4d29, 0x3476,
01259 0x5678, 0x4423, 0x392d, 0x3e31, 0x485f, 0x3e32, 0x3d78, 0x446c,
01260 0x4a79, 0x4539, 0x392e, 0x495c, 0x5679, 0x4559, 0x3a42, 0x384b,
01261 0x446d, 0x3043, 0x3d6e, 0x392f, 0x4d47, 0x567a, 0x567b, 0x4751,
01262 0x567c, 0x4e77, 0x4f2d, 0x567e, 0x567d, 0x3347, 0x5721, 0x5724,
01263 0x5725, 0x5723, 0x4940, 0x3e33, 0x5727, 0x5726, 0x5722, 0x5728,
01264 0x5729, 0x572a, 0x572d, 0x572b, 0x572c, 0x572e, 0x3164, 0x446e,
01265 0x572f, 0x377a, 0x3276, 0x4736, 0x5730, 0x467b, 0x4a5b, 0x5731,
01266 0x4f2e, 0x5732, 0x4a40, 0x5735, 0x5021, 0x5031, 0x3c30, 0x4675,
01267 0x5736, 0x355d, 0x4424, 0x307a, 0x5737, 0x4a26, 0x3930, 0x4350,
01268 0x446f, 0x4c6f, 0x3839, 0x384c, 0x5738, 0x5739, 0x573f, 0x3c65,
01269 0x4425, 0x362f, 0x573a, 0x492b, 0x4346, 0x573b, 0x573c, 0x3630,
01270 0x573d, 0x573e, 0x5740, 0x4576, 0x5741, 0x5742, 0x5743, 0x5734,
01271 0x5733, 0x5744, 0x3741, 0x4927, 0x3a4c, 0x4937, 0x4426, 0x494b,
01272 0x5745, 0x3e34, 0x3146, 0x5746, 0x5747, 0x4c72, 0x4860, 0x574a,
01273 0x317d, 0x402c, 0x5749, 0x5748, 0x3742, 0x4254, 0x574e, 0x574c,
01274 0x574b, 0x4e27, 0x3865, 0x3d79, 0x574d, 0x454c, 0x3d3e, 0x4640,
01275 0x5751, 0x5750, 0x574f, 0x5752, 0x3866, 0x5753, 0x497c, 0x3d5b,
01276 0x5754, 0x4879, 0x4641, 0x4427, 0x4530, 0x5755, 0x352b, 0x3f34,
01277 0x492c, 0x3477, 0x4726, 0x5756, 0x3b56, 0x4b3a, 0x4b3b, 0x317e,
```

```
01278 0x575b, 0x4369, 0x5758, 0x3277, 0x582d, 0x575a, 0x4730, 0x5759,
01279 0x5757, 0x397a, 0x575d, 0x5763, 0x5769, 0x5761, 0x455c, 0x5766,
01280 0x495d, 0x5760, 0x5765, 0x4e67, 0x3b57, 0x4255, 0x575e, 0x355e,
01281 0x5768, 0x402d, 0x3165, 0x5762, 0x3278, 0x5767, 0x3631, 0x5764,
01282 0x576a, 0x576c, 0x5776, 0x5774, 0x5771, 0x5770, 0x4e78, 0x5772,
01283 0x3632, 0x3931, 0x3d7a, 0x5779, 0x576b, 0x576f, 0x575f, 0x327a,
01284 0x5773, 0x5775, 0x4351, 0x3a28, 0x3238, 0x576d, 0x5778, 0x5777,
01285 0x3633, 0x4229, 0x3366, 0x3743, 0x576e, 0x577a, 0x577d, 0x5821,
01286 0x3c3d, 0x5827, 0x4470, 0x577b, 0x5825, 0x3279, 0x5823, 0x5824,
01287 0x577e, 0x5822, 0x3867, 0x4d2a, 0x3435, 0x3159, 0x5826, 0x473a,
01288 0x302d, 0x4861, 0x575c, 0x582c, 0x5830, 0x4c65, 0x5829, 0x4569,
01289 0x582e, 0x3e70, 0x582f, 0x4657, 0x4f47, 0x582b, 0x5831, 0x397b,
01290 0x404b, 0x3054, 0x582a, 0x5828, 0x415a, 0x577c, 0x3b34, 0x4246,
01291 0x583d, 0x415b, 0x5838, 0x5835, 0x5836, 0x3c66, 0x5839, 0x583c,
01292 0x5837, 0x3d25, 0x583a, 0x5834, 0x4c7c, 0x4c7b, 0x583e, 0x583f,
01293 0x3055, 0x5833, 0x3672, 0x3026, 0x3436, 0x583b, 0x5843, 0x5842,
01294 0x5847, 0x5848, 0x5846, 0x5849, 0x5841, 0x5845, 0x584a, 0x584b,
01295 0x5840, 0x3b7c, 0x5844, 0x4256, 0x3932, 0x5832, 0x3f35, 0x5858,
01296 0x4a69, 0x584e, 0x584f, 0x5850, 0x5857, 0x5856, 0x4b7d, 0x3437,
01297 0x5854, 0x3745, 0x3334, 0x5851, 0x4e38, 0x5853, 0x3056, 0x5855,
01298 0x584c, 0x5852, 0x5859, 0x3744, 0x584d, 0x4d5d, 0x4d2b, 0x585c,
01299 0x5860, 0x417e, 0x4e79, 0x5861, 0x585e, 0x585b, 0x585a, 0x585f,
01300 0x4a30, 0x4634, 0x3746, 0x5862, 0x585d, 0x5863, 0x377b, 0x3231,
01301 0x586b, 0x3438, 0x5869, 0x586a, 0x3a29, 0x5868, 0x5866, 0x5865,
01302 0x586c, 0x5864, 0x586e, 0x327b, 0x5870, 0x586f, 0x4428, 0x5873,
01303 0x5871, 0x5867, 0x377c, 0x5872, 0x5876, 0x5875, 0x5877, 0x5874,
01304 0x5878, 0x5879, 0x587a, 0x4a6a, 0x587c, 0x587b, 0x3d3f, 0x402e,
01305 0x3266, 0x327c, 0x587d, 0x303f, 0x404c, 0x587e, 0x6c43, 0x5921,
01306 0x3761, 0x5922, 0x406f, 0x5923, 0x5924, 0x353a, 0x5925, 0x5926,
01307 0x5927, 0x4257, 0x384d, 0x4c61, 0x4b3c, 0x3d6a, 0x5928, 0x4070,
01308 0x6e3d, 0x4862, 0x3c6a, 0x3a4d, 0x5929, 0x4247, 0x4a27, 0x4271,
01309 0x592c, 0x592a, 0x592d, 0x592b, 0x592e, 0x4a31, 0x3037, 0x495e,
01310 0x4863, 0x592f, 0x5932, 0x3e35, 0x353b, 0x5930, 0x5937, 0x3e36,
01311 0x5931, 0x4744, 0x4d5e, 0x5933, 0x5934, 0x5938, 0x456a, 0x5935,
01312 0x3933, 0x405e, 0x5946, 0x4834, 0x4272, 0x4864, 0x5a2d, 0x4a7a,
01313 0x4471, 0x4b75, 0x593b, 0x3221, 0x436a, 0x5944, 0x4334, 0x593e,
01314 0x5945, 0x5940, 0x5947, 0x5943, 0x5942, 0x476f, 0x593c, 0x327d,
01315 0x593a, 0x3571, 0x4273, 0x5936, 0x5939, 0x3934, 0x405b, 0x3e37,
01316 0x5941, 0x4752, 0x3572, 0x3348, 0x3367, 0x3f21, 0x5949, 0x594e,
01317 0x594a, 0x377d, 0x594f, 0x3b22, 0x3969, 0x3d26, 0x593d, 0x3b7d,
01318 0x594c, 0x3b58, 0x594d, 0x3044, 0x5948, 0x4429, 0x3573, 0x3634,
01319 0x594b, 0x3027, 0x3a43, 0x3f36, 0x4472, 0x4854, 0x5951, 0x415e,
01320 0x422a, 0x3b2b, 0x5952, 0x5954, 0x5950, 0x4a61, 0x443d, 0x415c,
01321 0x4a7b, 0x3c4e, 0x5960, 0x595f, 0x3f78, 0x377e, 0x5959, 0x3e39,
01322 0x4668, 0x4731, 0x5957, 0x415d, 0x3c78, 0x595c, 0x3e38, 0x5956,
01323 0x595b, 0x4753, 0x5955, 0x3721, 0x335d, 0x595d, 0x4e2b, 0x3a4e,
01324 0x4335, 0x595a, 0x405c, 0x3935, 0x3f64, 0x3166, 0x413c, 0x5958,
01325 0x3545, 0x3747, 0x444f, 0x595e, 0x415f, 0x5961, 0x5963, 0x4237,
01326 0x5969, 0x5964, 0x5966, 0x4941, 0x4473, 0x5967, 0x4d2c, 0x4d48,
01327 0x3439, 0x302e, 0x5965, 0x5962, 0x3478, 0x3167, 0x5968, 0x4d49,
01328 0x596c, 0x423b, 0x5973, 0x596d, 0x596a, 0x5971, 0x5953, 0x596e,
01329 0x5972, 0x4842, 0x456b, 0x596b, 0x596f, 0x3748, 0x3a71, 0x405d,
01330 0x5977, 0x4526, 0x5974, 0x4b60, 0x5975, 0x5976, 0x4c4e, 0x4022,
01331 0x3762, 0x597d, 0x3b35, 0x597a, 0x5979, 0x4732, 0x4635, 0x4531,
01332 0x597b, 0x597c, 0x496f, 0x4745, 0x3b23, 0x4071, 0x4b50, 0x3349,
01333 0x5a25, 0x597e, 0x4d4a, 0x5a27, 0x5a23, 0x5a24, 0x4160, 0x5a22,
01334 0x593f, 0x5a26, 0x5a21, 0x5a2b, 0x5a2c, 0x4527, 0x5a2e, 0x3b24,
01335 0x5a29, 0x353c, 0x5a2f, 0x5a28, 0x5a33, 0x5a32, 0x5a31, 0x5a34,
01336 0x5a36, 0x3e71, 0x5a35, 0x5a39, 0x5a37, 0x5a38, 0x5970, 0x5a3b,
01337 0x5a3a, 0x5978, 0x5a3c, 0x5a30, 0x3b59, 0x5a3d, 0x5a3e, 0x5a40,
01338 0x5a3f, 0x5a41, 0x327e, 0x3936, 0x4a7c, 0x402f, 0x384e, 0x5a43,
01339 0x5a46, 0x4952, 0x355f, 0x5a45, 0x5a44, 0x4754, 0x5a47, 0x3635,
01340 0x5a49, 0x5a48, 0x343a, 0x3b36, 0x4658, 0x3749, 0x3f74, 0x5a4a,
01341 0x4030, 0x4528, 0x495f, 0x5a4b, 0x5a4c, 0x5a4d, 0x4a38, 0x555d,
01342 0x4046, 0x494c, 0x3a58, 0x4865, 0x4843, 0x454d, 0x4e41, 0x5a4f,
01343 0x3c50, 0x5a50, 0x3036, 0x3654, 0x404d, 0x4960, 0x5a51, 0x3b42,
01344 0x4347, 0x3b5b, 0x3f37, 0x5a52, 0x4a7d, 0x3177, 0x3b5c, 0x5a55,
01345 0x5a53, 0x5a56, 0x4e39, 0x5a54, 0x407b, 0x5a57, 0x4232, 0x5a58,
01346 0x347a, 0x5a5a, 0x5a59, 0x5a5b, 0x5a5c, 0x347b, 0x467c, 0x4336,
01347 0x356c, 0x3b5d, 0x4161, 0x3d5c, 0x3030, 0x5a5d, 0x3222, 0x5a61,
01348 0x3937, 0x5a60, 0x3a2b, 0x3e3a, 0x5a5f, 0x3e3b, 0x4c40, 0x3a2a,
01349 0x3057, 0x404e, 0x5a66, 0x4031, 0x3147, 0x3d55, 0x4b66, 0x3a72,
01350 0x3e3c, 0x4027, 0x5a65, 0x5a63, 0x5a64, 0x436b, 0x5b26, 0x5a6a,
01351 0x3b7e, 0x3938, 0x5a68, 0x5a69, 0x3f38, 0x5a67, 0x3b2f, 0x5a6c,
01352 0x5a6b, 0x5a70, 0x5a71, 0x5a6d, 0x3322, 0x5a6e, 0x5a6f, 0x4855,
01353 0x4961, 0x374a, 0x5a72, 0x4032, 0x3e3d, 0x4352, 0x3647, 0x5a73,
01354 0x5a77, 0x324b, 0x5a74, 0x5a76, 0x5a75, 0x3d6b, 0x4348, 0x3045,
01355 0x5a78, 0x5a79, 0x442a, 0x4e71, 0x3b43, 0x4a6b, 0x4b3d, 0x5b22,
01356 0x5a7b, 0x5a7e, 0x5a7d, 0x5a7a, 0x5b21, 0x465e, 0x5a7c, 0x5b23,
01357 0x3d6c, 0x5b24, 0x4d4b, 0x4778, 0x5b25, 0x5b27, 0x5b28, 0x5b29,
01358 0x364a, 0x3148, 0x3939, 0x5b2a, 0x5b2b, 0x3d71, 0x4162, 0x5258,
01359 0x413e, 0x413d, 0x4258, 0x3a47, 0x5072, 0x376e, 0x4d2d, 0x4a7e,
01360 0x497e, 0x5b2c, 0x3a73, 0x443f, 0x5b2d, 0x4f2f, 0x4b3e, 0x442b,
01361 0x5b2e, 0x347c, 0x5b2f, 0x5b30, 0x4c5a, 0x4c24, 0x4b76, 0x4b5c,
01362 0x3b25, 0x5b32, 0x3c6b, 0x4b51, 0x5b34, 0x5b37, 0x5b36, 0x3479,
01363 0x3560, 0x5b33, 0x5b35, 0x5b38, 0x3f79, 0x4d7b, 0x3049, 0x3a60,
01364 0x423c, 0x3c5d, 0x3e73, 0x5b3b, 0x454e, 0x5b39, 0x422b, 0x5b3a,
```



```
01365 0x3e72, 0x4c5d, 0x5b3c, 0x5b3d, 0x4d68, 0x5b42, 0x393a, 0x4755,
01366 0x5b3f, 0x456c, 0x5a5e, 0x5a62, 0x354f, 0x4747, 0x5b41, 0x3e3e,
01367 0x4844, 0x5b47, 0x487a, 0x5b3e, 0x5b44, 0x5b43, 0x404f, 0x4b6d,
01368 0x4e53, 0x4b67, 0x324c, 0x3b5e, 0x4f48, 0x5b46, 0x3f75, 0x5b45,
01369 0x5b40, 0x384f, 0x5b4c, 0x5b4a, 0x324d, 0x5b48, 0x5b4e, 0x5b54,
01370 0x4248, 0x4a41, 0x5b56, 0x4922, 0x5b55, 0x4770, 0x4b3f, 0x343b,
01371 0x4077, 0x3d40, 0x4453, 0x4d2e, 0x5b51, 0x5b50, 0x5b52, 0x5b4f,
01372 0x5b57, 0x5b4d, 0x5b4b, 0x5b53, 0x5b49, 0x436c, 0x4c78, 0x3c46,
01373 0x3a74, 0x3a3a, 0x4b6f, 0x3341, 0x444e, 0x464a, 0x3149, 0x4072,
01374 0x4034, 0x372a, 0x5b59, 0x393b, 0x337c, 0x5b5b, 0x3374, 0x5b61,
01375 0x5b5e, 0x4073, 0x334b, 0x3a2c, 0x334a, 0x3a4f, 0x5b5c, 0x3765,
01376 0x374b, 0x456d, 0x5b5a, 0x3046, 0x5b5d, 0x5b5f, 0x364d, 0x372c,
01377 0x343c, 0x354b, 0x5b62, 0x3a79, 0x4b71, 0x3b37, 0x5b63, 0x4930,
01378 0x5b6f, 0x3233, 0x5b64, 0x5b75, 0x5b65, 0x4e42, 0x5b6c, 0x475f,
01379 0x5b74, 0x5b67, 0x3034, 0x5b69, 0x393c, 0x5b6b, 0x5b6a, 0x5b66,
01380 0x5b71, 0x3e3f, 0x546d, 0x3868, 0x4d7c, 0x5b68, 0x4474, 0x3323,
01381 0x3a2d, 0x5b60, 0x5b70, 0x3361, 0x5b6e, 0x5b72, 0x456e, 0x347e,
01382 0x5c32, 0x4c49, 0x5b77, 0x347d, 0x5b7e, 0x4b40, 0x5c21, 0x5c23,
01383 0x5c27, 0x5b79, 0x432a, 0x456f, 0x5c2b, 0x5b7c, 0x5c28, 0x5c22,
01384 0x3f39, 0x5c2c, 0x4033, 0x5c2a, 0x343d, 0x4f50, 0x5b76, 0x5c26,
01385 0x3058, 0x5b78, 0x4c3a, 0x5b7d, 0x3f22, 0x4447, 0x5b73, 0x5c25,
01386 0x3f7a, 0x5c2f, 0x3371, 0x3821, 0x5c31, 0x5b7a, 0x5c30, 0x5c29,
01387 0x5b7b, 0x5c2d, 0x5c2e, 0x5c3f, 0x464e, 0x5c24, 0x5c3b, 0x5c3d,
01388 0x4458, 0x4d4c, 0x4976, 0x5c38, 0x424a, 0x5c3e, 0x413f, 0x5c35,
01389 0x5c42, 0x5c41, 0x466f, 0x5c40, 0x466a, 0x5c44, 0x5c37, 0x3648,
01390 0x5c3a, 0x3d5d, 0x4760, 0x5c3c, 0x364b, 0x5c34, 0x5c36, 0x5c33,
01391 0x4f30, 0x335a, 0x5c39, 0x5c43, 0x3335, 0x3a67, 0x315d, 0x5c54,
01392 0x4f31, 0x5c57, 0x3f3a, 0x5c56, 0x5c55, 0x5c52, 0x5c46, 0x5c63,
01393 0x5c45, 0x5c58, 0x5c50, 0x5c4b, 0x5c48, 0x5c49, 0x5c51, 0x7422,
01394 0x5c4e, 0x393d, 0x4448, 0x4164, 0x5c4c, 0x5c47, 0x5c4a, 0x4d4d,
01395 0x4b6a, 0x5c4f, 0x5c59, 0x5c61, 0x5c5a, 0x5c67, 0x5c65, 0x5c60,
01396 0x5c5f, 0x4450, 0x4165, 0x5c5d, 0x5c5b, 0x5c62, 0x5c68, 0x4875,
01397 0x5c6e, 0x5c69, 0x5c6c, 0x5c66, 0x4374, 0x4938, 0x5c5c, 0x5c64,
01398 0x3e40, 0x4c4f, 0x5c78, 0x5c6b, 0x3822, 0x3223, 0x335f, 0x5c53,
01399 0x3e41, 0x5c70, 0x5c77, 0x3c79, 0x3372, 0x432e, 0x5c6d, 0x5c72,
01400 0x5c76, 0x3636, 0x354c, 0x5c74, 0x3521, 0x464b, 0x5c73, 0x5c75,
01401 0x5c6f, 0x5c71, 0x3360, 0x4349, 0x5c7c, 0x5c7a, 0x3869, 0x5c79,
01402 0x5d21, 0x5b58, 0x5c7b, 0x5c7d, 0x5c7e, 0x5d2c, 0x5d28, 0x5b6d,
01403 0x5d27, 0x5d26, 0x5d23, 0x5c6a, 0x5d25, 0x5d24, 0x5d2a, 0x4f26,
01404 0x5d2d, 0x367b, 0x5d29, 0x5d2b, 0x4827, 0x5d2e, 0x5d32, 0x5d2f,
01405 0x4d73, 0x5d30, 0x5c5e, 0x5d33, 0x5d34, 0x3135, 0x5d36, 0x3767,
01406 0x3c21, 0x3655, 0x3224, 0x4d5f, 0x5d38, 0x5d37, 0x5d3a, 0x353d,
01407 0x3656, 0x343e, 0x5d3d, 0x5d3c, 0x5d3e, 0x324e, 0x4337, 0x5d3f,
01408 0x343f, 0x5d41, 0x5d40, 0x5d42, 0x5d43, 0x5d44, 0x3b5f, 0x4035,
01409 0x3a21, 0x4970, 0x4a62, 0x4f44, 0x3b75, 0x3a50, 0x4e72, 0x5d45,
01410 0x5d46, 0x3b60, 0x5d47, 0x5d48, 0x5d4a, 0x5d49, 0x4b58, 0x3d5e,
01411 0x3c6c, 0x3b44, 0x5d4b, 0x5d4d, 0x3f23, 0x5d4c, 0x5d4e, 0x5d4f,
01412 0x5d50, 0x5d51, 0x5d52, 0x5d54, 0x5d53, 0x5d55, 0x3225, 0x434a,
01413 0x5d56, 0x3b26, 0x334c, 0x5d57, 0x4542, 0x544c, 0x3523, 0x5d58,
01414 0x5d59, 0x4a6c, 0x4b68, 0x4647, 0x5d5a, 0x4866, 0x487b, 0x4c53,
01415 0x5d5b, 0x5d5d, 0x5d5c, 0x5d5f, 0x5d5e, 0x5d61, 0x3b61, 0x4c31,
01416 0x5d62, 0x5d63, 0x3524, 0x5d64, 0x5d66, 0x5d65, 0x3f65, 0x4939,
01417 0x314a, 0x4845, 0x4475, 0x3d41, 0x3561, 0x4846, 0x3c2e, 0x5d68,
01418 0x3440, 0x3178, 0x4672, 0x5d67, 0x393e, 0x4353, 0x5d69, 0x5d71,
01419 0x5d6a, 0x4241, 0x3562, 0x5d72, 0x3768, 0x3525, 0x5d70, 0x5d6e,
01420 0x5d6b, 0x4d60, 0x4440, 0x4659, 0x5d6c, 0x5d74, 0x5d73, 0x3723,
01421 0x322d, 0x3a3b, 0x5d6d, 0x5d6f, 0x4b57, 0x4274, 0x4b77, 0x5d7c,
01422 0x5d7d, 0x324f, 0x4a28, 0x4c7d, 0x5e21, 0x3c23, 0x3e42, 0x5d78,
01423 0x5d7e, 0x3168, 0x3637, 0x5d75, 0x5d7a, 0x4074, 0x4771, 0x4867,
01424 0x5d77, 0x4b21, 0x5d79, 0x5e24, 0x5e22, 0x5d7b, 0x4b22, 0x4748,
01425 0x3563, 0x4525, 0x436d, 0x5e25, 0x5e23, 0x4259, 0x5d76, 0x314b,
01426 0x4d4e, 0x5e30, 0x5e2f, 0x4076, 0x5e2c, 0x4d6c, 0x4636, 0x5e26,
01427 0x4445, 0x314c, 0x393f, 0x5e29, 0x3d27, 0x5e2e, 0x5e2d, 0x5e28,
01428 0x5e2b, 0x3368, 0x5e2a, 0x4749, 0x4e2e, 0x3e74, 0x4075, 0x5e36,
01429 0x5e34, 0x494d, 0x5e31, 0x5e33, 0x313a, 0x3940, 0x4f32, 0x333d,
01430 0x4962, 0x4d61, 0x3324, 0x3f3b, 0x5e35, 0x5e3a, 0x3e43, 0x4d30,
01431 0x5e37, 0x5e32, 0x5e38, 0x4e5e, 0x4573, 0x4642, 0x3336, 0x3155,
01432 0x5e3e, 0x5e41, 0x4e43, 0x4d64, 0x5e48, 0x5e42, 0x5e3f, 0x4e54,
01433 0x5e45, 0x3d4a, 0x5e47, 0x5e4c, 0x4571, 0x5e4a, 0x5e44, 0x4338,
01434 0x5e4b, 0x5e40, 0x5e46, 0x5e4d, 0x307c, 0x5e43, 0x5e4e, 0x3f3c,
01435 0x3d5f, 0x4a25, 0x3a2e, 0x5e3b, 0x5e49, 0x453a, 0x4036, 0x3369,
01436 0x3a51, 0x3e44, 0x5e3d, 0x3d42, 0x374c, 0x5e3c, 0x5e52, 0x3d6d,
01437 0x383a, 0x5e61, 0x5e5b, 0x3574, 0x454f, 0x5e56, 0x5e5f, 0x302f,
01438 0x3132, 0x3239, 0x5e58, 0x422c, 0x5e4f, 0x5e51, 0x3941, 0x5e62,
01439 0x5e5d, 0x5e55, 0x5e5c, 0x4c2b, 0x5e5a, 0x5e5e, 0x3850, 0x3e45,
01440 0x4339, 0x5e54, 0x4d2f, 0x5e57, 0x5e50, 0x4572, 0x5e53, 0x5e59,
01441 0x4f51, 0x3c3e, 0x4b7e, 0x5e63, 0x482e, 0x5e6f, 0x383b, 0x3d60,
01442 0x5e65, 0x4e2f, 0x3942, 0x5e72, 0x306e, 0x5e70, 0x5e64, 0x5e6a,
01443 0x5e6c, 0x4d4f, 0x5e67, 0x452e, 0x5e69, 0x5e71, 0x5e6b, 0x4c47,
01444 0x5e66, 0x3c22, 0x5e7e, 0x336a, 0x5e68, 0x5e6d, 0x5e6e, 0x426c,
01445 0x425a, 0x5e76, 0x5e7c, 0x5e7a, 0x4529, 0x5f23, 0x5e77, 0x5e78,
01446 0x5e60, 0x3579, 0x493a, 0x3c3f, 0x3977, 0x4f33, 0x5e74, 0x5f22,
01447 0x3169, 0x4166, 0x4779, 0x3441, 0x4e7a, 0x4c21, 0x4452, 0x5e7b,
01448 0x5e7d, 0x4132, 0x5f21, 0x5e79, 0x5e73, 0x3443, 0x3769, 0x5f2f,
01449 0x5f2a, 0x4078, 0x3363, 0x3d61, 0x5f33, 0x5f2c, 0x442c, 0x5f29,
01450 0x4459, 0x5f4c, 0x5f26, 0x5f25, 0x5f2e, 0x5f28, 0x5f27, 0x5f2d,
01451 0x4021, 0x5f24, 0x5f30, 0x5f31, 0x3442, 0x5f36, 0x5f35, 0x5f37,
```

01452 0x5f3a, 0x4543, 0x5f34, 0x5f38, 0x3763, 0x4279, 0x5f32, 0x473b,
01453 0x5f39, 0x5f3e, 0x5f3c, 0x5f3f, 0x5f42, 0x5f3b, 0x396a, 0x4728,
01454 0x5e39, 0x4d74, 0x5f3d, 0x5f41, 0x4275, 0x5f40, 0x5f2b, 0x6f69,
01455 0x5f45, 0x5f49, 0x5f47, 0x5f43, 0x5f44, 0x5f48, 0x5f46, 0x494e,
01456 0x5f4e, 0x5f4b, 0x5f4a, 0x5f4d, 0x4654, 0x5f4f, 0x4375, 0x426d,
01457 0x4025, 0x5f50, 0x5f52, 0x5f51, 0x5e75, 0x5f53, 0x4667, 0x5f54,
01458 0x3250, 0x4574, 0x3325, 0x3564, 0x3c5e, 0x3a52, 0x4f27, 0x3f66,
01459 0x316a, 0x5f56, 0x5f55, 0x5f59, 0x433a, 0x5f5c, 0x5f57, 0x5f5b,
01460 0x5f5a, 0x4540, 0x3059, 0x4e75, 0x5f5e, 0x3128, 0x5f60, 0x5f5f,
01461 0x5f5d, 0x5f58, 0x4b23, 0x5f62, 0x5f61, 0x316b, 0x5f64, 0x4a32,
01462 0x5f63, 0x4c35, 0x3e47, 0x4133, 0x3e46, 0x4e7b, 0x5f6a, 0x4079,
01463 0x5f66, 0x5f6b, 0x316c, 0x5f69, 0x4761, 0x5f65, 0x5f68, 0x3e48,
01464 0x4851, 0x5f6c, 0x3c51, 0x407a, 0x5f6f, 0x5f67, 0x3727, 0x5f6d,
01465 0x4d50, 0x5f70, 0x7426, 0x3d4f, 0x5f71, 0x5f72, 0x472e, 0x5f74,
01466 0x5f75, 0x4733, 0x4735, 0x4575, 0x5f77, 0x5f79, 0x4e55, 0x5f76,
01467 0x316d, 0x5f73, 0x535b, 0x5f7a, 0x4167, 0x3b38, 0x5f7c, 0x5f7b,
01468 0x3f24, 0x5259, 0x5f7d, 0x6021, 0x5f6e, 0x5f7e, 0x6022, 0x477a,
01469 0x6023, 0x6024, 0x6025, 0x6026, 0x445e, 0x6028, 0x6027, 0x6029,
01470 0x602a, 0x3c5f, 0x4963, 0x4c6c, 0x602b, 0x602c, 0x4156, 0x3c24,
01471 0x602d, 0x602e, 0x602f, 0x4a52, 0x4847, 0x6030, 0x4757, 0x442d,
01472 0x6031, 0x3267, 0x356d, 0x4c46, 0x4c36, 0x3234, 0x4f34, 0x4b52,
01473 0x4a2a, 0x4037, 0x6032, 0x4643, 0x3823, 0x6033, 0x3a54, 0x6035,
01474 0x6034, 0x6036, 0x6037, 0x6038, 0x353e, 0x6039, 0x603a, 0x3824,
01475 0x4848, 0x603c, 0x3e75, 0x603b, 0x3638, 0x603d, 0x603f, 0x603e,
01476 0x6040, 0x3851, 0x6041, 0x3669, 0x4140, 0x397d, 0x6043, 0x6044,
01477 0x6042, 0x3c6d, 0x4648, 0x3639, 0x6046, 0x432c, 0x6045, 0x4f35,
01478 0x4762, 0x6049, 0x604b, 0x6048, 0x4c54, 0x604a, 0x604c, 0x4e44,
01479 0x6050, 0x604f, 0x4376, 0x472d, 0x3825, 0x604e, 0x604d, 0x4d31,
01480 0x4d32, 0x6051, 0x316e, 0x3976, 0x3b62, 0x6052, 0x6053, 0x6055,
01481 0x3d43, 0x6057, 0x6056, 0x6058, 0x334d, 0x605a, 0x6059, 0x605c,
01482 0x605b, 0x383c, 0x4e28, 0x364c, 0x3226, 0x366a, 0x3461, 0x4e68,
01483 0x605e, 0x6060, 0x6061, 0x3251, 0x605d, 0x3b39, 0x4441, 0x605f,
01484 0x6064, 0x3c6e, 0x6062, 0x373e, 0x4849, 0x6063, 0x607e, 0x6069,
01485 0x383d, 0x3565, 0x6066, 0x4d7d, 0x4e30, 0x4276, 0x6068, 0x606a,
01486 0x4e56, 0x3657, 0x487c, 0x474a, 0x606b, 0x606d, 0x6070, 0x606c,
01487 0x606f, 0x386a, 0x314d, 0x6071, 0x3f70, 0x606e, 0x4e5c, 0x6074,
01488 0x7424, 0x6072, 0x6075, 0x6067, 0x6073, 0x3a3c, 0x6076, 0x6077,
01489 0x4d7e, 0x6078, 0x6079, 0x6065, 0x607a, 0x3444, 0x3c25, 0x607b,
01490 0x607c, 0x607d, 0x313b, 0x6121, 0x493b, 0x6122, 0x3424, 0x6123,
01491 0x6124, 0x6125, 0x6127, 0x6128, 0x6126, 0x4953, 0x612a, 0x6129,
01492 0x612c, 0x612b, 0x612d, 0x612e, 0x6130, 0x612f, 0x3979, 0x6132,
01493 0x6131, 0x3445, 0x3f53, 0x453c, 0x6133, 0x4038, 0x3b3a, 0x3179,
01494 0x6134, 0x4d51, 0x4a63, 0x6135, 0x4544, 0x4d33, 0x3943, 0x3f3d,
01495 0x434b, 0x5234, 0x442e, 0x3268, 0x6136, 0x6137, 0x613c, 0x613a,
01496 0x6139, 0x5a42, 0x3326, 0x6138, 0x305a, 0x482a, 0x484a, 0x4e31,
01497 0x613d, 0x613b, 0x435c, 0x4026, 0x482b, 0x492d, 0x613f, 0x4e2c,
01498 0x374d, 0x6140, 0x613e, 0x4856, 0x6141, 0x6142, 0x305b, 0x3e76,
01499 0x6147, 0x6144, 0x466d, 0x6143, 0x3526, 0x614a, 0x6145, 0x6146,
01500 0x6149, 0x6148, 0x4925, 0x4142, 0x4141, 0x353f, 0x614b, 0x614c,
01501 0x614d, 0x614f, 0x614e, 0x3156, 0x6157, 0x4868, 0x6151, 0x6153,
01502 0x6155, 0x3f3e, 0x6156, 0x6154, 0x3c40, 0x6150, 0x6152, 0x4942,
01503 0x3e49, 0x6159, 0x6158, 0x615a, 0x3c26, 0x3a2f, 0x4577, 0x615b,
01504 0x444b, 0x615d, 0x4e21, 0x615c, 0x4169, 0x6162, 0x6164, 0x6165,
01505 0x4354, 0x6163, 0x6160, 0x615e, 0x615f, 0x6161, 0x6168, 0x6166,
01506 0x6167, 0x6169, 0x616b, 0x616c, 0x616d, 0x616e, 0x616a, 0x6170,
01507 0x616f, 0x6171, 0x4e45, 0x6174, 0x6172, 0x6173, 0x3462, 0x4c7e,
01508 0x4a4a, 0x6176, 0x6175, 0x6177, 0x6178, 0x617c, 0x6179, 0x617a,
01509 0x617b, 0x617d, 0x617e, 0x6221, 0x6222, 0x6223, 0x482f, 0x4550,
01510 0x6224, 0x4772, 0x4934, 0x6225, 0x6226, 0x452a, 0x3327, 0x3944,
01511 0x6227, 0x6228, 0x6229, 0x3b29, 0x622b, 0x622a, 0x622c, 0x622d,
01512 0x4869, 0x622e, 0x622f, 0x7369, 0x6230, 0x6231, 0x6232, 0x3b2e,
01513 0x6233, 0x4756, 0x4b5f, 0x314e, 0x3157, 0x6234, 0x6236, 0x6235,
01514 0x4570, 0x4039, 0x5d39, 0x6237, 0x4c41, 0x6238, 0x3446, 0x4857,
01515 0x6239, 0x623a, 0x623b, 0x4c5c, 0x4c55, 0x443e, 0x416a, 0x623d,
01516 0x3d62, 0x3e4a, 0x6240, 0x623f, 0x623e, 0x487d, 0x3447, 0x3829,
01517 0x6246, 0x6243, 0x3f3f, 0x4c32, 0x6242, 0x6244, 0x6245, 0x6241,
01518 0x6247, 0x6248, 0x442f, 0x3463, 0x4365, 0x6249, 0x624a, 0x624d,
01519 0x3f67, 0x4644, 0x624e, 0x4b53, 0x624b, 0x624c, 0x6251, 0x6250,
01520 0x624f, 0x6253, 0x6252, 0x6254, 0x6256, 0x6255, 0x4a4d, 0x3d56,
01521 0x4e46, 0x6257, 0x4637, 0x6258, 0x6259, 0x625d, 0x625b, 0x625c,
01522 0x625a, 0x625e, 0x625f, 0x6260, 0x6261, 0x4c37, 0x6262, 0x4c70,
01523 0x6263, 0x434e, 0x476a, 0x366b, 0x433b, 0x6264, 0x363a, 0x4050,
01524 0x6265, 0x3a3d, 0x6266, 0x6267, 0x3826, 0x3a55, 0x6269, 0x4556,
01525 0x3a56, 0x354e, 0x4b24, 0x474b, 0x4557, 0x395c, 0x626b, 0x3e4b,
01526 0x4e32, 0x3945, 0x3827, 0x4823, 0x626d, 0x626f, 0x386b, 0x626e,
01527 0x4476, 0x6271, 0x3337, 0x626c, 0x486a, 0x3130, 0x3a6c, 0x4f52,
01528 0x6270, 0x6272, 0x4a4b, 0x4059, 0x6274, 0x6275, 0x6273, 0x334e,
01529 0x627b, 0x627a, 0x3c27, 0x627c, 0x6277, 0x627d, 0x6278, 0x4858,
01530 0x6276, 0x6279, 0x6322, 0x6321, 0x4b61, 0x627e, 0x306b, 0x6324,
01531 0x6323, 0x3e4c, 0x6325, 0x4143, 0x6327, 0x6326, 0x6328, 0x6268,
01532 0x626a, 0x632a, 0x6329, 0x3c28, 0x4e69, 0x3c52, 0x632b, 0x3737,
01533 0x3540, 0x3527, 0x3b63, 0x4d34, 0x6331, 0x6330, 0x4144, 0x632d,
01534 0x632f, 0x3d4b, 0x3f40, 0x632e, 0x632c, 0x472a, 0x3e4d, 0x493c,
01535 0x3a57, 0x4578, 0x6332, 0x6333, 0x6349, 0x3658, 0x4f3d, 0x4135,
01536 0x6334, 0x3252, 0x4477, 0x4a21, 0x6335, 0x357a, 0x6336, 0x6338,
01537 0x6339, 0x4729, 0x633a, 0x633b, 0x633c, 0x3659, 0x3253, 0x4645,
01538 0x3d28, 0x3b64, 0x633d, 0x3d29, 0x324a, 0x4943, 0x633e, 0x486b,

```
01539 0x4145, 0x6341, 0x6342, 0x4769, 0x3f41, 0x633f, 0x4361, 0x6340,
01540 0x3e4e, 0x305c, 0x3529, 0x6343, 0x4478, 0x6344, 0x4047, 0x4c2d,
01541 0x4923, 0x6345, 0x6346, 0x4355, 0x4e47, 0x6348, 0x6347, 0x3c6f,
01542 0x634a, 0x3070, 0x634d, 0x634b, 0x3254, 0x374e, 0x634c, 0x3946,
01543 0x3972, 0x4a66, 0x634e, 0x4b54, 0x6350, 0x4051, 0x314f, 0x323a,
01544 0x302c, 0x634f, 0x6351, 0x6352, 0x3e77, 0x6353, 0x334f, 0x6355,
01545 0x376a, 0x3566, 0x6356, 0x3675, 0x6357, 0x407c, 0x464d, 0x4060,
01546 0x3a75, 0x6358, 0x4362, 0x416b, 0x635a, 0x635c, 0x6359, 0x635b,
01547 0x3722, 0x635d, 0x3726, 0x3567, 0x4d52, 0x635f, 0x6360, 0x312e,
01548 0x6363, 0x3376, 0x6362, 0x6361, 0x6365, 0x635e, 0x6366, 0x4e29,
01549 0x6367, 0x6368, 0x5474, 0x636a, 0x6369, 0x636b, 0x636c, 0x4e35,
01550 0x636d, 0x706f, 0x3e4f, 0x636e, 0x636f, 0x3d57, 0x4638, 0x6370,
01551 0x4328, 0x6371, 0x433c, 0x6372, 0x3625, 0x513f, 0x435d, 0x3c33,
01552 0x3448, 0x6373, 0x6422, 0x6376, 0x3568, 0x6375, 0x6424, 0x6374,
01553 0x3e50, 0x6378, 0x6379, 0x452b, 0x637a, 0x335e, 0x3f5a, 0x4964,
01554 0x637c, 0x4268, 0x6377, 0x637b, 0x637d, 0x3a7b, 0x6426, 0x492e,
01555 0x4826, 0x4579, 0x365a, 0x6425, 0x6423, 0x4835, 0x637e, 0x435e,
01556 0x457b, 0x457a, 0x3a76, 0x6438, 0x6428, 0x642a, 0x642d, 0x642e,
01557 0x642b, 0x642c, 0x6429, 0x6427, 0x6421, 0x4a4f, 0x3255, 0x6435,
01558 0x6432, 0x6437, 0x6436, 0x4773, 0x4c27, 0x3b3b, 0x6430, 0x6439,
01559 0x6434, 0x6433, 0x642f, 0x6431, 0x3449, 0x433d, 0x407d, 0x4822,
01560 0x643e, 0x4824, 0x4061, 0x643b, 0x484f, 0x643f, 0x4a53, 0x435b,
01561 0x643a, 0x643c, 0x643d, 0x6440, 0x3c44, 0x4646, 0x6445, 0x6444,
01562 0x6441, 0x4f36, 0x644a, 0x644e, 0x644b, 0x6447, 0x6448, 0x644d,
01563 0x6442, 0x5255, 0x6449, 0x6443, 0x644c, 0x6452, 0x344a, 0x644f,
01564 0x6450, 0x6451, 0x6454, 0x6453, 0x4876, 0x6455, 0x4e7c, 0x4a6d,
01565 0x645a, 0x6457, 0x6456, 0x4052, 0x6459, 0x645b, 0x6458, 0x645f,
01566 0x645c, 0x645d, 0x6446, 0x645e, 0x6460, 0x6461, 0x4a46, 0x6462,
01567 0x4c62, 0x364e, 0x3729, 0x6463, 0x4a34, 0x3f68, 0x4c30, 0x6464,
01568 0x4e33, 0x4774, 0x4146, 0x4734, 0x3d4d, 0x3040, 0x6469, 0x6467,
01569 0x6465, 0x3421, 0x3e51, 0x646a, 0x6468, 0x6466, 0x646e, 0x646d,
01570 0x646c, 0x646b, 0x646f, 0x6470, 0x403a, 0x6471, 0x6473, 0x6472,
01571 0x3852, 0x4138, 0x6475, 0x457c, 0x6474, 0x6476, 0x4a35, 0x416c,
01572 0x3947, 0x6477, 0x4e48, 0x6479, 0x647a, 0x647b, 0x647c, 0x3b65,
01573 0x647d, 0x374f, 0x356a, 0x352a, 0x6521, 0x4c73, 0x3948, 0x647e,
01574 0x6524, 0x4c66, 0x473c, 0x4933, 0x3d63, 0x6523, 0x3c53, 0x3949,
01575 0x3b66, 0x3569, 0x4a36, 0x6522, 0x4147, 0x4b42, 0x3a77, 0x3b67,
01576 0x445d, 0x6527, 0x4e5f, 0x3a59, 0x6528, 0x3f42, 0x652a, 0x3e52,
01577 0x3a30, 0x6529, 0x3d2a, 0x383e, 0x4148, 0x6525, 0x652b, 0x6526,
01578 0x3750, 0x652e, 0x6532, 0x376b, 0x652d, 0x6536, 0x394a, 0x4d6d,
01579 0x303c, 0x6533, 0x356b, 0x6530, 0x6531, 0x457d, 0x652f, 0x652c,
01580 0x3328, 0x4064, 0x3828, 0x6538, 0x6537, 0x6539, 0x6534, 0x3751,
01581 0x4233, 0x6539, 0x416e, 0x6546, 0x6542, 0x653c, 0x6540, 0x3c7a,
01582 0x305d, 0x653b, 0x6543, 0x6547, 0x394b, 0x4c56, 0x4456, 0x653d,
01583 0x6545, 0x653a, 0x433e, 0x653f, 0x303d, 0x4c4a, 0x653e, 0x3b6b,
01584 0x486c, 0x416d, 0x4e50, 0x3d6f, 0x656e, 0x6548, 0x407e, 0x6544,
01585 0x6549, 0x654b, 0x4479, 0x654e, 0x654a, 0x4a54, 0x344b, 0x4c4b,
01586 0x305e, 0x654d, 0x4e7d, 0x654c, 0x316f, 0x466c, 0x654f, 0x6556,
01587 0x6550, 0x6557, 0x6553, 0x477b, 0x3c4a, 0x6555, 0x6552, 0x6558,
01588 0x6551, 0x3d44, 0x4b25, 0x3d4c, 0x6554, 0x6560, 0x655c, 0x655f,
01589 0x655d, 0x6561, 0x655b, 0x6541, 0x4053, 0x484b, 0x655e, 0x6559,
01590 0x4121, 0x3752, 0x3d2b, 0x3f25, 0x4136, 0x6564, 0x6566, 0x6567,
01591 0x6563, 0x6565, 0x655a, 0x6562, 0x656a, 0x6569, 0x4b7a, 0x372b,
01592 0x6568, 0x656c, 0x656b, 0x656f, 0x6571, 0x3b3c, 0x656d, 0x6572,
01593 0x6573, 0x6574, 0x657a, 0x453b, 0x6576, 0x6575, 0x6577, 0x6578,
01594 0x6579, 0x657b, 0x657c, 0x344c, 0x657d, 0x657e, 0x6621, 0x6622,
01595 0x6623, 0x6624, 0x6625, 0x6626, 0x6628, 0x6627, 0x6629, 0x662a,
01596 0x662b, 0x662e, 0x662c, 0x662d, 0x3a61, 0x3753, 0x4356, 0x4833,
01597 0x3d70, 0x474d, 0x486d, 0x662f, 0x586d, 0x6630, 0x6632, 0x4d65,
01598 0x6631, 0x6634, 0x6633, 0x4d53, 0x6635, 0x487e, 0x6636, 0x6639,
01599 0x6638, 0x6637, 0x663a, 0x3732, 0x4122, 0x3541, 0x663e, 0x663b,
01600 0x663c, 0x663f, 0x6640, 0x663d, 0x3129, 0x3227, 0x6642, 0x6643,
01601 0x6644, 0x4d62, 0x3d2c, 0x6646, 0x6645, 0x3f69, 0x6647, 0x6648,
01602 0x6649, 0x3465, 0x344d, 0x664a, 0x664b, 0x4b5d, 0x4d63, 0x4d54,
01603 0x4f37, 0x394d, 0x664e, 0x3c54, 0x664d, 0x664f, 0x3c29, 0x4251,
01604 0x6650, 0x394c, 0x4c57, 0x6651, 0x6652, 0x6653, 0x6654, 0x6655,
01605 0x3c2a, 0x4c6d, 0x6657, 0x433f, 0x6656, 0x6659, 0x6658, 0x665a,
01606 0x403b, 0x665b, 0x665c, 0x4a39, 0x665d, 0x416f, 0x665e, 0x665f,
01607 0x4e7e, 0x6662, 0x6661, 0x6660, 0x4430, 0x6663, 0x3f26, 0x6664,
01608 0x6665, 0x4f38, 0x6666, 0x6667, 0x6669, 0x6668, 0x4825, 0x4679,
01609 0x4f3e, 0x4829, 0x666b, 0x3e53, 0x492a, 0x666c, 0x666a, 0x344e,
01610 0x3854, 0x3b68, 0x486e, 0x382a, 0x4b43, 0x666f, 0x666d, 0x394e,
01611 0x394f, 0x3069, 0x3a68, 0x4759, 0x305f, 0x6674, 0x4340, 0x4758,
01612 0x425b, 0x6676, 0x6672, 0x6675, 0x6670, 0x6673, 0x4b26, 0x3855,
01613 0x307d, 0x6671, 0x6678, 0x6679, 0x4639, 0x363b, 0x6726, 0x473d,
01614 0x3b69, 0x363c, 0x4048, 0x4f46, 0x4c2e, 0x6677, 0x4054, 0x3553,
01615 0x667a, 0x667c, 0x667b, 0x667d, 0x4326, 0x473e, 0x4431, 0x6723,
01616 0x6722, 0x667e, 0x3f55, 0x4965, 0x6725, 0x6724, 0x3950, 0x4f53,
01617 0x6735, 0x6729, 0x672a, 0x3c70, 0x6728, 0x3978, 0x6727, 0x672b,
01618 0x4432, 0x4a22, 0x4123, 0x425c, 0x672f, 0x6730, 0x672c, 0x672d,
01619 0x672e, 0x3951, 0x6736, 0x6732, 0x4966, 0x4b6c, 0x4928, 0x6731,
01620 0x6734, 0x6733, 0x4b44, 0x6737, 0x6738, 0x4137, 0x6739, 0x673b,
01621 0x673f, 0x673c, 0x673a, 0x473f, 0x673d, 0x673e, 0x3232, 0x6745,
01622 0x6740, 0x6741, 0x6742, 0x4221, 0x6744, 0x6743, 0x6746, 0x6747,
01623 0x6748, 0x3f43, 0x3269, 0x6749, 0x4e57, 0x3c2b, 0x3d2d, 0x3b6a,
01624 0x4357, 0x674a, 0x674b, 0x3131, 0x674c, 0x674d, 0x674e, 0x674f,
01625 0x6750, 0x363d, 0x5a2a, 0x6751, 0x4065, 0x6752, 0x3c4b, 0x6753,
```

```

01626 0x5030, 0x6754, 0x4a5e, 0x345c, 0x4124, 0x3d58, 0x4971, 0x3d2e,
01627 0x6755, 0x3952, 0x6756, 0x484c, 0x6764, 0x6758, 0x4249, 0x4775,
01628 0x383f, 0x6757, 0x4125, 0x6759, 0x447a, 0x675b, 0x675a, 0x675d,
01629 0x675c, 0x675e, 0x6760, 0x675f, 0x344f, 0x6761, 0x6762, 0x6763,
01630 0x3a31, 0x4e49, 0x6765, 0x3f27, 0x3170, 0x6766, 0x6767, 0x6768,
01631 0x3072, 0x3257, 0x676a, 0x4967, 0x3c47, 0x676c, 0x3329, 0x3032,
01632 0x676b, 0x676e, 0x474e, 0x3f44, 0x3256, 0x4b27, 0x375d, 0x365c,
01633 0x676d, 0x326a, 0x3423, 0x3171, 0x6772, 0x4e6a, 0x425d, 0x4944,
01634 0x677e, 0x3257, 0x677c, 0x677a, 0x6771, 0x676f, 0x6770, 0x3c63,
01635 0x366c, 0x4377, 0x4651, 0x3151, 0x6774, 0x6773, 0x6779, 0x6775,
01636 0x6778, 0x4c50, 0x6777, 0x3258, 0x337d, 0x677b, 0x677d, 0x3754,
01637 0x6823, 0x682c, 0x682d, 0x302b, 0x6834, 0x3071, 0x682b, 0x682a,
01638 0x6825, 0x6824, 0x6822, 0x6821, 0x4363, 0x427b, 0x6827, 0x6826,
01639 0x6829, 0x4170, 0x3755, 0x3141, 0x6828, 0x3953, 0x4171, 0x683a,
01640 0x683b, 0x3259, 0x322e, 0x6838, 0x682e, 0x6836, 0x683d, 0x6837,
01641 0x6835, 0x6776, 0x6833, 0x682f, 0x3450, 0x6831, 0x683c, 0x6832,
01642 0x683e, 0x6830, 0x477c, 0x4d69, 0x6839, 0x684f, 0x6847, 0x3f7b,
01643 0x3546, 0x365d, 0x6842, 0x325b, 0x3e54, 0x6845, 0x3a5a, 0x4551,
01644 0x684a, 0x4a6e, 0x6841, 0x325a, 0x3856, 0x4929, 0x684b, 0x683f,
01645 0x6848, 0x6852, 0x6843, 0x6844, 0x463a, 0x6849, 0x6846, 0x4b28,
01646 0x684c, 0x3060, 0x6840, 0x684d, 0x684e, 0x476b, 0x6854, 0x685f,
01647 0x337e, 0x6862, 0x6850, 0x6855, 0x4d6e, 0x685e, 0x4d55, 0x4e2a,
01648 0x4378, 0x336b, 0x4972, 0x6864, 0x4621, 0x3031, 0x685d, 0x6859,
01649 0x4172, 0x6853, 0x685b, 0x6860, 0x472c, 0x302a, 0x6858, 0x6861,
01650 0x4978, 0x685c, 0x6857, 0x3e55, 0x3d2f, 0x3c2c, 0x4c58, 0x4947,
01651 0x6867, 0x6870, 0x685a, 0x3377, 0x3e78, 0x6865, 0x686a, 0x4173,
01652 0x6866, 0x686d, 0x435f, 0x686e, 0x4d56, 0x6863, 0x3338, 0x6869,
01653 0x686c, 0x4c2c, 0x686f, 0x6868, 0x686b, 0x4b29, 0x4f21, 0x6873,
01654 0x687a, 0x6872, 0x3c43, 0x6851, 0x4a4e, 0x4c22, 0x6879, 0x6878,
01655 0x6874, 0x6875, 0x3136, 0x6877, 0x6871, 0x4455, 0x6876, 0x307e,
01656 0x4222, 0x4a43, 0x687b, 0x6921, 0x4859, 0x687e, 0x3e56, 0x3c49,
01657 0x6923, 0x363e, 0x6924, 0x4979, 0x687d, 0x6856, 0x687c, 0x4f4f,
01658 0x4622, 0x4973, 0x692b, 0x6931, 0x6932, 0x6925, 0x4776, 0x692f,
01659 0x6927, 0x6929, 0x6933, 0x6928, 0x692c, 0x3172, 0x4665, 0x692d,
01660 0x6930, 0x6926, 0x4126, 0x692a, 0x3b27, 0x3f45, 0x3730, 0x4c74,
01661 0x4c79, 0x3d72, 0x6937, 0x6935, 0x4f4e, 0x6934, 0x4d75, 0x6936,
01662 0x6938, 0x6939, 0x693c, 0x693a, 0x4623, 0x693b, 0x484d, 0x692e,
01663 0x3d73, 0x693d, 0x6942, 0x4174, 0x6941, 0x6922, 0x6943, 0x4149,
01664 0x693e, 0x6940, 0x693f, 0x5d31, 0x5d22, 0x6945, 0x6944, 0x4d76,
01665 0x623c, 0x6946, 0x6947, 0x6948, 0x3857, 0x3554, 0x694a, 0x515d,
01666 0x3575, 0x4e3a, 0x3673, 0x694b, 0x694c, 0x436e, 0x694d, 0x467a,
01667 0x303a, 0x3263, 0x6952, 0x6953, 0x694e, 0x3b3d, 0x694f, 0x4742,
01668 0x6950, 0x6951, 0x695b, 0x6955, 0x6958, 0x6954, 0x6956, 0x6957,
01669 0x3c58, 0x6959, 0x4341, 0x3756, 0x3342, 0x695c, 0x333f, 0x6961,
01670 0x695d, 0x6960, 0x483a, 0x695e, 0x695f, 0x4948, 0x485a, 0x6962,
01671 0x427d, 0x696c, 0x6968, 0x326b, 0x6966, 0x4b2a, 0x6967, 0x6964,
01672 0x6965, 0x696a, 0x696d, 0x696b, 0x6969, 0x6963, 0x4358, 0x6974,
01673 0x4c2a, 0x6972, 0x6973, 0x696e, 0x6970, 0x6971, 0x696f, 0x4066,
01674 0x4f39, 0x6978, 0x6979, 0x6a21, 0x3f2a, 0x697b, 0x697e, 0x6976,
01675 0x6975, 0x6a22, 0x325c, 0x697c, 0x6a23, 0x697d, 0x697a, 0x4433,
01676 0x6977, 0x4768, 0x6a27, 0x4d3b, 0x6a26, 0x6a25, 0x6a2e, 0x6a28,
01677 0x6a30, 0x4d66, 0x6a33, 0x6a2a, 0x6a2b, 0x6a2f, 0x6a32, 0x6a31,
01678 0x6a29, 0x6a2c, 0x6a3d, 0x6a36, 0x6a34, 0x6a35, 0x6a3a, 0x6a3b,
01679 0x332a, 0x3542, 0x6a39, 0x6a38, 0x6a3c, 0x6a37, 0x6a3e,
01680 0x6a40, 0x6a3f, 0x6a42, 0x6a41, 0x695a, 0x6a46, 0x6a43, 0x6a44,
01681 0x6a45, 0x6a47, 0x376c, 0x6a49, 0x6a48, 0x3d30, 0x3954, 0x5e27,
01682 0x6a4a, 0x3d51, 0x3339, 0x6a4b, 0x3152, 0x3e57, 0x6a4c, 0x3955,
01683 0x6a4d, 0x3061, 0x493d, 0x6a4e, 0x3f6a, 0x6a55, 0x6a52, 0x436f,
01684 0x6a53, 0x6a50, 0x365e, 0x6a4f, 0x6a56, 0x3736, 0x425e, 0x6a5c,
01685 0x6a58, 0x4235, 0x6a57, 0x6a5a, 0x6a51, 0x6a5b, 0x6a5d, 0x486f,
01686 0x6a59, 0x6a5e, 0x6a60, 0x3853, 0x6a54, 0x3041, 0x6a5f, 0x3a5b,
01687 0x4e76, 0x6a61, 0x6a62, 0x4175, 0x4e22, 0x6a63, 0x4d35, 0x6a64,
01688 0x6a65, 0x4a64, 0x6a66, 0x3a40, 0x4e23, 0x6a6b, 0x6a6c, 0x3e58,
01689 0x6a6a, 0x4d67, 0x6a67, 0x6a69, 0x403d, 0x3f7e, 0x6a68, 0x6a6d,
01690 0x4a23, 0x6a6f, 0x6a6e, 0x336c, 0x4b2b, 0x6a70, 0x6a7c, 0x6a72,
01691 0x6a73, 0x6a74, 0x6a75, 0x6a79, 0x6a7a, 0x6a78, 0x6a76, 0x6a71,
01692 0x6a77, 0x6a7b, 0x7037, 0x3228, 0x6a7e, 0x365f, 0x6a7d, 0x6b22,
01693 0x6b21, 0x6b24, 0x6b23, 0x6b25, 0x3d31, 0x6b26, 0x6b27, 0x6b28,
01694 0x403e, 0x4d57, 0x6b29, 0x4a24, 0x4746, 0x6b2a, 0x6b2b, 0x382b,
01695 0x352c, 0x6b2c, 0x3b6b, 0x4741, 0x6b2d, 0x3350, 0x6b2e, 0x6b30,
01696 0x4d77, 0x6b2f, 0x3f46, 0x6b31, 0x6b32, 0x6b33, 0x3451, 0x6b34,
01697 0x6b35, 0x6b36, 0x6b37, 0x3351, 0x6b38, 0x6b39, 0x6b3a, 0x3272,
01698 0x3f28, 0x6b3b, 0x6b3c, 0x6b3d, 0x3840, 0x447b, 0x6b3e, 0x3757,
01699 0x3f56, 0x6b41, 0x4624, 0x6b40, 0x3731, 0x6b3f, 0x4277, 0x352d,
01700 0x6b42, 0x6b43, 0x3e59, 0x376d, 0x6b44, 0x4b2c, 0x405f, 0x3576,
01701 0x4c75, 0x414a, 0x6b45, 0x3f47, 0x4370, 0x3e5a, 0x6b46, 0x6b49,
01702 0x6b4a, 0x3a3e, 0x4242, 0x6b48, 0x3e5b, 0x493e, 0x6b47, 0x3b6c,
01703 0x3153, 0x6b4e, 0x3758, 0x3b6e, 0x3b6d, 0x4f4d, 0x6b4d, 0x6b4c,
01704 0x4127, 0x354d, 0x4f43, 0x333a, 0x3e5c, 0x6b4b, 0x6b50, 0x6b51,
01705 0x6b4f, 0x3858, 0x4d40, 0x3b6f, 0x4727, 0x6b54, 0x4040, 0x4342,
01706 0x4d36, 0x6b57, 0x386c, 0x403f, 0x6b53, 0x6b58, 0x386d, 0x6b55,
01707 0x6b56, 0x6b52, 0x4062, 0x4649, 0x432f, 0x325d, 0x4870, 0x3543,
01708 0x4434, 0x6b5b, 0x6b59, 0x434c, 0x4041, 0x3452, 0x6b5a, 0x3f5b,
01709 0x4e4a, 0x4f40, 0x6b5c, 0x6b67, 0x4435, 0x6b66, 0x6b63, 0x6b6b,
01710 0x6b64, 0x6b60, 0x447c, 0x6b5f, 0x6b5d, 0x4d21, 0x3b70, 0x6b61,
01711 0x6b5e, 0x6b65, 0x3d74, 0x3841, 0x427a, 0x4b45, 0x315a, 0x3062,
01712 0x4625, 0x6b69, 0x6b68, 0x4666, 0x6b6d, 0x6b62, 0x6b6c, 0x6b6e,

```

```
01713 0x382c, 0x6b6a, 0x3956, 0x3c55, 0x6b6f, 0x4d58, 0x6b72, 0x6b75,
01714 0x6b73, 0x4935, 0x6b70, 0x3660, 0x6b74, 0x6b76, 0x6b7a, 0x6b77,
01715 0x6b79, 0x6b78, 0x6b7b, 0x3c31, 0x6b7d, 0x6b7c, 0x4968, 0x6c21,
01716 0x3759, 0x6b7e, 0x6c22, 0x6c23, 0x3544, 0x6641, 0x3e79, 0x6c24,
01717 0x386e, 0x6c25, 0x6c26, 0x3b3e, 0x5a4e, 0x6c27, 0x6c28, 0x3d32,
01718 0x6c29, 0x6c2a, 0x6c2b, 0x6c2c, 0x6c2d, 0x432b, 0x6c2e, 0x6c30,
01719 0x6c2f, 0x4626, 0x6c31, 0x4b2d, 0x6c32, 0x6c33, 0x6c34, 0x6c35,
01720 0x465a, 0x3e5d, 0x6c36, 0x396b, 0x502e, 0x6c37, 0x6c38, 0x493f,
01721 0x6c39, 0x6c41, 0x6c3a, 0x6c3c, 0x6c3b, 0x6c3d, 0x4b46, 0x6c3e,
01722 0x6c3f, 0x6c40, 0x6c42, 0x332d, 0x4467, 0x4969, 0x3a62, 0x3957,
01723 0x494f, 0x325f, 0x484e, 0x6c45, 0x3453, 0x4055, 0x6c44, 0x6c49,
01724 0x4379, 0x4c63, 0x6c47, 0x6c48, 0x352e, 0x6c4a, 0x4763, 0x425f,
01725 0x4871, 0x453d, 0x6c46, 0x4b47, 0x326c, 0x6c4c, 0x4f28, 0x4442,
01726 0x4f45, 0x3b71, 0x6c4b, 0x4231, 0x6c5c, 0x4128, 0x4678, 0x4950,
01727 0x6c4f, 0x3b3f, 0x3b72, 0x3e5e, 0x4765, 0x382d, 0x6c4e, 0x6c4d,
01728 0x496a, 0x3c41, 0x4552, 0x6c51, 0x6c52, 0x3958, 0x6c50, 0x6c53,
01729 0x6c54, 0x6c56, 0x4223, 0x6c55, 0x3466, 0x6c58, 0x6c57, 0x6c59,
01730 0x6c5b, 0x6c5d, 0x6c5e, 0x4056, 0x3c4f, 0x6c5f, 0x3352, 0x6c60,
01731 0x4176, 0x6c61, 0x6c62, 0x496b, 0x352f, 0x6c63, 0x4436, 0x315b,
01732 0x6c64, 0x3c71, 0x3f76, 0x422d, 0x6c67, 0x6c66, 0x6c65, 0x6c6d,
01733 0x6c6b, 0x6c6c, 0x6c6a, 0x6c69, 0x6c6e, 0x3577, 0x6c70, 0x4057,
01734 0x6c71, 0x3859, 0x6c6e, 0x6c6f, 0x4f29, 0x4437, 0x4129, 0x6c72,
01735 0x6c75, 0x6c73, 0x6c74, 0x4d59, 0x4627, 0x6c78, 0x6c76, 0x6c77,
01736 0x6c79, 0x6d29, 0x6c7c, 0x6c7d, 0x6c7b, 0x6c7a, 0x447d, 0x6d21,
01737 0x6d25, 0x6d22, 0x6c7e, 0x6d23, 0x6d24, 0x6d2b, 0x6d26, 0x4058,
01738 0x6d28, 0x6d2a, 0x6d27, 0x6d2d, 0x3d33, 0x6d2c, 0x6d2e, 0x6d2f,
01739 0x6d32, 0x6d31, 0x6d30, 0x6d34, 0x6d33, 0x4c76, 0x6d36, 0x6d35,
01740 0x6d37, 0x6d38, 0x6d3a, 0x6d39, 0x3f48, 0x6d3b, 0x366d, 0x6d3c,
01741 0x6d3e, 0x6d3f, 0x6d40, 0x6d3d, 0x6d41, 0x3c56, 0x6d42, 0x3530,
01742 0x3733, 0x382e, 0x6d43, 0x4670, 0x453e, 0x6d44, 0x6d47, 0x3c34,
01743 0x6d46, 0x6d45, 0x375a, 0x6d48, 0x3353, 0x6d4a, 0x3a5c, 0x6d49,
01744 0x6d52, 0x6d4c, 0x6d4e, 0x4a65, 0x6d4b, 0x6d4d, 0x6d51, 0x6d4f,
01745 0x3531, 0x6d50, 0x6d53, 0x475a, 0x4e58, 0x3d34, 0x6d54, 0x4d22,
01746 0x6d56, 0x6d55, 0x6d59, 0x4d41, 0x6d58, 0x336d, 0x6d57, 0x6d5c,
01747 0x6d5b, 0x6d5a, 0x4532, 0x6d5d, 0x6d5e, 0x6d5f, 0x396c, 0x3725,
01748 0x6d60, 0x6d61, 0x6d62, 0x3f49, 0x6d63, 0x3c2d, 0x6d64, 0x6d65,
01749 0x5221, 0x517e, 0x6d66, 0x6570, 0x6d67, 0x4324, 0x3f2b, 0x4740,
01750 0x6d68, 0x4a55, 0x4454, 0x397e, 0x4329, 0x312a, 0x4b78, 0x3f57,
01751 0x375e, 0x3661, 0x4a56, 0x6d69, 0x6d6b, 0x6d6a, 0x3260, 0x4676,
01752 0x6d6c, 0x4777, 0x4533, 0x6d6d, 0x3d52, 0x6d6f, 0x4c42, 0x6d7e,
01753 0x6d71, 0x6d72, 0x4449, 0x4260, 0x4177, 0x4628, 0x6d70, 0x3555,
01754 0x6d79, 0x6d76, 0x6e25, 0x4629, 0x4360, 0x6d73, 0x447e, 0x4553,
01755 0x6d74, 0x6d78, 0x3f60, 0x4767, 0x444c, 0x4042, 0x6d77, 0x422e,
01756 0x4224, 0x6d75, 0x3029, 0x4f22, 0x6d7a, 0x4261, 0x3d35, 0x3f4a,
01757 0x6d7c, 0x6d7b, 0x306f, 0x6d7d, 0x492f, 0x6e27, 0x465b, 0x3f6b,
01758 0x4359, 0x3678, 0x6e26, 0x4d37, 0x313f, 0x4a57, 0x3261, 0x6e21,
01759 0x6e22, 0x6e23, 0x6e24, 0x463b, 0x4323, 0x3063, 0x6e28, 0x6e29,
01760 0x7423, 0x423d, 0x6e2a, 0x3173, 0x414c, 0x382f, 0x4d5a, 0x6e2b,
01761 0x452c, 0x4178, 0x3c57, 0x6e2c, 0x6e2f, 0x3d65, 0x6e2d, 0x412b,
01762 0x412a, 0x3064, 0x4e4b, 0x6e31, 0x4872, 0x6e33, 0x6e32, 0x6e30,
01763 0x6364, 0x3454, 0x6d6e, 0x6e35, 0x6e34, 0x6e36, 0x4d38, 0x4661,
01764 0x4b2e, 0x6e37, 0x3c59, 0x6e38, 0x6e39, 0x6e3a, 0x4521, 0x306a,
01765 0x3959, 0x4f3a, 0x6e3e, 0x3734, 0x6e3b, 0x6e3c, 0x4974, 0x3354,
01766 0x4d39, 0x363f, 0x4554, 0x6e3f, 0x6e40, 0x6e41, 0x4522, 0x6e43,
01767 0x6e42, 0x4653, 0x6e44, 0x3d36, 0x3c60, 0x475b, 0x4371, 0x3c72,
01768 0x3f6c, 0x6e45, 0x6e46, 0x3f5d, 0x6e47, 0x6e48, 0x6e49, 0x4d6f,
01769 0x3d37, 0x6e4b, 0x6e4a, 0x395a, 0x3973, 0x3b40, 0x6e4e, 0x3d66,
01770 0x6e4d, 0x6e4c, 0x4269, 0x386f, 0x4043, 0x4830, 0x3d39, 0x6e4f,
01771 0x3e5f, 0x6e52, 0x6e50, 0x6e51, 0x6e54, 0x6e53, 0x3e7a, 0x6e55,
01772 0x6e56, 0x6e57, 0x4850, 0x3a53, 0x3c61, 0x6e58, 0x6e59, 0x4e24,
01773 0x3d45, 0x4c6e, 0x4e4c, 0x6e5a, 0x3662, 0x6e5b, 0x4523, 0x6e5e,
01774 0x3378, 0x3f4b, 0x6e5c, 0x6e5d, 0x4460, 0x4b55, 0x367c, 0x6e60,
01775 0x6e61, 0x6e5f, 0x6e63, 0x465f, 0x3343, 0x6e67, 0x6e64, 0x6e66,
01776 0x6e62, 0x6f4f, 0x6e65, 0x4e6b, 0x385a, 0x6e6f, 0x4534, 0x6e6a,
01777 0x6e6d, 0x6e6b, 0x6e70, 0x6e71, 0x6e69, 0x6e76, 0x3174, 0x6e68,
01778 0x482d, 0x6e6c, 0x3e60, 0x395b, 0x4b48, 0x3664, 0x3d46, 0x463c,
01779 0x412d, 0x6e74, 0x6e6e, 0x6e73, 0x4c43, 0x4438, 0x6e75, 0x6e72,
01780 0x412c, 0x6e79, 0x6e78, 0x6e77, 0x4b2f, 0x3d7b, 0x6e7a, 0x4a5f,
01781 0x3154, 0x4946, 0x4372, 0x3578, 0x6e7c, 0x395d, 0x3b2c, 0x6e7b,
01782 0x3f6d, 0x3f6e, 0x6f21, 0x6f23, 0x3e7b, 0x6f22, 0x6f24, 0x3653,
01783 0x4945, 0x3c62, 0x4f23, 0x6e7e, 0x3a78, 0x4f3f, 0x6f26, 0x6f25,
01784 0x6f27, 0x6e7d, 0x4669, 0x4555, 0x4457, 0x6f2c, 0x4343, 0x6f28,
01785 0x6f29, 0x372d, 0x6f2b, 0x3830, 0x6f2a, 0x3e61, 0x3379, 0x6f30,
01786 0x3a3f, 0x4179, 0x444a, 0x333b, 0x6f2e, 0x6f2f, 0x4443, 0x6f2d,
01787 0x6f31, 0x6f37, 0x6f3a, 0x6f39, 0x452d, 0x6f32, 0x6f33, 0x6f36,
01788 0x6f38, 0x3640, 0x6f3b, 0x6f35, 0x6f34, 0x6f3f, 0x6f40, 0x6f41,
01789 0x6f3e, 0x6f3d, 0x3e62, 0x462a, 0x6f3c, 0x6f45, 0x6f43, 0x6f44,
01790 0x6f42, 0x4278, 0x6f46, 0x6f47, 0x6f49, 0x3455, 0x6f48, 0x4c7a,
01791 0x6f54, 0x6f4a, 0x6f4d, 0x6f4b, 0x6f4c, 0x6f4e, 0x6f50, 0x6f51,
01792 0x6f52, 0x6f55, 0x6f53, 0x6f56, 0x6f58, 0x6f57, 0x4439, 0x4c67,
01793 0x6f59, 0x412e, 0x6f5a, 0x4a44, 0x6f5b, 0x332b, 0x313c, 0x3457,
01794 0x3456, 0x6f5c, 0x6f5d, 0x6f5e, 0x6f5f, 0x6f60, 0x3458, 0x3355,
01795 0x395e, 0x4836, 0x6f62, 0x6f61, 0x6f63, 0x315c, 0x6f66, 0x6f65,
01796 0x6f64, 0x6f67, 0x6f6a, 0x3047, 0x6f68, 0x6f6c, 0x6f6b, 0x6f6e,
01797 0x6f6d, 0x6f6f, 0x462e, 0x6f70, 0x6f71, 0x6f73, 0x6f72, 0x496c,
01798 0x6f74, 0x6f75, 0x3a65, 0x6f76, 0x6f77, 0x4b49, 0x414b, 0x3024,
01799 0x424b, 0x6f78, 0x496d, 0x6f7b, 0x6f79, 0x395f, 0x6f7a, 0x3842,
```

```
01800 0x4a45, 0x6f7d, 0x7021, 0x6f7e, 0x7022, 0x3121, 0x3f58, 0x3d7c,
01801 0x3459, 0x7023, 0x4766, 0x7025, 0x3122, 0x7024, 0x4444, 0x4e4d,
01802 0x462b, 0x6f7c, 0x4e26, 0x3831, 0x4d5b, 0x3679, 0x4e34, 0x3728,
01803 0x4262, 0x6721, 0x7026, 0x332c, 0x3f6f, 0x3356, 0x7028, 0x7029,
01804 0x7027, 0x3764, 0x3a5d, 0x3e63, 0x3123, 0x4e59, 0x702b, 0x6e2e,
01805 0x702a, 0x702e, 0x702c, 0x702d, 0x702f, 0x7030, 0x4e6c, 0x7031,
01806 0x7032, 0x4049, 0x483b, 0x3f7d, 0x3467, 0x4d3a, 0x326d, 0x3d38,
01807 0x385b, 0x7035, 0x7034, 0x3b73, 0x7036, 0x7033, 0x3b28, 0x703a,
01808 0x6a2d, 0x5256, 0x3f77, 0x7038, 0x4e25, 0x4671, 0x312b, 0x4063,
01809 0x3c36, 0x4a37, 0x3140, 0x4e6d, 0x4d6b, 0x703b, 0x4545, 0x3c7b,
01810 0x703c, 0x703d, 0x3f4c, 0x703e, 0x4e6e, 0x7039, 0x7040, 0x7042,
01811 0x7041, 0x703f, 0x7043, 0x7044, 0x417a, 0x3262, 0x7045, 0x4c38,
01812 0x7046, 0x7047, 0x4f2a, 0x5b31, 0x7048, 0x7049, 0x704a, 0x704e,
01813 0x704b, 0x704c, 0x704d, 0x704f, 0x4044, 0x4c77, 0x4045, 0x7050,
01814 0x4873, 0x7051, 0x7353, 0x4c4c, 0x7052, 0x7053, 0x7054, 0x3357,
01815 0x7056, 0x3f59, 0x7057, 0x3724, 0x7058, 0x705c, 0x705a, 0x705b,
01816 0x3373, 0x7059, 0x705d, 0x705e, 0x3048, 0x705f, 0x7060, 0x3e64,
01817 0x7061, 0x3547, 0x7064, 0x7063, 0x7062, 0x6b71, 0x4a5c, 0x7065,
01818 0x7066, 0x7067, 0x7068, 0x7069, 0x706a, 0x345a, 0x706b, 0x706c,
01819 0x4723, 0x706e, 0x323b, 0x7071, 0x7070, 0x3124, 0x3641, 0x4a47,
01820 0x443a, 0x3a22, 0x3960, 0x3d67, 0x3f5c, 0x7073, 0x7072, 0x4d42,
01821 0x3468, 0x4852, 0x465c, 0x3f7c, 0x4e4e, 0x375b, 0x7076, 0x7075,
01822 0x4b4b, 0x462c, 0x3150, 0x7077, 0x7074, 0x4951, 0x4d6a, 0x7078,
01823 0x7079, 0x707b, 0x426a, 0x335b, 0x335c, 0x707a, 0x3469, 0x3832,
01824 0x346a, 0x453f, 0x4e60, 0x385c, 0x707c, 0x707d, 0x707e, 0x7121,
01825 0x7123, 0x7122, 0x4977, 0x7124, 0x7125, 0x7126, 0x7127, 0x7129,
01826 0x7128, 0x712a, 0x4874, 0x664c, 0x3f29, 0x3532, 0x712b, 0x712c,
01827 0x522c, 0x5d3b, 0x4853, 0x307b, 0x303b, 0x3b74, 0x4b30, 0x3e7e,
01828 0x712d, 0x4c5f, 0x712e, 0x4d5c, 0x3142, 0x3b41, 0x712f, 0x326e,
01829 0x7130, 0x7131, 0x7132, 0x7133, 0x7134, 0x7136, 0x7137, 0x345b,
01830 0x7137, 0x7138, 0x7139, 0x713a, 0x713b, 0x713d, 0x713c, 0x713f,
01831 0x7142, 0x713e, 0x7140, 0x7141, 0x7143, 0x3642, 0x3c73, 0x7144,
01832 0x7145, 0x3961, 0x7146, 0x333e, 0x474f, 0x7147, 0x7148, 0x435a,
01833 0x466b, 0x7149, 0x477d, 0x424c, 0x3158, 0x366e, 0x366f, 0x4373,
01834 0x714e, 0x3670, 0x326f, 0x714d, 0x714b, 0x714c, 0x714a, 0x7158,
01835 0x714f, 0x7150, 0x7151, 0x7152, 0x7154, 0x7153, 0x3d59, 0x7155,
01836 0x7157, 0x3533, 0x7156, 0x417b, 0x3833, 0x7159, 0x424d, 0x715a,
01837 0x462d, 0x715b, 0x7160, 0x715e, 0x715d, 0x715f, 0x715c, 0x7162,
01838 0x7161, 0x7164, 0x3643, 0x7163, 0x7165, 0x7166, 0x7168, 0x7167,
01839 0x7169, 0x716b, 0x716a, 0x397c, 0x716c, 0x716d, 0x333c, 0x716e,
01840 0x716f, 0x3f71, 0x7170, 0x7171, 0x7172, 0x7173, 0x3962, 0x7174,
01841 0x7175, 0x7176, 0x7177, 0x7178, 0x4831, 0x717a, 0x4926, 0x717b,
01842 0x7179, 0x717d, 0x717c, 0x717e, 0x7221, 0x7222, 0x7223, 0x7224,
01843 0x7225, 0x7226, 0x7227, 0x7228, 0x7229, 0x722a, 0x722b, 0x722c,
01844 0x722d, 0x722e, 0x5d35, 0x722f, 0x6478, 0x3534, 0x3321, 0x3a32,
01845 0x7231, 0x7230, 0x4c25, 0x7233, 0x7234, 0x7232, 0x7235, 0x4b62,
01846 0x7236, 0x357b, 0x4f25, 0x7237, 0x7239, 0x303e, 0x723a, 0x4a2b,
01847 0x7238, 0x723b, 0x723c, 0x723d, 0x723e, 0x723f, 0x4b6e, 0x3b2d,
01848 0x3a7a, 0x412f, 0x7240, 0x7243, 0x7241, 0x7244, 0x3871, 0x7242,
01849 0x7245, 0x7246, 0x7247, 0x724b, 0x3b2a, 0x4264, 0x724c, 0x7249,
01850 0x7248, 0x724a, 0x375f, 0x7250, 0x724f, 0x724e, 0x3033, 0x725a,
01851 0x7256, 0x7257, 0x7253, 0x7259, 0x7255, 0x3362, 0x4f4c, 0x7258,
01852 0x7254, 0x7252, 0x7251, 0x725c, 0x725f, 0x725e, 0x725d, 0x4949,
01853 0x725b, 0x3073, 0x7260, 0x7262, 0x336f, 0x724d, 0x3137, 0x7264,
01854 0x7263, 0x7261, 0x432d, 0x4b70, 0x4e5a, 0x7265, 0x7266, 0x7267,
01855 0x7268, 0x7269, 0x443b, 0x726a, 0x4837, 0x726f, 0x726b, 0x726c,
01856 0x4b31, 0x4c44, 0x4650, 0x7270, 0x7271, 0x463e, 0x726e, 0x726d,
01857 0x322a, 0x7279, 0x7278, 0x3175, 0x7276, 0x7275, 0x7273, 0x337b,
01858 0x7272, 0x3c32, 0x3229, 0x3963, 0x727c, 0x727b, 0x727a, 0x7277,
01859 0x727d, 0x727e, 0x7325, 0x7324, 0x7326, 0x312d, 0x7321, 0x7322,
01860 0x3974, 0x4c39, 0x7323, 0x4b32, 0x732b, 0x7327, 0x732c, 0x7329,
01861 0x7328, 0x375c, 0x732d, 0x732e, 0x732f, 0x732a, 0x7274, 0x7330,
01862 0x4461, 0x7334, 0x7335, 0x7333, 0x7332, 0x7338, 0x7331, 0x7336,
01863 0x7337, 0x733a, 0x7339, 0x733c, 0x733d, 0x733e, 0x4f49, 0x733b,
01864 0x426b, 0x3a6d, 0x733f, 0x7340, 0x7341, 0x7342, 0x7343, 0x3834,
01865 0x7344, 0x7345, 0x3c2f, 0x7346, 0x7347, 0x7348, 0x7349, 0x734c,
01866 0x734a, 0x4f3c, 0x734b, 0x4e6f, 0x734d, 0x4e5b, 0x734e, 0x477e,
01867 0x734f, 0x7351, 0x7352, 0x7350, 0x396d, 0x4c4d, 0x4b63, 0x5677,
01868 0x5d60, 0x4b7b, 0x322b, 0x7354, 0x3550, 0x7355, 0x7356, 0x7357,
01869 0x3975, 0x7358, 0x6054, 0x4c5b, 0x4263, 0x7359, 0x735b, 0x735a,
01870 0x735c, 0x735d, 0x735e, 0x735f, 0x7360, 0x7361, 0x7362, 0x7363,
01871 0x7364, 0x7365, 0x7366, 0x7367, 0x7368, 0x4524, 0x385d, 0x736a,
01872 0x414d, 0x736b, 0x736c, 0x4921, 0x736d, 0x736e, 0x6337, 0x6c5a,
01873 0x706d, 0x736f, 0x7370, 0x7372, 0x7373, 0x7374, 0x4e70, 0x7371,
01874 0x7375, 0x7376, 0x7378, 0x7377, 0x737a, 0x737b, 0x7379, 0x4e36,
01875 0x737c, 0x737d, 0x6354, 0x737e, 0x212a, 0x2174, 0x2170, 0x2173,
01876 0x2175, 0x2176, 0x214a, 0x214b, 0x2176, 0x215c, 0x2124, 0x2125, 0x213f,
01877 0x2330, 0x2331, 0x2332, 0x2333, 0x2334, 0x2335, 0x2336, 0x2337,
01878 0x2338, 0x2339, 0x2127, 0x2128, 0x2163, 0x2161, 0x2164, 0x2129,
01879 0x2177, 0x2341, 0x2342, 0x2343, 0x2344, 0x2345, 0x2346, 0x2347,
01880 0x2348, 0x2349, 0x234a, 0x234b, 0x234c, 0x234d, 0x234e, 0x234f,
01881 0x2350, 0x2351, 0x2352, 0x2353, 0x2354, 0x2355, 0x2356, 0x2357,
01882 0x2358, 0x2359, 0x235a, 0x214e, 0x214f, 0x2130, 0x2132, 0x212e,
01883 0x2361, 0x2362, 0x2363, 0x2364, 0x2365, 0x2366, 0x2367, 0x2368,
01884 0x2369, 0x236a, 0x236b, 0x236c, 0x236d, 0x236e, 0x236f, 0x2370,
01885 0x2371, 0x2372, 0x2373, 0x2374, 0x2375, 0x2376, 0x2377, 0x2378,
01886 0x2379, 0x237a, 0x2150, 0x2143, 0x2151, 0x2131, 0x216f,
```

```
01887 };
01888
01889 static const Summary16 jisx0208_uni2indx_page00[16] = {
01890     /* 0x0000 */
01891     { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 },
01892     { 0, 0x0000 }, { 0, 0x1000 }, { 1, 0x0000 }, { 1, 0x0000 },
01893     { 1, 0x0000 }, { 1, 0x0000 }, { 1, 0x118c }, { 6, 0x0053 },
01894     { 10, 0x0000 }, { 10, 0x0080 }, { 11, 0x0000 }, { 11, 0x0080 },
01895 };
01896 static const Summary16 jisx0208_uni2indx_page03[22] = {
01897     /* 0x0300 */
01898     { 12, 0x0000 }, { 12, 0x0000 }, { 12, 0x0000 }, { 12, 0x0000 },
01899     { 12, 0x0000 }, { 12, 0x0000 }, { 12, 0x0000 }, { 12, 0x0000 },
01900     { 12, 0x0000 }, { 12, 0xffff }, { 27, 0x03fb }, { 36, 0xffff },
01901     { 51, 0x03fb }, { 60, 0x0000 }, { 60, 0x0000 }, { 60, 0x0000 },
01902     /* 0x0400 */
01903     { 60, 0x0002 }, { 61, 0xffff }, { 77, 0xffff }, { 93, 0xffff },
01904     { 109, 0xffff }, { 125, 0x0002 },
01905 };
01906 static const Summary16 jisx0208_uni2indx_page20[50] = {
01907     /* 0x2000 */
01908     { 126, 0x0000 }, { 126, 0x3361 }, { 133, 0x0063 }, { 137, 0x080d },
01909     { 141, 0x0000 }, { 141, 0x0000 }, { 141, 0x0000 }, { 141, 0x0000 },
01910     { 141, 0x0000 }, { 141, 0x0000 }, { 141, 0x0000 }, { 141, 0x0000 },
01911     { 141, 0x0000 }, { 141, 0x0000 }, { 141, 0x0000 }, { 141, 0x0000 },
01912     /* 0x2100 */
01913     { 141, 0x0008 }, { 142, 0x0000 }, { 142, 0x0800 }, { 143, 0x0000 },
01914     { 143, 0x0000 }, { 143, 0x0000 }, { 143, 0x0000 }, { 143, 0x0000 },
01915     { 143, 0x0000 }, { 143, 0x000f }, { 147, 0x0000 }, { 147, 0x0000 },
01916     { 147, 0x0000 }, { 147, 0x0014 }, { 149, 0x0000 }, { 149, 0x0000 },
01917     /* 0x2200 */
01918     { 149, 0x098d }, { 155, 0x6404 }, { 159, 0x1f81 }, { 166, 0x2030 },
01919     { 169, 0x0000 }, { 169, 0x0004 }, { 170, 0x0cc3 }, { 176, 0x0000 },
01920     { 176, 0x00cc }, { 180, 0x0000 }, { 180, 0x0020 }, { 181, 0x0000 },
01921     { 181, 0x0000 }, { 181, 0x0000 }, { 181, 0x0000 }, { 181, 0x0000 },
01922     /* 0x2300 */
01923     { 181, 0x0000 }, { 181, 0x0004 },
01924 };
01925 static const Summary16 jisx0208_uni2indx_page25[23] = {
01926     /* 0x2500 */
01927     { 182, 0x900f }, { 188, 0x3999 }, { 196, 0x9939 }, { 204, 0x9999 },
01928     { 212, 0x0804 }, { 214, 0x0000 }, { 214, 0x0000 }, { 214, 0x0000 },
01929     { 214, 0x0000 }, { 214, 0x0000 }, { 214, 0x0003 }, { 216, 0x300c },
01930     { 220, 0xc8c0 }, { 225, 0x0000 }, { 225, 0x8000 }, { 226, 0x0000 },
01931     /* 0x2600 */
01932     { 226, 0x0060 }, { 228, 0x0000 }, { 228, 0x0000 }, { 228, 0x0000 },
01933     { 228, 0x0005 }, { 230, 0x0000 }, { 230, 0xa400 },
01934 };
01935 static const Summary16 jisx0208_uni2indx_page30[16] = {
01936     /* 0x3000 */
01937     { 233, 0xffef }, { 248, 0x103f }, { 255, 0x0000 }, { 255, 0x0000 },
01938     { 255, 0xffff }, { 270, 0xffff }, { 286, 0xffff }, { 302, 0xffff },
01939     { 318, 0xffff }, { 334, 0x780f }, { 342, 0xffff }, { 357, 0xffff },
01940     { 373, 0xffff }, { 389, 0xffff }, { 405, 0xffff }, { 421, 0x787f },
01941 };
01942 static const Summary16 jisx0208_uni2indx_page4e[1307] = {
01943     /* 0x4e00 */
01944     { 432, 0x6f8b }, { 442, 0x43f3 }, { 451, 0x2442 }, { 455, 0x9b46 },
01945     { 463, 0xe82c }, { 470, 0xe3e0 }, { 478, 0x0004 }, { 479, 0x400a },
01946     { 482, 0x5f65 }, { 492, 0xdb36 }, { 502, 0x7977 }, { 513, 0x0449 },
01947     { 517, 0xecd7 }, { 528, 0xe3f0 }, { 537, 0x6038 }, { 542, 0x08c5 },
01948     /* 0x4f00 */
01949     { 547, 0xe602 }, { 553, 0x3403 }, { 558, 0x8000 }, { 559, 0x3551 },
01950     { 566, 0xe0c8 }, { 572, 0x7eab }, { 583, 0x8200 }, { 585, 0x9869 },
01951     { 592, 0xa948 }, { 598, 0x2942 }, { 603, 0xe803 }, { 609, 0x8060 },
01952     { 612, 0x441c }, { 617, 0xad93 }, { 626, 0xc03a }, { 632, 0x4568 },
01953     /* 0x5000 */
01954     { 638, 0xaa60 }, { 644, 0x8656 }, { 651, 0x3f7a }, { 662, 0x0240 },
01955     { 664, 0x8388 }, { 669, 0x1461 }, { 674, 0x1020 }, { 676, 0x2174 },
01956     { 682, 0x2021 }, { 685, 0x0702 }, { 689, 0x3000 }, { 691, 0x40bc },
01957     { 697, 0xa624 }, { 703, 0x4462 }, { 708, 0x60a8 }, { 713, 0x0a20 },
01958     /* 0x5100 */
01959     { 716, 0x0217 }, { 721, 0x8574 }, { 728, 0x0402 }, { 730, 0x9c84 },
01960     { 736, 0x7bfb }, { 749, 0x1415 }, { 754, 0x7f24 }, { 763, 0x11e2 },
01961     { 769, 0xb665 }, { 778, 0x02ef }, { 786, 0x1f75 }, { 796, 0x20ff },
01962     { 805, 0x3a70 }, { 812, 0x3840 }, { 816, 0x26c3 }, { 823, 0x6763 },
01963     /* 0x5200 */
01964     { 832, 0x4dd9 }, { 841, 0x2092 }, { 845, 0x46b0 }, { 851, 0x0fc9 },
01965     { 859, 0xbc98 }, { 867, 0x4850 }, { 871, 0x8638 }, { 877, 0xa03f },
01966     { 885, 0x2388 }, { 890, 0x8816 }, { 895, 0x3e09 }, { 902, 0x5232 },
01967     { 908, 0x22aa }, { 914, 0xe3a4 }, { 922, 0x00dd }, { 928, 0xc72c },
01968     /* 0x5300 */
01969     { 936, 0xa166 }, { 943, 0x26e1 }, { 950, 0x840b }, { 955, 0x8f0a },
01970     { 962, 0x27eb }, { 972, 0x559e }, { 981, 0xc241 }, { 986, 0x89bb },
01971     { 995, 0x0014 }, { 997, 0x8540 }, { 1001, 0x6361 }, { 1008, 0x0849 },
01972     { 1012, 0x7f0c }, { 1021, 0x8ad0 }, { 1027, 0xff3e }, { 1040, 0x05cf },
01973     /* 0x5400 */
```

```

01974 { 1048, 0xff1a }, { 1059, 0xa803 }, { 1064, 0x7a41 }, { 1071, 0x7b40 },
01975 { 1078, 0x4745 }, { 1085, 0x8002 }, { 1087, 0x0500 }, { 1089, 0x38eb },
01976 { 1098, 0xd851 }, { 1105, 0x0005 }, { 1107, 0x9934 }, { 1114, 0x710c },
01977 { 1120, 0x0397 }, { 1127, 0x0100 }, { 1128, 0x6366 }, { 1136, 0x2404 },
01978 /* 0x5500 */
01979 { 1139, 0x80d0 }, { 1143, 0x0051 }, { 1146, 0xc000 }, { 1148, 0x430a },
01980 { 1153, 0x9071 }, { 1159, 0x30c8 }, { 1164, 0x0008 }, { 1165, 0x5800 },
01981 { 1168, 0x0e99 }, { 1175, 0xf700 }, { 1182, 0x5f80 }, { 1189, 0x0041 },
01982 { 1191, 0x00b0 }, { 1194, 0x9410 }, { 1198, 0x0018 }, { 1200, 0x6280 },
01983 /* 0x5600 */
01984 { 1204, 0x0240 }, { 1206, 0x09d0 }, { 1211, 0x8200 }, { 1213, 0x0156 },
01985 { 1218, 0x5004 }, { 1221, 0x0801 }, { 1223, 0x1d10 }, { 1228, 0x0510 },
01986 { 1231, 0x84c1 }, { 1236, 0x0010 }, { 1237, 0x4025 }, { 1241, 0x1050 },
01987 { 1244, 0x410f }, { 1250, 0x4d8a }, { 1257, 0x4009 }, { 1260, 0xa60d },
01988 /* 0x5700 */
01989 { 1267, 0xab19 }, { 1275, 0x914c }, { 1281, 0x21c0 }, { 1285, 0x0981 },
01990 { 1289, 0xc485 }, { 1295, 0x0003 }, { 1297, 0x0652 }, { 1302, 0x8000 },
01991 { 1303, 0x0b04 }, { 1307, 0x0008 }, { 1308, 0x041d }, { 1313, 0x0009 },
01992 { 1315, 0x4849 }, { 1320, 0x905c }, { 1326, 0x0009 }, { 1328, 0x1690 },
01993 /* 0x5800 */
01994 { 1333, 0x0c65 }, { 1339, 0x2220 }, { 1342, 0x8412 }, { 1346, 0x2433 },
01995 { 1352, 0x0c03 }, { 1356, 0x4796 }, { 1364, 0x0a04 }, { 1367, 0x4225 },
01996 { 1372, 0x0028 }, { 1374, 0x9088 }, { 1378, 0x4900 }, { 1381, 0x4f08 },
01997 { 1387, 0x14a2 }, { 1392, 0xd3aa }, { 1401, 0xd830 }, { 1407, 0x3e87 },
01998 /* 0x5900 */
01999 { 1416, 0x8604 }, { 1420, 0x1f61 }, { 1428, 0x7ea4 }, { 1437, 0x4186 },
02000 { 1442, 0xc390 }, { 1448, 0x05b3 }, { 1455, 0x57a5 }, { 1464, 0x2118 },
02001 { 1468, 0x241e }, { 1474, 0x2a48 }, { 1479, 0x1128 }, { 1483, 0x4a04 },
02002 { 1487, 0x0a40 }, { 1490, 0x161b }, { 1497, 0x0d60 }, { 1502, 0x8840 },
02003 /* 0x5a00 */
02004 { 1505, 0x020a }, { 1508, 0x9502 }, { 1513, 0x8221 }, { 1517, 0x1060 },
02005 { 1520, 0x0243 }, { 1524, 0x0400 }, { 1525, 0x1444 }, { 1529, 0x8000 },
02006 { 1530, 0x0000 }, { 1530, 0x0c04 }, { 1533, 0x0000 }, { 1533, 0x7000 },
02007 { 1536, 0x1a06 }, { 1541, 0x00c1 }, { 1544, 0x024a }, { 1548, 0x0c00 },
02008 /* 0x5b00 */
02009 { 1550, 0x1a00 }, { 1553, 0x0040 }, { 1554, 0x1404 }, { 1557, 0x4045 },
02010 { 1561, 0x0029 }, { 1564, 0xbdb3 }, { 1575, 0x0a78 }, { 1581, 0x052b },
02011 { 1587, 0xbba9 }, { 1597, 0xbfa0 }, { 1606, 0x407c }, { 1612, 0x8379 },
02012 { 1620, 0x12fc }, { 1628, 0xe81d }, { 1636, 0x4bf6 }, { 1646, 0xc569 },
02013 /* 0x5c00 */
02014 { 1654, 0xefff6 }, { 1667, 0x044a }, { 1671, 0x2115 }, { 1676, 0xff02 },
02015 { 1685, 0xed63 }, { 1695, 0x402b }, { 1700, 0xd033 }, { 1707, 0x0242 },
02016 { 1710, 0x1000 }, { 1711, 0x0013 }, { 1714, 0x1b02 }, { 1719, 0x59ca },
02017 { 1727, 0x00a0 }, { 1729, 0x0200 }, { 1730, 0xa703 }, { 1737, 0x2c41 },
02018 /* 0x5d00 */
02019 { 1742, 0x4880 }, { 1745, 0x8ff2 }, { 1755, 0x0204 }, { 1757, 0x0000 },
02020 { 1757, 0x5800 }, { 1760, 0x1005 }, { 1763, 0x9200 }, { 1766, 0x0048 },
02021 { 1768, 0x1894 }, { 1773, 0x2001 }, { 1775, 0x5004 }, { 1778, 0x3480 },
02022 { 1782, 0x3200 }, { 1785, 0x684c }, { 1791, 0x49ea }, { 1799, 0x68be },
02023 /* 0x5e00 */
02024 { 1808, 0x184c }, { 1813, 0x2e42 }, { 1819, 0xa820 }, { 1823, 0x21c9 },
02025 { 1829, 0x50b9 }, { 1836, 0x80b0 }, { 1840, 0x001e }, { 1844, 0xff7c },
02026 { 1857, 0x849a }, { 1863, 0x14e0 }, { 1868, 0x28c1 }, { 1873, 0x01e0 },
02027 { 1877, 0x870e }, { 1884, 0xac49 }, { 1891, 0x130f }, { 1898, 0xdddb },
02028 /* 0x5f00 */
02029 { 1910, 0xbel1a }, { 1919, 0x89fb }, { 1929, 0xa2e0 }, { 1935, 0x51a2 },
02030 { 1941, 0x5502 }, { 1946, 0x32ca }, { 1953, 0x3e46 }, { 1961, 0x928b },
02031 { 1968, 0x1dbf }, { 1979, 0x438f }, { 1987, 0x6703 }, { 1994, 0x3218 },
02032 { 1999, 0x3028 }, { 2003, 0x33c0 }, { 2009, 0x0811 }, { 2012, 0xa923 },
02033 /* 0x6000 */
02034 { 2019, 0xc000 }, { 2021, 0x3a65 }, { 2029, 0x8fe3 }, { 2039, 0x0402 },
02035 { 2041, 0x2c4e }, { 2048, 0x8625 }, { 2054, 0xbfd3d }, { 2066, 0x00a1 },
02036 { 2069, 0x3a1a }, { 2076, 0x8cd4 }, { 2083, 0x06c9 }, { 2089, 0x317c },
02037 { 2097, 0x00e0 }, { 2100, 0x950a }, { 2106, 0x018b }, { 2111, 0x0edb },
02038 /* 0x6100 */
02039 { 2120, 0xe34b }, { 2129, 0x8c20 }, { 2133, 0x1182 }, { 2137, 0xf010 },
02040 { 2142, 0x7d94 }, { 2151, 0xa728 }, { 2158, 0xc9ac }, { 2166, 0x40fb },
02041 { 2174, 0x4484 }, { 2178, 0x0653 }, { 2184, 0x5a90 }, { 2190, 0x4444 },
02042 { 2194, 0x3fc8 }, { 2203, 0x0001 }, { 2204, 0x0048 }, { 2206, 0xf5d4 },
02043 /* 0x6200 */
02044 { 2216, 0x7701 }, { 2223, 0xec57 }, { 2233, 0xc442 }, { 2238, 0x891d },
02045 { 2245, 0x6b83 }, { 2253, 0x4928 }, { 2258, 0x4109 }, { 2262, 0xd242 },
02046 { 2268, 0x061d }, { 2274, 0x59fe }, { 2285, 0x1800 }, { 2287, 0x3a22 },
02047 { 2293, 0xb7e4 }, { 2303, 0x3b9f }, { 2314, 0xf003 }, { 2320, 0xc0ea },
02048 /* 0x6300 */
02049 { 2327, 0x1386 }, { 2333, 0x8202 }, { 2336, 0x8980 }, { 2340, 0xe400 },
02050 { 2344, 0xb200 }, { 2348, 0x10a1 }, { 2352, 0x4b80 }, { 2357, 0x0cc4 },
02051 { 2362, 0xd309 }, { 2369, 0x8944 }, { 2374, 0x1faf }, { 2385, 0x4834 },
02052 { 2390, 0x8259 }, { 2396, 0x0c45 }, { 2401, 0x420a }, { 2405, 0x0450 },
02053 /* 0x6400 */
02054 { 2408, 0xa040 }, { 2411, 0x10c8 }, { 2415, 0x3140 }, { 2419, 0x4450 },
02055 { 2423, 0x4004 }, { 2425, 0x0100 }, { 2426, 0x8280 }, { 2429, 0x0540 },
02056 { 2432, 0x0108 }, { 2434, 0x442c }, { 2439, 0x6a30 }, { 2445, 0x1a05 },
02057 { 2450, 0x20a6 }, { 2455, 0x0514 }, { 2459, 0x90cf }, { 2467, 0x6456 },
02058 /* 0x6500 */
02059 { 2474, 0x0021 }, { 2476, 0x3100 }, { 2479, 0x9c18 }, { 2485, 0xcbf0 },
02060 { 2494, 0xa120 }, { 2498, 0x63e2 }, { 2506, 0x104c }, { 2510, 0x01b5 },

```



```
02061 { 2516, 0x538c }, { 2523, 0x9a83 }, { 2530, 0xb8b2 }, { 2538, 0x3281 },
02062 { 2543, 0x987a }, { 2551, 0x0a84 }, { 2555, 0x33e7 }, { 2565, 0x0c02 },
02063 /* 0x6600 */
02064 { 2568, 0xd4cc }, { 2576, 0x9018 }, { 2580, 0xa1a1 }, { 2586, 0x9070 },
02065 { 2591, 0x8a1e }, { 2598, 0xe004 }, { 2602, 0xc3d4 }, { 2610, 0x0451 },
02066 { 2614, 0x439a }, { 2621, 0x21c2 }, { 2626, 0x4844 }, { 2630, 0x5310 },
02067 { 2635, 0x0292 }, { 2639, 0x3640 }, { 2644, 0x0241 }, { 2647, 0xf3bd },
02068 /* 0x6700 */
02069 { 2659, 0xab09 }, { 2666, 0xe8f0 }, { 2674, 0x7dc0 }, { 2682, 0xa5d2 },
02070 { 2690, 0xc242 }, { 2695, 0xd24b }, { 2703, 0xa43f }, { 2712, 0xd0af },
02071 { 2721, 0x1aa0 }, { 2726, 0x34a1 }, { 2732, 0x8247 }, { 2738, 0x03d8 },
02072 { 2744, 0xc452 }, { 2750, 0x651b }, { 2758, 0xd294 }, { 2765, 0xc83a },
02073 /* 0x6800 */
02074 { 2772, 0x001c }, { 2775, 0x40c8 }, { 2779, 0x0e06 }, { 2784, 0x3314 },
02075 { 2790, 0x614f }, { 2798, 0xb21b }, { 2806, 0x0088 }, { 2808, 0xc0d0 },
02076 { 2813, 0xa02a }, { 2818, 0xa898 }, { 2824, 0xalc5 }, { 2831, 0x166b },
02077 { 2839, 0x2e50 }, { 2845, 0x85b4 }, { 2852, 0xc08b }, { 2858, 0x0604 },
02078 /* 0x6900 */
02079 { 2861, 0xf933 }, { 2871, 0x1e04 }, { 2876, 0x056e }, { 2883, 0xa251 },
02080 { 2889, 0x0400 }, { 2890, 0x7638 }, { 2898, 0xec07 }, { 2906, 0x73b8 },
02081 { 2915, 0x4406 }, { 2919, 0x1832 }, { 2924, 0x4081 }, { 2927, 0xc816 },
02082 { 2933, 0x7c8a }, { 2941, 0x6309 }, { 2947, 0x2980 }, { 2951, 0xaa04 },
02083 /* 0x6a00 */
02084 { 2956, 0x1c24 }, { 2961, 0xca9c }, { 2969, 0x4e0e }, { 2976, 0x2760 },
02085 { 2982, 0x0990 }, { 2986, 0x8300 }, { 2989, 0x0046 }, { 2992, 0x8104 },
02086 { 2995, 0x6011 }, { 2999, 0x1081 }, { 3002, 0x540d }, { 3008, 0x0908 },
02087 { 3011, 0x000e }, { 3014, 0xcc0a }, { 3020, 0x0500 }, { 3022, 0x0c00 },
02088 /* 0x6b00 */
02089 { 3024, 0x0430 }, { 3027, 0xa044 }, { 3031, 0x008b }, { 3035, 0x6784 },
02090 { 3042, 0x5288 }, { 3047, 0x8a19 }, { 3053, 0x865e }, { 3061, 0x8b18 },
02091 { 3067, 0x2e59 }, { 3075, 0x4160 }, { 3079, 0x8c10 }, { 3083, 0x9cbe },
02092 { 3093, 0x6861 }, { 3099, 0x891c }, { 3105, 0x9800 }, { 3108, 0x0008 },
02093 /* 0x6c00 */
02094 { 3109, 0x8100 }, { 3111, 0x089a }, { 3116, 0x0018 }, { 3118, 0x4190 },
02095 { 3122, 0x4007 }, { 3126, 0xe4a1 }, { 3133, 0x0505 }, { 3137, 0x640d },
02096 { 3143, 0x310e }, { 3149, 0x0e4d }, { 3156, 0x4806 }, { 3160, 0xff0a },
02097 { 3170, 0x1632 }, { 3176, 0x2aa8 }, { 3182, 0x852e }, { 3189, 0x000b },
02098 /* 0x6d00 */
02099 { 3192, 0x1800 }, { 3194, 0xca84 }, { 3200, 0x0e20 }, { 3204, 0x696c },
02100 { 3212, 0x0032 }, { 3215, 0x1600 }, { 3218, 0x5658 }, { 3225, 0x0390 },
02101 { 3229, 0x5120 }, { 3233, 0x1a28 }, { 3238, 0x8000 }, { 3239, 0x1124 },
02102 { 3243, 0x18e1 }, { 3249, 0x4326 }, { 3255, 0x5d52 }, { 3263, 0x0eaa },
02103 /* 0x6e00 */
02104 { 3270, 0x0fa0 }, { 3276, 0xae28 }, { 3283, 0xfa7b }, { 3295, 0x4500 },
02105 { 3298, 0x6408 }, { 3302, 0x8940 }, { 3306, 0xc880 }, { 3310, 0xc044 },
02106 { 3314, 0x9005 }, { 3318, 0xb141 }, { 3324, 0x8424 }, { 3328, 0x24c4 },
02107 { 3333, 0x1a34 }, { 3339, 0x603a }, { 3345, 0x9000 }, { 3347, 0xc194 },
02108 /* 0x6f00 */
02109 { 3353, 0x8246 }, { 3358, 0x003a }, { 3362, 0x180d }, { 3367, 0xc106 },
02110 { 3372, 0x0022 }, { 3374, 0x9910 }, { 3379, 0xe050 }, { 3384, 0x1511 },
02111 { 3389, 0x4057 }, { 3395, 0x0082 }, { 3397, 0x041a }, { 3401, 0x020a },
02112 { 3404, 0x004f }, { 3409, 0x8930 }, { 3414, 0xd813 }, { 3421, 0x444a },
02113 /* 0x7000 */
02114 { 3426, 0x8a02 }, { 3430, 0xed22 }, { 3438, 0x10c0 }, { 3441, 0x4005 },
02115 { 3444, 0x1000 }, { 3445, 0x0102 }, { 3447, 0x8808 }, { 3450, 0x3101 },
02116 { 3454, 0x4600 }, { 3457, 0x0204 }, { 3459, 0xf000 }, { 3463, 0x0708 },
02117 { 3467, 0x8900 }, { 3470, 0xa200 }, { 3473, 0x0000 }, { 3473, 0x2202 },
02118 /* 0x7100 */
02119 { 3476, 0x0200 }, { 3477, 0x1610 }, { 3481, 0x0042 }, { 3483, 0x1040 },
02120 { 3485, 0x5200 }, { 3488, 0x0260 }, { 3491, 0x52f4 }, { 3499, 0x2000 },
02121 { 3500, 0x8510 }, { 3504, 0x8230 }, { 3508, 0x1100 }, { 3510, 0x4202 },
02122 { 3513, 0x4308 }, { 3517, 0x80b5 }, { 3523, 0x70e1 }, { 3530, 0x9a20 },
02123 /* 0x7200 */
02124 { 3535, 0x2040 }, { 3537, 0x0801 }, { 3539, 0x3500 }, { 3543, 0xfc65 },
02125 { 3553, 0x19c1 }, { 3559, 0xab04 }, { 3565, 0x0286 }, { 3569, 0x6214 },
02126 { 3574, 0x0087 }, { 3578, 0x0044 }, { 3580, 0x9085 }, { 3585, 0x0244 },
02127 { 3588, 0x405c }, { 3593, 0x0a85 }, { 3598, 0x3207 }, { 3604, 0x3380 },
02128 /* 0x7300 */
02129 { 3609, 0x0400 }, { 3610, 0xb8c0 }, { 3616, 0xce20 }, { 3622, 0xc0d0 },
02130 { 3627, 0xc030 }, { 3631, 0x0080 }, { 3632, 0x0508 }, { 3635, 0x0d25 },
02131 { 3641, 0x0a90 }, { 3645, 0x0040 }, { 3646, 0x0200 }, { 3647, 0x080c },
02132 { 3650, 0x6505 }, { 3656, 0x4000 }, { 3657, 0x6421 }, { 3662, 0x4102 },
02133 /* 0x7400 */
02134 { 3665, 0x0268 }, { 3669, 0x0000 }, { 3669, 0x0024 }, { 3671, 0x847c },
02135 { 3678, 0x0002 }, { 3679, 0xde20 }, { 3686, 0x8619 }, { 3692, 0x4049 },
02136 { 3696, 0x0808 }, { 3698, 0x4000 }, { 3699, 0x0084 }, { 3701, 0x2001 },
02137 { 3703, 0x8400 }, { 3705, 0x1010 }, { 3707, 0x42cd }, { 3714, 0x01c7 },
02138 /* 0x7500 */
02139 { 3720, 0x7038 }, { 3726, 0xd52a }, { 3734, 0x1968 }, { 3740, 0x1d8f },
02140 { 3749, 0xbe50 }, { 3757, 0x3e12 }, { 3764, 0x2ef5 }, { 3774, 0x81d9 },
02141 { 3781, 0xcec4 }, { 3789, 0x2412 }, { 3793, 0x0828 }, { 3796, 0x732e },
02142 { 3805, 0x224c }, { 3811, 0x4b34 }, { 3818, 0x020c }, { 3821, 0xd41d },
02143 /* 0x7600 */
02144 { 3829, 0x2a02 }, { 3833, 0x8000 }, { 3834, 0x0097 }, { 3839, 0x0811 },
02145 { 3842, 0x11c4 }, { 3847, 0x1144 }, { 3851, 0x1786 }, { 3858, 0x7d45 },
02146 { 3867, 0x49d9 }, { 3875, 0x0649 }, { 3880, 0x4000 }, { 3881, 0x8791 },
02147 { 3888, 0x254c }, { 3894, 0xd8c4 }, { 3901, 0x44ba }, { 3908, 0x4914 },
```

```

02148  /* 0x7700 */
02149  { 3913, 0x1b92 }, { 3920, 0xc800 }, { 3923, 0x0271 }, { 3928, 0x1580 },
02150  { 3932, 0x0081 }, { 3934, 0x0c00 }, { 3936, 0x096a }, { 3942, 0xc200 },
02151  { 3945, 0x4800 }, { 3947, 0x4002 }, { 3949, 0x3021 }, { 3953, 0xba49 },
02152  { 3961, 0x2080 }, { 3963, 0x1c80 }, { 3967, 0xe2ac }, { 3975, 0x1008 },
02153  /* 0x7800 */
02154  { 3977, 0x1004 }, { 3979, 0x0034 }, { 3982, 0x00e1 }, { 3986, 0x8414 },
02155  { 3990, 0x0020 }, { 3991, 0x2000 }, { 3992, 0x9800 }, { 3995, 0x1014 },
02156  { 3998, 0x70c2 }, { 4004, 0x04aa }, { 4009, 0x8688 }, { 4014, 0x5420 },
02157  { 4018, 0x0c62 }, { 4023, 0x0413 }, { 4027, 0x9180 }, { 4031, 0x2010 },
02158  /* 0x7900 */
02159  { 4033, 0x4082 }, { 4036, 0x0206 }, { 4039, 0x1c40 }, { 4043, 0x5400 },
02160  { 4046, 0x0383 }, { 4051, 0xe4e9 }, { 4060, 0x2125 }, { 4065, 0x8480 },
02161  { 4068, 0xe433 }, { 4076, 0x2000 }, { 4077, 0x44c0 }, { 4081, 0xe609 },
02162  { 4088, 0x0a03 }, { 4092, 0x8126 }, { 4097, 0x12da }, { 4104, 0x0801 },
02163  /* 0x7a00 */
02164  { 4106, 0x6901 }, { 4111, 0x9790 }, { 4118, 0x4001 }, { 4120, 0xf886 },
02165  { 4128, 0xe24d }, { 4136, 0x0081 }, { 4138, 0x0a0e }, { 4143, 0xa651 },
02166  { 4150, 0x011a }, { 4154, 0x81ec }, { 4161, 0xc600 }, { 4165, 0x8441 },
02167  { 4169, 0xadb8 }, { 4178, 0xb62c }, { 4186, 0xa46f }, { 4195, 0x8741 },
02168  /* 0x7b00 */
02169  { 4201, 0x8d54 }, { 4208, 0x4b02 }, { 4213, 0x1161 }, { 4218, 0x0268 },
02170  { 4222, 0xbb60 }, { 4230, 0x2057 }, { 4236, 0x50a0 }, { 4240, 0x0433 },
02171  { 4245, 0xa8c0 }, { 4250, 0xb7b4 }, { 4260, 0x2402 }, { 4263, 0x0112 },
02172  { 4266, 0x9ad3 }, { 4275, 0x2000 }, { 4276, 0x2271 }, { 4282, 0x00c8 },
02173  /* 0x7c00 */
02174  { 4285, 0x2081 }, { 4288, 0x809e }, { 4294, 0x0c8a }, { 4299, 0xe180 },
02175  { 4304, 0xb009 }, { 4309, 0x8151 }, { 4314, 0x1031 }, { 4318, 0x4028 },
02176  { 4321, 0x2a0e }, { 4327, 0x89a5 }, { 4334, 0x69b6 }, { 4343, 0x620e },
02177  { 4349, 0x4425 }, { 4354, 0xd144 }, { 4360, 0x8085 }, { 4364, 0x4d54 },
02178  /* 0x7d00 */
02179  { 4371, 0x2c75 }, { 4379, 0x1fb1 }, { 4388, 0xd807 }, { 4395, 0x862d },
02180  { 4402, 0xd87c }, { 4411, 0x4841 }, { 4415, 0x414e }, { 4421, 0x226e },
02181  { 4428, 0x8200 }, { 4430, 0x9e08 }, { 4436, 0xf80c }, { 4443, 0xed37 },
02182  { 4454, 0x8c80 }, { 4458, 0x7526 }, { 4466, 0x9313 }, { 4473, 0x0814 },
02183  /* 0x7e00 */
02184  { 4476, 0x0e32 }, { 4482, 0xc804 }, { 4486, 0x484e }, { 4492, 0x6ea6 },
02185  { 4501, 0x2c4a }, { 4507, 0x6670 }, { 4514, 0x26c0 }, { 4519, 0xba01 },
02186  { 4525, 0xd30c }, { 4532, 0x185d }, { 4539, 0x0000 }, { 4539, 0x0000 },
02187  { 4539, 0x0000 }, { 4539, 0x0000 }, { 4539, 0x0000 }, { 4539, 0x0000 },
02188  /* 0x7f00 */
02189  { 4539, 0x0000 }, { 4539, 0x0000 }, { 4539, 0x0000 }, { 4539, 0x0540 },
02190  { 4542, 0x7020 }, { 4546, 0x8133 }, { 4552, 0x4f81 }, { 4559, 0x03a5 },
02191  { 4565, 0x55ec }, { 4574, 0x6410 }, { 4578, 0xc318 }, { 4584, 0x2344 },
02192  { 4589, 0x1462 }, { 4594, 0x0034 }, { 4597, 0x0a43 }, { 4602, 0x1a09 },
02193  /* 0x8000 */
02194  { 4607, 0x187b }, { 4615, 0x13a5 }, { 4622, 0x0102 }, { 4624, 0xa848 },
02195  { 4629, 0x0440 }, { 4631, 0xc544 }, { 4637, 0x8106 }, { 4641, 0xe2dd },
02196  { 4651, 0x1af0 }, { 4658, 0x2d48 }, { 4664, 0xb626 }, { 4672, 0x0416 },
02197  { 4676, 0x5058 }, { 4681, 0x6e40 }, { 4687, 0x8032 }, { 4691, 0x3112 },
02198  /* 0x8100 */
02199  { 4696, 0x07e4 }, { 4703, 0x0c00 }, { 4705, 0x8208 }, { 4708, 0x420a },
02200  { 4712, 0x4840 }, { 4715, 0x803b }, { 4721, 0x4860 }, { 4725, 0x8713 },
02201  { 4732, 0x850d }, { 4738, 0x3428 }, { 4743, 0x0319 }, { 4748, 0xe529 },
02202  { 4756, 0x2345 }, { 4762, 0x870a }, { 4768, 0x25a9 }, { 4775, 0x5c18 },
02203  /* 0x8200 */
02204  { 4781, 0x77a6 }, { 4791, 0xd9c5 }, { 4800, 0x5e00 }, { 4805, 0x03e8 },
02205  { 4811, 0x0081 }, { 4813, 0xa700 }, { 4818, 0xcd54 }, { 4826, 0x41c6 },
02206  { 4832, 0x2800 }, { 4834, 0xa204 }, { 4838, 0xb860 }, { 4844, 0x2b0a },
02207  { 4850, 0x0020 }, { 4851, 0xda9e }, { 4861, 0x08ea }, { 4867, 0x0e1a },
02208  /* 0x8300 */
02209  { 4873, 0x427c }, { 4880, 0x11c0 }, { 4884, 0x8908 }, { 4888, 0x0376 },
02210  { 4895, 0x8621 }, { 4900, 0x0105 }, { 4903, 0x0000 }, { 4903, 0x18a8 },
02211  { 4908, 0x46a0 }, { 4913, 0xc448 }, { 4918, 0x0d05 }, { 4923, 0x2022 },
02212  { 4926, 0x5422 }, { 4931, 0x9148 }, { 4936, 0x8a01 }, { 4940, 0x2897 },
02213  /* 0x8400 */
02214  { 4947, 0x7898 }, { 4954, 0x0008 }, { 4955, 0x1605 }, { 4960, 0x3122 },
02215  { 4965, 0x4240 }, { 4968, 0x0880 }, { 4970, 0xfa4e }, { 4980, 0x06a2 },
02216  { 4985, 0x0814 }, { 4988, 0x9211 }, { 4993, 0x2002 }, { 4995, 0x9b04 },
02217  { 5001, 0x2e52 }, { 5008, 0x0643 }, { 5013, 0x5000 }, { 5015, 0x9010 },
02218  /* 0x8500 */
02219  { 5018, 0x0041 }, { 5020, 0x85ba }, { 5028, 0x3042 }, { 5032, 0x2020 },
02220  { 5034, 0x4f0b }, { 5042, 0x05a0 }, { 5046, 0x2708 }, { 5051, 0x4080 },
02221  { 5053, 0x0591 }, { 5058, 0x1a93 }, { 5065, 0xdf50 }, { 5074, 0x0600 },
02222  { 5076, 0xa202 }, { 5080, 0x3021 }, { 5084, 0x0630 }, { 5088, 0x4e80 },
02223  /* 0x8600 */
02224  { 5093, 0x0cc4 }, { 5098, 0x04c8 }, { 5102, 0xa004 }, { 5105, 0x8001 },
02225  { 5107, 0x6000 }, { 5109, 0xd431 }, { 5116, 0x0880 }, { 5118, 0x0a02 },
02226  { 5121, 0x1c00 }, { 5124, 0x0028 }, { 5126, 0x8e18 }, { 5132, 0x0041 },
02227  { 5134, 0x6ad0 }, { 5141, 0xca10 }, { 5146, 0xf210 }, { 5152, 0x4b00 },
02228  /* 0x8700 */
02229  { 5156, 0x274d }, { 5164, 0x1506 }, { 5169, 0x0220 }, { 5171, 0x8890 },
02230  { 5175, 0x5a00 }, { 5179, 0x82a8 }, { 5184, 0x4549 }, { 5190, 0x8150 },
02231  { 5194, 0x2004 }, { 5196, 0x8000 }, { 5197, 0x8804 }, { 5200, 0x2c08 },
02232  { 5204, 0x08d1 }, { 5209, 0x0005 }, { 5211, 0x8001 }, { 5213, 0x4ac4 },
02233  /* 0x8800 */
02234  { 5219, 0xe020 }, { 5223, 0x0062 }, { 5226, 0x008e }, { 5230, 0x0a42 },

```

```
02235 { 5234, 0x3055 }, { 5240, 0x6a8c }, { 5247, 0x090e }, { 5252, 0xe0a5 },
02236 { 5259, 0x2906 }, { 5264, 0x42c4 }, { 5269, 0x4814 }, { 5273, 0x80b3 },
02237 { 5279, 0x803e }, { 5285, 0xb330 }, { 5292, 0x0102 }, { 5294, 0x731c },
02238 /* 0x8900 */
02239 { 5302, 0x1494 }, { 5307, 0x600d }, { 5312, 0x0c20 }, { 5315, 0x0940 },
02240 { 5318, 0x301a }, { 5323, 0xc040 }, { 5326, 0xa451 }, { 5332, 0xc094 },
02241 { 5337, 0x8dca }, { 5345, 0x05c8 }, { 5350, 0x96c2 }, { 5357, 0xa40c },
02242 { 5362, 0x0001 }, { 5363, 0x3404 }, { 5367, 0x00c8 }, { 5370, 0x0110 },
02243 /* 0x8a00 */
02244 { 5372, 0x550d }, { 5379, 0xa9c9 }, { 5387, 0x2428 }, { 5391, 0x1c5a },
02245 { 5398, 0x0142 }, { 5401, 0x4837 }, { 5408, 0x7a4d }, { 5417, 0x100f },
02246 { 5422, 0x32b4 }, { 5429, 0x452a }, { 5435, 0x317b }, { 5444, 0x9205 },
02247 { 5449, 0xb894 }, { 5456, 0x5c44 }, { 5462, 0x68d7 }, { 5471, 0x458a },
02248 /* 0x8b00 */
02249 { 5477, 0x5097 }, { 5484, 0x2ed1 }, { 5492, 0x1943 }, { 5498, 0x4208 },
02250 { 5501, 0xd202 }, { 5506, 0x9d40 }, { 5512, 0x9840 }, { 5516, 0x2097 },
02251 { 5522, 0x5409 }, { 5527, 0x064d }, { 5533, 0x0000 }, { 5533, 0x0000 },
02252 { 5533, 0x0000 }, { 5533, 0x0000 }, { 5533, 0x0000 }, { 5533, 0x0000 },
02253 /* 0x8c00 */
02254 { 5533, 0x0000 }, { 5533, 0x0000 }, { 5533, 0x0000 }, { 5533, 0x8480 },
02255 { 5536, 0x5542 }, { 5542, 0x0421 }, { 5545, 0x1c06 }, { 5550, 0x1700 },
02256 { 5554, 0x7624 }, { 5561, 0x6110 }, { 5565, 0xff87 }, { 5577, 0xb9dd },
02257 { 5588, 0x659f }, { 5598, 0x5c0a }, { 5604, 0x245d }, { 5611, 0x3c00 },
02258 /* 0x8d00 */
02259 { 5615, 0xadb0 }, { 5623, 0x0059 }, { 5627, 0x0000 }, { 5627, 0x0000 },
02260 { 5627, 0x0000 }, { 5627, 0x0000 }, { 5627, 0x28d0 }, { 5632, 0x009b },
02261 { 5637, 0x0422 }, { 5640, 0x0200 }, { 5641, 0x0108 }, { 5643, 0x4408 },
02262 { 5646, 0x9804 }, { 5650, 0xac40 }, { 5655, 0x8d0a }, { 5661, 0x9028 },
02263 /* 0x8e00 */
02264 { 5665, 0x8700 }, { 5669, 0xe001 }, { 5673, 0x0400 }, { 5674, 0x0031 },
02265 { 5677, 0x1794 }, { 5684, 0x8221 }, { 5688, 0x0019 }, { 5691, 0x1054 },
02266 { 5695, 0x2cb2 }, { 5702, 0x021a }, { 5706, 0x9c02 }, { 5711, 0x4003 },
02267 { 5714, 0x3d60 }, { 5721, 0x8804 }, { 5724, 0x080c }, { 5727, 0x7900 },
02268 /* 0x8f00 */
02269 { 5732, 0x1628 }, { 5737, 0xba3c }, { 5746, 0x8640 }, { 5750, 0xcb08 },
02270 { 5756, 0x7274 }, { 5764, 0x9080 }, { 5767, 0x001e }, { 5771, 0x0000 },
02271 { 5771, 0x0000 }, { 5771, 0xd800 }, { 5775, 0xe188 }, { 5781, 0x9c87 },
02272 { 5789, 0x4034 }, { 5793, 0x0412 }, { 5796, 0xae64 }, { 5804, 0x2791 },
02273 /* 0x9000 */
02274 { 5811, 0xe86b }, { 5820, 0xe6fb }, { 5832, 0x408f }, { 5838, 0x5366 },
02275 { 5846, 0xeea6 }, { 5856, 0x537f }, { 5867, 0xe32b }, { 5876, 0xb5e4 },
02276 { 5885, 0x869f }, { 5894, 0x0002 }, { 5895, 0x8548 }, { 5900, 0x0122 },
02277 { 5903, 0x4402 }, { 5906, 0x0800 }, { 5907, 0x2116 }, { 5912, 0x20a0 },
02278 /* 0x9100 */
02279 { 5915, 0x0004 }, { 5916, 0x0204 }, { 5918, 0x2000 }, { 5919, 0x0005 },
02280 { 5921, 0x7e00 }, { 5927, 0x0154 }, { 5931, 0x162c }, { 5937, 0x01ac },
02281 { 5942, 0x2a84 }, { 5947, 0x1085 }, { 5951, 0x8c14 }, { 5956, 0x0530 },
02282 { 5960, 0xfbc3 }, { 5971, 0xb943 }, { 5979, 0x00ca }, { 5983, 0x9060 },
02283 /* 0x9200 */
02284 { 5987, 0x6000 }, { 5989, 0x4032 }, { 5993, 0x1200 }, { 5995, 0x8090 },
02285 { 5998, 0x0b30 }, { 6003, 0x4c81 }, { 6008, 0x0054 }, { 6011, 0x4002 },
02286 { 6013, 0x0029 }, { 6016, 0x1d6a }, { 6024, 0x2000 }, { 6025, 0x0280 },
02287 { 6027, 0x8000 }, { 6028, 0x0004 }, { 6029, 0x2610 }, { 6033, 0x150c },
02288 /* 0x9300 */
02289 { 6038, 0x8040 }, { 6040, 0x0701 }, { 6044, 0xd94d }, { 6053, 0x0c24 },
02290 { 6057, 0x2810 }, { 6060, 0x1850 }, { 6064, 0x5001 }, { 6067, 0x5020 },
02291 { 6070, 0x1000 }, { 6071, 0x04d0 }, { 6075, 0x7080 }, { 6079, 0x0201 },
02292 { 6081, 0x0108 }, { 6083, 0x21c3 }, { 6089, 0x0132 }, { 6093, 0x0000 },
02293 /* 0x9400 */
02294 { 6093, 0x0088 }, { 6095, 0x0719 }, { 6101, 0x0802 }, { 6103, 0x0560 },
02295 { 6107, 0x0012 }, { 6109, 0x4c0e }, { 6115, 0x0405 }, { 6118, 0xf0a1 },
02296 { 6125, 0x0002 }, { 6126, 0x0000 }, { 6126, 0x0000 }, { 6126, 0x0000 },
02297 { 6126, 0x0000 }, { 6126, 0x0000 }, { 6126, 0x0000 }, { 6126, 0x0000 },
02298 /* 0x9500 */
02299 { 6126, 0x0000 }, { 6126, 0x0000 }, { 6126, 0x0000 }, { 6126, 0x0000 },
02300 { 6126, 0x0000 }, { 6126, 0x0000 }, { 6126, 0x0000 }, { 6126, 0x0080 },
02301 { 6127, 0x8e8d }, { 6135, 0x035a }, { 6141, 0x21bd }, { 6149, 0x5a04 },
02302 { 6154, 0x3488 }, { 6159, 0x1170 }, { 6164, 0x0026 }, { 6167, 0x0000 },
02303 /* 0x9600 */
02304 { 6167, 0x0000 }, { 6167, 0x1000 }, { 6168, 0xc502 }, { 6173, 0x8804 },
02305 { 6176, 0xb815 }, { 6183, 0xf801 }, { 6189, 0x147c }, { 6196, 0x25ed },
02306 { 6205, 0xed60 }, { 6213, 0x1bb0 }, { 6220, 0x0589 }, { 6225, 0x1bd7 },
02307 { 6235, 0x7af3 }, { 6246, 0x1a62 }, { 6252, 0x0d0c }, { 6257, 0x0ac5 },
02308 /* 0x9700 */
02309 { 6263, 0xe5d1 }, { 6272, 0x524a }, { 6278, 0x0490 }, { 6281, 0x6305 },
02310 { 6287, 0x0354 }, { 6292, 0x5244 }, { 6297, 0x2b57 }, { 6306, 0x1612 },
02311 { 6311, 0xa872 }, { 6318, 0x1101 }, { 6321, 0x2949 }, { 6327, 0x0018 },
02312 { 6329, 0x0948 }, { 6333, 0x1008 }, { 6335, 0x6000 }, { 6337, 0x886c },
02313 /* 0x9800 */
02314 { 6343, 0x916e }, { 6351, 0x058f }, { 6358, 0x3012 }, { 6362, 0x3990 },
02315 { 6368, 0xf840 }, { 6374, 0x4930 }, { 6379, 0x8880 }, { 6382, 0x001b },
02316 { 6386, 0x0000 }, { 6386, 0x0000 }, { 6386, 0x8500 }, { 6389, 0x0042 },
02317 { 6391, 0x0058 }, { 6394, 0x9800 }, { 6397, 0xea04 }, { 6403, 0x7014 },
02318 /* 0x9900 */
02319 { 6408, 0x1628 }, { 6413, 0x611d }, { 6420, 0x5113 }, { 6426, 0x6000 },
02320 { 6428, 0x1a2a }, { 6433, 0x00a7 }, { 6438, 0x0000 }, { 6438, 0x0000 },
02321 { 6438, 0x0000 }, { 6438, 0x03c0 }, { 6442, 0x7120 }, { 6447, 0x1018 },
```

```

02322 { 6450, 0x0172 }, { 6455, 0xa927 }, { 6463, 0x6004 }, { 6466, 0x8906 },
02323 /* 0x9a00 */
02324 { 6471, 0xc022 }, { 6475, 0x020c }, { 6478, 0x0900 }, { 6480, 0x4081 },
02325 { 6483, 0x202d }, { 6488, 0x8ca0 }, { 6493, 0x0e34 }, { 6499, 0x0000 },
02326 { 6499, 0x0000 }, { 6499, 0x0000 }, { 6499, 0x2100 }, { 6501, 0x1101 },
02327 { 6504, 0x8011 }, { 6507, 0xc11a }, { 6513, 0xec4c }, { 6521, 0x0892 },
02328 /* 0x9b00 */
02329 { 6525, 0x0040 }, { 6526, 0x8500 }, { 6529, 0xc7ac }, { 6538, 0x1806 },
02330 { 6542, 0xe03e }, { 6550, 0x0512 }, { 6554, 0x8000 }, { 6555, 0x0010 },
02331 { 6556, 0x4008 }, { 6558, 0x80ce }, { 6564, 0x6d01 }, { 6570, 0x0210 },
02332 { 6572, 0x8641 }, { 6577, 0x0856 }, { 6582, 0x011e }, { 6587, 0x0027 },
02333 /* 0x9c00 */
02334 { 6591, 0x3750 }, { 6598, 0x083d }, { 6604, 0xe032 }, { 6610, 0x4e05 },
02335 { 6616, 0x01c0 }, { 6619, 0x0484 }, { 6622, 0x0081 }, { 6624, 0x0140 },
02336 { 6626, 0x0000 }, { 6626, 0x0000 }, { 6626, 0x0000 }, { 6626, 0x0000 },
02337 { 6626, 0x0000 }, { 6626, 0x0000 }, { 6626, 0x1aa0 }, { 6631, 0x0059 },
02338 /* 0x9d00 */
02339 { 6635, 0x43c8 }, { 6641, 0x8824 }, { 6645, 0x1d48 }, { 6651, 0xc800 },
02340 { 6654, 0x0152 }, { 6658, 0x7203 }, { 6664, 0x9013 }, { 6669, 0x0404 },
02341 { 6671, 0x8280 }, { 6674, 0x0400 }, { 6675, 0x8a10 }, { 6679, 0x0d14 },
02342 { 6684, 0x8056 }, { 6689, 0x0208 }, { 6691, 0xa040 }, { 6694, 0x2704 },
02343 /* 0x9e00 */
02344 { 6699, 0x0000 }, { 6699, 0x4c00 }, { 6702, 0x0000 }, { 6702, 0x0000 },
02345 { 6702, 0x0000 }, { 6702, 0x0000 }, { 6702, 0x0000 }, { 6702, 0xa320 },
02346 { 6707, 0x1902 }, { 6711, 0xa0ae }, { 6718, 0x2660 }, { 6723, 0xdf00 },
02347 { 6730, 0xf010 }, { 6735, 0x7b15 }, { 6744, 0x8121 }, { 6748, 0x3ad0 },
02348 /* 0x9f00 */
02349 { 6755, 0x4180 }, { 6758, 0x0028 }, { 6760, 0x1003 }, { 6763, 0x4800 },
02350 { 6765, 0xcc00 }, { 6769, 0x8014 }, { 6772, 0x14cf }, { 6780, 0x00c4 },
02351 { 6783, 0x2000 }, { 6784, 0x3020 }, { 6787, 0x0001 },
02352 };
02353 static const Summary16 jisx0208_uni2indx_pageff[15] = {
02354 /* 0xff00 */
02355 { 6788, 0xdf7a }, { 6800, 0xffff }, { 6816, 0xffff }, { 6832, 0xffff },
02356 { 6847, 0xffff }, { 6863, 0x3fff }, { 6877, 0x0000 }, { 6877, 0x0000 },
02357 { 6877, 0x0000 }, { 6877, 0x0000 }, { 6877, 0x0000 }, { 6877, 0x0000 },
02358 { 6877, 0x0000 }, { 6877, 0x0000 }, { 6877, 0x0028 },
02359 };
02360
02361 static int
02362 jisx0208_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
02363 {
02364     (void)conv;
02365     if (n >= 2) {
02366         const Summary16 *summary = NULL;
02367         if (wc < 0x0100)
02368             summary = &jisx0208_uni2indx_page00[(wc>4)];
02369         else if (wc >= 0x0300 && wc < 0x0460)
02370             summary = &jisx0208_uni2indx_page03[(wc>4)-0x030];
02371         else if (wc >= 0x2000 && wc < 0x2320)
02372             summary = &jisx0208_uni2indx_page20[(wc>4)-0x200];
02373         else if (wc >= 0x2500 && wc < 0x2670)
02374             summary = &jisx0208_uni2indx_page25[(wc>4)-0x250];
02375         else if (wc >= 0x3000 && wc < 0x3100)
02376             summary = &jisx0208_uni2indx_page30[(wc>4)-0x300];
02377         else if (wc >= 0x4e00 && wc < 0x9fb0)
02378             summary = &jisx0208_uni2indx_page4e[(wc>4)-0x4e0];
02379         else if (wc >= 0xff00 && wc < 0xffff)
02380             summary = &jisx0208_uni2indx_pageff[(wc>4)-0xff0];
02381         if (summary) {
02382             unsigned short used = summary->used;
02383             unsigned int i = wc & 0x0f;
02384             if (used & ((unsigned short) 1 << i)) {
02385                 unsigned short c;
02386                 /* Keep in 'used' only the bits 0..i-1. */
02387                 used &= ((unsigned short) 1 << i) - 1;
02388                 /* Add 'summary->indx' and the number of bits set in 'used'. */
02389                 used = (used & 0x5555) + ((used & 0xaaaa) >> 1);
02390                 used = (used & 0x3333) + ((used & 0xcccc) >> 2);
02391                 used = (used & 0x0f0f) + ((used & 0xf0f0) >> 4);
02392                 used = (used & 0x00ff) + (used >> 8);
02393                 c = jisx0208_2charset[summary->indx + used];
02394                 r[0] = (c >> 8); r[1] = (c & 0xff);
02395                 return 2;
02396             }
02397         }
02398         return RET_ILSEQ;
02399     }
02400     return RET_TOOSMALL;
02401 }
02402 #endif /* NEED_TOMB */

```

12.287 jisx0212.h

```
00001 /* $XFree86: xc/lib/X11/lcUniConv/jisx0212.h,v 1.5 2003/05/27 22:26:31 tsi Exp $ */
```

```
00002
00003 /*
00004  * JISX0212.1990-0
00005  */
00006 #ifndef NEED_TOWC
00007
00008 static const unsigned short jisx0212_2uni_page22[81] = {
00009     /* 0x22 */
00010     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00011     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x02d8, 0x02c7,
00012     0x00b8, 0x02d9, 0x02dd, 0x00af, 0x02db, 0x02da, 0x007e, 0x0384,
00013     0x0385, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00014     0xffff, 0x00a1, 0x00a6, 0x00bf, 0xffff, 0xffff, 0xffff, 0xffff,
00015     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00016     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00017     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00018     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00019     0xffff, 0xffff, 0x00ba, 0x00aa, 0x00a9, 0x00ae, 0x2122, 0x00a4,
00020     0x2116,
00021 };
00022 static const unsigned short jisx0212_2uni_page26[188] = {
00023     /* 0x26 */
00024     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00025     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00026     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00027     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00028     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00029     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00030     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00031     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00032     0x0386, 0x0388, 0x0389, 0x038a, 0x03aa, 0xffff, 0x038c, 0xffff,
00033     0x038e, 0x03ab, 0xffff, 0x038f, 0xffff, 0xffff, 0xffff, 0xffff,
00034     0x03ac, 0x03ad, 0x03ae, 0x03af, 0x03ca, 0x0390, 0x03cc, 0x03c2,
00035     0x03cd, 0x03cb, 0x03b0, 0x03ce, 0xffff, 0xffff,
00036     /* 0x27 */
00037     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00038     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00039     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00040     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00041     0xffff, 0x0402, 0x0403, 0x0404, 0x0405, 0x0406, 0x0407, 0x0408,
00042     0x0409, 0x040a, 0x040b, 0x040c, 0x040e, 0x040f, 0xffff, 0xffff,
00043     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00044     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00045     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00046     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00047     0xffff, 0x0452, 0x0453, 0x0454, 0x0455, 0x0456, 0x0457, 0x0458,
00048     0x0459, 0x045a, 0x045b, 0x045c, 0x045e, 0x045f,
00049 };
00050 static const unsigned short jisx0212_2uni_page29[275] = {
00051     /* 0x29 */
00052     0x00c6, 0x0110, 0xffff, 0x0126, 0xffff, 0x0132, 0xffff, 0x0141,
00053     0x013f, 0xffff, 0x014a, 0x00d8, 0x0152, 0xffff, 0x0166, 0x00de,
00054     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00055     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00056     0x00e6, 0x0111, 0x00f0, 0x0127, 0x0131, 0x0133, 0x0138, 0x0142,
00057     0x0140, 0x0149, 0x014b, 0x00f8, 0x0153, 0x00df, 0x0167, 0x00fe,
00058     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00059     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00060     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00061     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00062     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00063     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00064     /* 0x2a */
00065     0x00c1, 0x00c0, 0x00c4, 0x00c2, 0x0102, 0x01cd, 0x0100, 0x0104,
00066     0x00c5, 0x00c3, 0x0106, 0x0108, 0x010c, 0x00c7, 0x010a, 0x010e,
00067     0x00c9, 0x00c8, 0x00cb, 0x00ca, 0x011a, 0x0116, 0x0112, 0x0118,
00068     0xffff, 0x011c, 0x011e, 0x0122, 0x0120, 0x0124, 0x00cd, 0x00cc,
00069     0x00cf, 0x00ce, 0x01cf, 0x0130, 0x012a, 0x012e, 0x0128, 0x0134,
00070     0x0136, 0x0139, 0x013d, 0x013b, 0x0143, 0x0147, 0x0145, 0x00d1,
00071     0x00d3, 0x00d2, 0x00d6, 0x00d4, 0x01d1, 0x0150, 0x014c, 0x00d5,
00072     0x0154, 0x0158, 0x0156, 0x015c, 0x015a, 0x015e, 0x0164,
00073     0x0162, 0x00da, 0x00d9, 0x00dc, 0x00db, 0x016c, 0x01d3, 0x0170,
00074     0x016a, 0x0172, 0x016e, 0x0168, 0x01d7, 0x01db, 0x01d9, 0x01d5,
00075     0x0174, 0x00dd, 0x0178, 0x0176, 0x0179, 0x017d, 0x017b, 0xffff,
00076     0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00077     /* 0x2b */
00078     0x00e1, 0x00e0, 0x00e4, 0x00e2, 0x0103, 0x01ce, 0x0101, 0x0105,
00079     0x00e5, 0x00e3, 0x0107, 0x0109, 0x010d, 0x00e7, 0x010b, 0x010f,
00080     0x00e9, 0x00e8, 0x00eb, 0x00ea, 0x011b, 0x0117, 0x0113, 0x0119,
00081     0x01f5, 0x011d, 0x011f, 0xffff, 0x0121, 0x0125, 0x00ed, 0x00ec,
00082     0x00ef, 0x00ee, 0x01d0, 0xffff, 0x012b, 0x012f, 0x0129, 0x0135,
00083     0x0137, 0x013a, 0x013e, 0x013c, 0x0144, 0x0148, 0x0146, 0x00f1,
00084     0x00f3, 0x00f2, 0x00f6, 0x00f4, 0x01d2, 0x0151, 0x014d, 0x00f5,
00085     0x0155, 0x0159, 0x0157, 0x015b, 0x015d, 0x0161, 0x015f, 0x0165,
00086     0x0163, 0x00fa, 0x00f9, 0x00fc, 0x00fb, 0x016d, 0x01d4, 0x0171,
00087     0x016b, 0x0173, 0x016f, 0x0169, 0x01d8, 0x01dc, 0x01da, 0x01d6,
00088     0x0175, 0x00fd, 0x00ff, 0x0177, 0x017a, 0x017e, 0x017c,
```

```
00089 };
00090 static const unsigned short jisx0212_2uni_page30[5801] = {
00091     /* 0x30 */
00092     0x4e02, 0x4e04, 0x4e05, 0x4e0c, 0x4e12, 0x4e1f, 0x4e23, 0x4e24,
00093     0x4e28, 0x4e2b, 0x4e2e, 0x4e2f, 0x4e30, 0x4e35, 0x4e40, 0x4e41,
00094     0x4e44, 0x4e47, 0x4e51, 0x4e5a, 0x4e5c, 0x4e63, 0x4e68, 0x4e69,
00095     0x4e74, 0x4e75, 0x4e79, 0x4e7f, 0x4e8d, 0x4e96, 0x4e97, 0x4e9d,
00096     0x4eaf, 0x4eb9, 0x4ec3, 0x4ed0, 0x4eda, 0x4edb, 0x4ee0, 0x4ee1,
00097     0x4ee2, 0x4ee8, 0x4eef, 0x4ef1, 0x4ef3, 0x4ef5, 0x4efd, 0x4efe,
00098     0x4eff, 0x4f00, 0x4f02, 0x4f03, 0x4f08, 0x4f0b, 0x4f0c, 0x4f12,
00099     0x4f15, 0x4f16, 0x4f17, 0x4f19, 0x4f2e, 0x4f31, 0x4f60, 0x4f33,
00100     0x4f35, 0x4f37, 0x4f39, 0x4f3b, 0x4f3e, 0x4f40, 0x4f42, 0x4f48,
00101     0x4f49, 0x4f4b, 0x4f4c, 0x4f52, 0x4f54, 0x4f56, 0x4f58, 0x4f5f,
00102     0x4f63, 0x4f6a, 0x4f6c, 0x4f6e, 0x4f71, 0x4f77, 0x4f78, 0x4f79,
00103     0x4f7a, 0x4f7d, 0x4f7e, 0x4f81, 0x4f82, 0x4f84,
00104     /* 0x31 */
00105     0x4f85, 0x4f89, 0x4f8a, 0x4f8c, 0x4f8e, 0x4f90, 0x4f92, 0x4f93,
00106     0x4f94, 0x4f97, 0x4f99, 0x4f9a, 0x4f9e, 0x4f9f, 0x4fb2, 0x4fb7,
00107     0x4fb9, 0x4fbb, 0x4fbc, 0x4fbd, 0x4fbe, 0x4fc0, 0x4fc1, 0x4fc5,
00108     0x4fc6, 0x4fc8, 0x4fc9, 0x4fcb, 0x4fcc, 0x4fcd, 0x4fcf, 0x4fd2,
00109     0x4fdc, 0x4fe0, 0x4fe2, 0x4ff0, 0x4ff2, 0x4ffc, 0x4ffd, 0x4fff,
00110     0x5000, 0x5001, 0x5004, 0x5007, 0x500a, 0x500c, 0x500e, 0x5010,
00111     0x5013, 0x5017, 0x5018, 0x501b, 0x501c, 0x501d, 0x501e, 0x5022,
00112     0x5027, 0x502e, 0x5030, 0x5032, 0x5033, 0x5035, 0x5040, 0x5041,
00113     0x5042, 0x5045, 0x5046, 0x504a, 0x504c, 0x504e, 0x5051, 0x5052,
00114     0x5053, 0x5057, 0x5059, 0x505f, 0x5060, 0x5062, 0x5063, 0x5066,
00115     0x5067, 0x506a, 0x506d, 0x5070, 0x5071, 0x507b, 0x5081, 0x5083,
00116     0x5084, 0x5086, 0x508a, 0x508e, 0x508f, 0x5090,
00117     /* 0x32 */
00118     0x5092, 0x5093, 0x5094, 0x5096, 0x509b, 0x509c, 0x509e, 0x509f,
00119     0x50a0, 0x50a1, 0x50a2, 0x50aa, 0x50af, 0x50b0, 0x50b9, 0x50ba,
00120     0x50bd, 0x50c0, 0x50c3, 0x50c4, 0x50c7, 0x50cc, 0x50ce, 0x50d0,
00121     0x50d3, 0x50d4, 0x50d8, 0x50dc, 0x50dd, 0x50df, 0x50e2, 0x50e4,
00122     0x50e6, 0x50e8, 0x50e9, 0x50ef, 0x50f1, 0x50f6, 0x50fa, 0x50fe,
00123     0x5103, 0x5106, 0x5107, 0x5108, 0x510b, 0x510c, 0x510d, 0x510e,
00124     0x510f, 0x5110, 0x5117, 0x5119, 0x511b, 0x511c, 0x511d, 0x511e,
00125     0x5123, 0x5127, 0x5128, 0x512c, 0x512d, 0x512f, 0x5131, 0x5133,
00126     0x5134, 0x5135, 0x5138, 0x5139, 0x5142, 0x514a, 0x514f, 0x5153,
00127     0x5155, 0x5157, 0x5158, 0x515f, 0x5164, 0x5166, 0x517e, 0x5183,
00128     0x5184, 0x518b, 0x518e, 0x5198, 0x519d, 0x51a1, 0x51a3, 0x51ad,
00129     0x51b8, 0x51ba, 0x51bc, 0x51be, 0x51bf, 0x51c2,
00130     /* 0x33 */
00131     0x51c8, 0x51cf, 0x51d1, 0x51d2, 0x51d3, 0x51d5, 0x51d8, 0x51de,
00132     0x51e2, 0x51e5, 0x51ee, 0x51f2, 0x51f3, 0x51f4, 0x51f7, 0x5201,
00133     0x5202, 0x5205, 0x5212, 0x5213, 0x5215, 0x5216, 0x5218, 0x5222,
00134     0x5228, 0x5231, 0x5232, 0x5235, 0x523c, 0x5245, 0x5249, 0x5255,
00135     0x5257, 0x5258, 0x525a, 0x525c, 0x525f, 0x5260, 0x5261, 0x5266,
00136     0x526e, 0x5277, 0x5278, 0x5279, 0x5280, 0x5282, 0x5285, 0x528a,
00137     0x528c, 0x5293, 0x5295, 0x5296, 0x5297, 0x5298, 0x529a, 0x529c,
00138     0x52a4, 0x52a5, 0x52a6, 0x52a7, 0x52af, 0x52b0, 0x52b6, 0x52b7,
00139     0x52b8, 0x52ba, 0x52bb, 0x52bd, 0x52c0, 0x52c4, 0x52c6, 0x52c8,
00140     0x52cc, 0x52cf, 0x52d1, 0x52d4, 0x52d6, 0x52db, 0x52dc, 0x52e1,
00141     0x52e5, 0x52e8, 0x52e9, 0x52ea, 0x52ec, 0x52f0, 0x52f1, 0x52f4,
00142     0x52f6, 0x52f7, 0x5300, 0x5303, 0x530a, 0x530b,
00143     /* 0x34 */
00144     0x530c, 0x5311, 0x5313, 0x5318, 0x531b, 0x531c, 0x531e, 0x531f,
00145     0x5325, 0x5327, 0x5328, 0x5329, 0x532b, 0x532c, 0x532d, 0x5330,
00146     0x5332, 0x5335, 0x533c, 0x533d, 0x533e, 0x5342, 0x534c, 0x534b,
00147     0x5359, 0x535b, 0x5361, 0x5363, 0x5365, 0x536c, 0x536d, 0x5372,
00148     0x5379, 0x537e, 0x5383, 0x5387, 0x5388, 0x538e, 0x5393, 0x5394,
00149     0x5399, 0x539d, 0x53a1, 0x53a4, 0x53aa, 0x53ab, 0x53af, 0x53b2,
00150     0x53b4, 0x53b5, 0x53b7, 0x53b8, 0x53ba, 0x53bd, 0x53c0, 0x53c5,
00151     0x53cf, 0x53d2, 0x53d3, 0x53d5, 0x53da, 0x53dd, 0x53de, 0x53e0,
00152     0x53e6, 0x53e7, 0x53f5, 0x5402, 0x5413, 0x541a, 0x5421, 0x5427,
00153     0x5428, 0x542a, 0x542f, 0x5431, 0x5434, 0x5435, 0x5443, 0x5444,
00154     0x5447, 0x544d, 0x544f, 0x545e, 0x5462, 0x5464, 0x5466, 0x5467,
00155     0x5469, 0x546b, 0x546d, 0x546e, 0x5474, 0x547f,
00156     /* 0x35 */
00157     0x5481, 0x5483, 0x5485, 0x5488, 0x5489, 0x548d, 0x5491, 0x5495,
00158     0x5496, 0x549c, 0x549f, 0x54a1, 0x54a6, 0x54a7, 0x54a9, 0x54aa,
00159     0x54ad, 0x54ae, 0x54b1, 0x54b7, 0x54b9, 0x54ba, 0x54bb, 0x54bf,
00160     0x54c6, 0x54ca, 0x54cd, 0x54ce, 0x54e0, 0x54ea, 0x54ec, 0x54ef,
00161     0x54f6, 0x54fc, 0x54fe, 0x54ff, 0x5500, 0x5501, 0x5505, 0x5508,
00162     0x5509, 0x550c, 0x550d, 0x550e, 0x5515, 0x552a, 0x552b, 0x5532,
00163     0x5535, 0x5536, 0x553b, 0x553c, 0x553d, 0x5541, 0x5547, 0x5549,
00164     0x554a, 0x554d, 0x5550, 0x5551, 0x5558, 0x555a, 0x555b, 0x555e,
00165     0x5560, 0x5561, 0x5564, 0x5566, 0x557f, 0x5581, 0x5582, 0x5586,
00166     0x5588, 0x558e, 0x558f, 0x5591, 0x5592, 0x5593, 0x5594, 0x5597,
00167     0x55a3, 0x55a4, 0x55ad, 0x55b2, 0x55bf, 0x55c1, 0x55c3, 0x55c6,
00168     0x55c9, 0x55cb, 0x55cc, 0x55ce, 0x55d1, 0x55d2,
00169     /* 0x36 */
00170     0x55d3, 0x55d7, 0x55d8, 0x55db, 0x55de, 0x55e2, 0x55e9, 0x55f6,
00171     0x55ff, 0x5605, 0x5608, 0x560a, 0x560d, 0x560e, 0x560f, 0x5610,
00172     0x5611, 0x5612, 0x5619, 0x562c, 0x5630, 0x5633, 0x5635, 0x5637,
00173     0x5639, 0x563b, 0x563c, 0x563d, 0x563f, 0x5640, 0x5641, 0x5643,
00174     0x5644, 0x5646, 0x5649, 0x564b, 0x564d, 0x564f, 0x5654, 0x565e,
00175     0x5660, 0x5661, 0x5662, 0x5663, 0x5666, 0x5669, 0x566d, 0x566f,
```

```
00176 0x5671, 0x5672, 0x5675, 0x5684, 0x5685, 0x5688, 0x568b, 0x568c,
00177 0x5695, 0x5699, 0x569a, 0x569d, 0x569e, 0x569f, 0x56a6, 0x56a7,
00178 0x56a8, 0x56a9, 0x56ab, 0x56ac, 0x56ad, 0x56b1, 0x56b3, 0x56b7,
00179 0x56be, 0x56c5, 0x56c9, 0x56ca, 0x56cb, 0x56cf, 0x56d0, 0x56cc,
00180 0x56cd, 0x56d9, 0x56dc, 0x56dd, 0x56df, 0x56e1, 0x56e4, 0x56e5,
00181 0x56e6, 0x56e7, 0x56e8, 0x56f1, 0x56eb, 0x56ed,
00182 /* 0x37 */
00183 0x56f6, 0x56f7, 0x5701, 0x5702, 0x5707, 0x570a, 0x570c, 0x5711,
00184 0x5715, 0x571a, 0x571b, 0x571d, 0x5720, 0x5722, 0x5723, 0x5724,
00185 0x5725, 0x5729, 0x572a, 0x572c, 0x572e, 0x572f, 0x5733, 0x5734,
00186 0x573d, 0x573e, 0x573f, 0x5745, 0x5746, 0x574c, 0x574d, 0x5752,
00187 0x5762, 0x5765, 0x5767, 0x5768, 0x576b, 0x576d, 0x576e, 0x576f,
00188 0x5770, 0x5771, 0x5773, 0x5774, 0x5775, 0x5777, 0x5779, 0x577a,
00189 0x577b, 0x577c, 0x577e, 0x5781, 0x5783, 0x578c, 0x5794, 0x5797,
00190 0x5799, 0x579a, 0x579c, 0x579d, 0x579e, 0x579f, 0x57a1, 0x5795,
00191 0x57a7, 0x57a8, 0x57a9, 0x57ac, 0x57b8, 0x57bd, 0x57c7, 0x57c8,
00192 0x57cc, 0x57cf, 0x57d5, 0x57dd, 0x57de, 0x57e4, 0x57e6, 0x57e7,
00193 0x57e9, 0x57ed, 0x57f0, 0x57f5, 0x57f6, 0x57f8, 0x57fd, 0x57fe,
00194 0x57ff, 0x5803, 0x5804, 0x5808, 0x5809, 0x57e1,
00195 /* 0x38 */
00196 0x580c, 0x580d, 0x581b, 0x581e, 0x581f, 0x5820, 0x5826, 0x5827,
00197 0x582d, 0x5832, 0x5839, 0x583f, 0x5849, 0x584c, 0x584d, 0x584f,
00198 0x5850, 0x5855, 0x585f, 0x5861, 0x5864, 0x5867, 0x5868, 0x5878,
00199 0x587c, 0x587f, 0x5880, 0x5881, 0x5887, 0x5888, 0x5889, 0x588a,
00200 0x588c, 0x588d, 0x588f, 0x5890, 0x5894, 0x5896, 0x589d, 0x58a0,
00201 0x58a1, 0x58a2, 0x58a6, 0x58a9, 0x58b1, 0x58b2, 0x58c4, 0x58bc,
00202 0x58c2, 0x58c8, 0x58cd, 0x58ce, 0x58d0, 0x58d2, 0x58d4, 0x58d6,
00203 0x58da, 0x58dd, 0x58e1, 0x58e2, 0x58e9, 0x58f3, 0x5905, 0x5906,
00204 0x590b, 0x590c, 0x5912, 0x5913, 0x5914, 0x8641, 0x591d, 0x5921,
00205 0x5923, 0x5924, 0x5928, 0x592f, 0x5930, 0x5933, 0x5935, 0x5936,
00206 0x593f, 0x5943, 0x5946, 0x5952, 0x5953, 0x5959, 0x595b, 0x595d,
00207 0x595e, 0x595f, 0x5961, 0x5963, 0x596b, 0x596d,
00208 /* 0x39 */
00209 0x596f, 0x5972, 0x5975, 0x5976, 0x5979, 0x597b, 0x597c, 0x598b,
00210 0x598c, 0x598e, 0x5992, 0x5995, 0x5997, 0x599f, 0x59a4, 0x59a7,
00211 0x59ad, 0x59ae, 0x59af, 0x59b0, 0x59b3, 0x59b7, 0x59ba, 0x59bc,
00212 0x59c1, 0x59c3, 0x59c4, 0x59c8, 0x59ca, 0x59cd, 0x59d2, 0x59dd,
00213 0x59de, 0x59df, 0x59e3, 0x59e4, 0x59e7, 0x59ee, 0x59ef, 0x59f1,
00214 0x59f2, 0x59f4, 0x59f7, 0x5a00, 0x5a04, 0x5a0c, 0x5a0d, 0x5a0e,
00215 0x5a12, 0x5a13, 0x5a1e, 0x5a23, 0x5a24, 0x5a27, 0x5a28, 0x5a2a,
00216 0x5a2d, 0x5a30, 0x5a44, 0x5a45, 0x5a47, 0x5a48, 0x5a4c, 0x5a50,
00217 0x5a55, 0x5a5e, 0x5a63, 0x5a65, 0x5a67, 0x5a6d, 0x5a77, 0x5a7a,
00218 0x5a7b, 0x5a7e, 0x5a8b, 0x5a90, 0x5a93, 0x5a96, 0x5a99, 0x5a9c,
00219 0x5a9e, 0x5a9f, 0x5aa0, 0x5aa2, 0x5aa7, 0x5aac, 0x5ab1, 0x5ab2,
00220 0x5ab3, 0x5ab5, 0x5ab8, 0x5aba, 0x5abb, 0x5abf,
00221 /* 0x3a */
00222 0x5ac4, 0x5ac6, 0x5ac8, 0x5acf, 0x5ada, 0x5adc, 0x5ae0, 0x5ae5,
00223 0x5aea, 0x5aee, 0x5aef, 0x5af5, 0x5afd, 0x5b00, 0x5b01, 0x5b08,
00224 0x5b17, 0x5b34, 0x5b19, 0x5b1b, 0x5b1d, 0x5b21, 0x5b25, 0x5b2d,
00225 0x5b38, 0x5b41, 0x5b4b, 0x5b4c, 0x5b52, 0x5b56, 0x5b5e, 0x5b68,
00226 0x5b6e, 0x5b6f, 0x5b7c, 0x5b7d, 0x5b7e, 0x5b7f, 0x5b81, 0x5b84,
00227 0x5b86, 0x5b8a, 0x5b8e, 0x5b90, 0x5b91, 0x5b93, 0x5b94, 0x5b96,
00228 0x5ba8, 0x5ba9, 0x5bac, 0x5bad, 0x5baf, 0x5bb1, 0x5bb2, 0x5bb7,
00229 0x5bba, 0x5bbc, 0x5bcc, 0x5bcd, 0x5bcd, 0x5bcf, 0x5bd6, 0x5bd7,
00230 0x5bd8, 0x5bd9, 0x5bda, 0x5be0, 0x5bef, 0x5bf1, 0x5bf4, 0x5bfd,
00231 0x5c0c, 0x5c17, 0x5c1e, 0x5c1f, 0x5c23, 0x5c26, 0x5c29, 0x5c2b,
00232 0x5c2c, 0x5c2e, 0x5c30, 0x5c32, 0x5c35, 0x5c36, 0x5c59, 0x5c5a,
00233 0x5c5c, 0x5c62, 0x5c63, 0x5c67, 0x5c68, 0x5c69,
00234 /* 0x3b */
00235 0x5c6d, 0x5c70, 0x5c74, 0x5c75, 0x5c7a, 0x5c7b, 0x5c7c, 0x5c7d,
00236 0x5c87, 0x5c88, 0x5c8a, 0x5c8f, 0x5c92, 0x5c9d, 0x5c9f, 0x5ca0,
00237 0x5ca2, 0x5ca3, 0x5ca6, 0x5caa, 0x5cb2, 0x5cb4, 0x5cb5, 0x5cba,
00238 0x5cc9, 0x5ccb, 0x5ccd, 0x5cd7, 0x5cee, 0x5cf1, 0x5cf2,
00239 0x5cf4, 0x5d01, 0x5d06, 0x5d0d, 0x5d12, 0x5d2b, 0x5d23, 0x5d24,
00240 0x5d26, 0x5d27, 0x5d31, 0x5d34, 0x5d39, 0x5d3d, 0x5d3f, 0x5d42,
00241 0x5d43, 0x5d46, 0x5d48, 0x5d55, 0x5d51, 0x5d59, 0x5d4a, 0x5d5f,
00242 0x5d60, 0x5d61, 0x5d62, 0x5d64, 0x5d6a, 0x5d6d, 0x5d70, 0x5d79,
00243 0x5d7a, 0x5d7e, 0x5d7f, 0x5d81, 0x5d83, 0x5d88, 0x5d8a, 0x5d92,
00244 0x5d93, 0x5d94, 0x5d95, 0x5d99, 0x5d9b, 0x5d9f, 0x5da0, 0x5da7,
00245 0x5dab, 0x5db0, 0x5db4, 0x5db8, 0x5db9, 0x5dc3, 0x5dc7, 0x5dcb,
00246 0x5dd0, 0x5dce, 0x5dd8, 0x5dd9, 0x5de0, 0x5de4,
00247 /* 0x3c */
00248 0x5de9, 0x5df8, 0x5df9, 0x5e00, 0x5e07, 0x5e0d, 0x5e12, 0x5e14,
00249 0x5e15, 0x5e18, 0x5e1f, 0x5e20, 0x5e2e, 0x5e28, 0x5e32, 0x5e35,
00250 0x5e3e, 0x5e4b, 0x5e50, 0x5e49, 0x5e51, 0x5e56, 0x5e58, 0x5e5b,
00251 0x5e5c, 0x5e5e, 0x5e68, 0x5e6a, 0x5e6b, 0x5e6c, 0x5e6d, 0x5e6e,
00252 0x5e70, 0x5e80, 0x5e8b, 0x5e8e, 0x5ea2, 0x5ea4, 0x5ea5, 0x5ea8,
00253 0x5eaa, 0x5eac, 0x5eb1, 0x5eb3, 0x5ebd, 0x5ebe, 0x5ebf, 0x5ec6,
00254 0x5ecc, 0x5ecb, 0x5ece, 0x5ed1, 0x5ed2, 0x5ed4, 0x5ed5, 0x5edc,
00255 0x5ede, 0x5ee5, 0x5eeb, 0x5ef0, 0x5ef0, 0x5ef0, 0x5ef0, 0x5ef0,
00256 0x5f19, 0x5f1c, 0x5f1d, 0x5f21, 0x5f22, 0x5f23, 0x5f24, 0x5f28,
00257 0x5f2b, 0x5f2c, 0x5f2e, 0x5f30, 0x5f34, 0x5f36, 0x5f3b, 0x5f3d,
00258 0x5f3f, 0x5f40, 0x5f44, 0x5f45, 0x5f47, 0x5f4d, 0x5f50, 0x5f54,
00259 0x5f58, 0x5f5b, 0x5f60, 0x5f63, 0x5f64, 0x5f67,
00260 /* 0x3d */
00261 0x5f6f, 0x5f72, 0x5f74, 0x5f75, 0x5f78, 0x5f7a, 0x5f7d, 0x5f7e,
00262 0x5f89, 0x5f8d, 0x5f8f, 0x5f96, 0x5f9c, 0x5f9d, 0x5fa2, 0x5fa7,
```



```

00263 0x5fab, 0x5fa4, 0x5fac, 0x5faf, 0x5fb0, 0x5fb1, 0x5fb8, 0x5fc4,
00264 0x5fc7, 0x5fc8, 0x5fc9, 0x5fcb, 0x5fd0, 0x5fd1, 0x5fd2, 0x5fd3,
00265 0x5fd4, 0x5fde, 0x5fe1, 0x5fe2, 0x5fe8, 0x5fe9, 0x5fea, 0x5fec,
00266 0x5fed, 0x5fee, 0x5fef, 0x5ff2, 0x5ff3, 0x5ff6, 0x5ffa, 0x5ffc,
00267 0x6007, 0x600a, 0x600d, 0x6013, 0x6014, 0x6017, 0x6018, 0x601a,
00268 0x601f, 0x6024, 0x602d, 0x6033, 0x6035, 0x6040, 0x6047, 0x6048,
00269 0x6049, 0x604c, 0x6051, 0x6054, 0x6056, 0x6057, 0x605d, 0x6061,
00270 0x6067, 0x6071, 0x607e, 0x607f, 0x6082, 0x6086, 0x6088, 0x608a,
00271 0x608e, 0x6091, 0x6093, 0x6095, 0x6098, 0x609d, 0x609e, 0x60a2,
00272 0x60a4, 0x60a5, 0x60a8, 0x60b0, 0x60b1, 0x60b7,
00273 /* 0x3e */
00274 0x60bb, 0x60be, 0x60c2, 0x60c4, 0x60c8, 0x60c9, 0x60ca, 0x60cb,
00275 0x60ce, 0x60cf, 0x60d4, 0x60d5, 0x60d9, 0x60db, 0x60dd, 0x60de,
00276 0x60e2, 0x60e5, 0x60f2, 0x60f5, 0x60f8, 0x60fc, 0x60fd, 0x6102,
00277 0x6107, 0x610a, 0x610c, 0x6110, 0x6111, 0x6112, 0x6113, 0x6114,
00278 0x6116, 0x6117, 0x6119, 0x611c, 0x611e, 0x6122, 0x612a, 0x612b,
00279 0x6130, 0x6131, 0x6135, 0x6136, 0x6137, 0x6139, 0x6141, 0x6145,
00280 0x6146, 0x6149, 0x615e, 0x6160, 0x616c, 0x6172, 0x6178, 0x617b,
00281 0x617c, 0x617f, 0x6180, 0x6181, 0x6183, 0x6184, 0x618b, 0x618d,
00282 0x6192, 0x6193, 0x6197, 0x6198, 0x619c, 0x619d, 0x619f, 0x61a0,
00283 0x61a5, 0x61a8, 0x61aa, 0x61ad, 0x61b8, 0x61b9, 0x61bc, 0x61c0,
00284 0x61c1, 0x61c2, 0x61ce, 0x61cf, 0x61d5, 0x61dc, 0x61dd, 0x61de,
00285 0x61df, 0x61e1, 0x61e2, 0x61e7, 0x61e9, 0x61e5,
00286 /* 0x3f */
00287 0x61ec, 0x61ed, 0x61ef, 0x6201, 0x6203, 0x6204, 0x6207, 0x6213,
00288 0x6215, 0x621c, 0x6220, 0x6222, 0x6223, 0x6227, 0x6229, 0x622b,
00289 0x6239, 0x623d, 0x6242, 0x6243, 0x6244, 0x6246, 0x624c, 0x6250,
00290 0x6251, 0x6252, 0x6254, 0x6256, 0x625a, 0x625c, 0x6264, 0x626d,
00291 0x626f, 0x6273, 0x627a, 0x627d, 0x628d, 0x628e, 0x628f, 0x6290,
00292 0x62a6, 0x62a8, 0x62b3, 0x62b6, 0x62b7, 0x62ba, 0x62be, 0x62bf,
00293 0x62c4, 0x62ce, 0x62d5, 0x62d6, 0x62da, 0x62ea, 0x62f2, 0x62f4,
00294 0x62fc, 0x62fd, 0x6303, 0x6304, 0x630a, 0x630b, 0x630d, 0x6310,
00295 0x6313, 0x6316, 0x6318, 0x6319, 0x6329, 0x632a, 0x632d, 0x6335,
00296 0x6339, 0x633c, 0x6341, 0x6342, 0x6343, 0x6344, 0x6346, 0x634a,
00297 0x634b, 0x634e, 0x6352, 0x6353, 0x6354, 0x6358, 0x635b, 0x6365,
00298 0x6366, 0x636c, 0x636d, 0x6371, 0x6374, 0x6375,
00299 /* 0x40 */
00300 0x6378, 0x637c, 0x637d, 0x637f, 0x6382, 0x6384, 0x6387, 0x638a,
00301 0x6390, 0x6394, 0x6395, 0x6399, 0x639a, 0x639e, 0x63a4, 0x63a6,
00302 0x63ad, 0x63ae, 0x63af, 0x63bd, 0x63c1, 0x63c5, 0x63c8, 0x63ce,
00303 0x63d1, 0x63d3, 0x63d4, 0x63d5, 0x63dc, 0x63e0, 0x63e5, 0x63ea,
00304 0x63ec, 0x63f2, 0x63f3, 0x63f5, 0x63f8, 0x63f9, 0x6409, 0x640a,
00305 0x6410, 0x6412, 0x6414, 0x6418, 0x641e, 0x6420, 0x6422, 0x6424,
00306 0x6425, 0x6429, 0x642a, 0x642f, 0x6430, 0x6435, 0x643d, 0x643f,
00307 0x644b, 0x644e, 0x6451, 0x6452, 0x6453, 0x6454, 0x645a, 0x645b,
00308 0x645c, 0x645d, 0x645f, 0x6460, 0x6461, 0x6463, 0x646d, 0x6473,
00309 0x6474, 0x647b, 0x647d, 0x6485, 0x6487, 0x648f, 0x6490, 0x6491,
00310 0x6498, 0x6499, 0x649b, 0x649d, 0x649f, 0x64a1, 0x64a3, 0x64a6,
00311 0x64a8, 0x64ac, 0x64b3, 0x64bd, 0x64be, 0x64bf,
00312 /* 0x41 */
00313 0x64c4, 0x64c9, 0x64ca, 0x64cb, 0x64cc, 0x64ce, 0x64d0, 0x64d1,
00314 0x64d5, 0x64d7, 0x64e4, 0x64e5, 0x64e9, 0x64ea, 0x64ed, 0x64f0,
00315 0x64f5, 0x64f7, 0x64fb, 0x64ff, 0x6501, 0x6504, 0x6508, 0x6509,
00316 0x650a, 0x650f, 0x6513, 0x6514, 0x6516, 0x6519, 0x651b, 0x651e,
00317 0x651f, 0x6522, 0x6526, 0x6529, 0x652e, 0x6531, 0x653a, 0x653c,
00318 0x653d, 0x6543, 0x6547, 0x6549, 0x6550, 0x6552, 0x6554, 0x655f,
00319 0x6560, 0x6567, 0x656b, 0x657a, 0x657d, 0x6581, 0x6585, 0x658a,
00320 0x6592, 0x6595, 0x6598, 0x659d, 0x65a0, 0x65a3, 0x65a6, 0x65ae,
00321 0x65b2, 0x65b3, 0x65b4, 0x65bf, 0x65c2, 0x65c8, 0x65c9, 0x65ce,
00322 0x65d0, 0x65d4, 0x65d8, 0x65df, 0x65f0, 0x65f2, 0x65f4,
00323 0x65f5, 0x65f9, 0x65fe, 0x65ff, 0x6600, 0x6604, 0x6608, 0x6609,
00324 0x660d, 0x6611, 0x6612, 0x6615, 0x6616, 0x661d,
00325 /* 0x42 */
00326 0x661e, 0x6621, 0x6622, 0x6623, 0x6624, 0x6626, 0x6629, 0x662a,
00327 0x662b, 0x662c, 0x662e, 0x6630, 0x6631, 0x6633, 0x6639, 0x6637,
00328 0x6640, 0x6645, 0x6646, 0x664a, 0x664c, 0x6651, 0x664e, 0x6657,
00329 0x6658, 0x6659, 0x665b, 0x665c, 0x6660, 0x6661, 0x66fb, 0x666a,
00330 0x666b, 0x666c, 0x667e, 0x6673, 0x6675, 0x667f, 0x6677, 0x6678,
00331 0x6679, 0x667b, 0x6680, 0x667c, 0x668b, 0x668c, 0x668d, 0x6690,
00332 0x6692, 0x6699, 0x669a, 0x669b, 0x669c, 0x669f, 0x66a0, 0x66a4,
00333 0x66ad, 0x66b1, 0x66b2, 0x66b5, 0x66bb, 0x66bf, 0x66c0, 0x66c2,
00334 0x66c3, 0x66c8, 0x66cc, 0x66ce, 0x66cf, 0x66d4, 0x66db, 0x66df,
00335 0x66e8, 0x66ee, 0x66ec, 0x66ee, 0x66fa, 0x6705, 0x6707, 0x670e,
00336 0x6713, 0x6719, 0x671c, 0x6720, 0x6722, 0x6733, 0x673e, 0x6745,
00337 0x6747, 0x6748, 0x674c, 0x6754, 0x6755, 0x675d,
00338 /* 0x43 */
00339 0x6766, 0x676c, 0x676e, 0x6774, 0x6776, 0x677b, 0x6781, 0x6784,
00340 0x678e, 0x678f, 0x6791, 0x6793, 0x6796, 0x6798, 0x6799, 0x679b,
00341 0x67b0, 0x67b1, 0x67b2, 0x67b5, 0x67bb, 0x67bc, 0x67bd, 0x67f9,
00342 0x67c0, 0x67c2, 0x67c3, 0x67c5, 0x67c8, 0x67c9, 0x67d2, 0x67d7,
00343 0x67d9, 0x67dc, 0x67e1, 0x67e6, 0x67f0, 0x67f2, 0x67f6, 0x67f7,
00344 0x6852, 0x6814, 0x6819, 0x681d, 0x681f, 0x6828, 0x6827, 0x682c,
00345 0x682d, 0x682f, 0x6830, 0x6831, 0x6833, 0x683b, 0x683f, 0x6844,
00346 0x6845, 0x684a, 0x684c, 0x6855, 0x6857, 0x6858, 0x685b, 0x686b,
00347 0x686e, 0x686f, 0x6870, 0x6871, 0x6872, 0x6875, 0x6879, 0x687a,
00348 0x687b, 0x687c, 0x6882, 0x6884, 0x6886, 0x6888, 0x6896, 0x6898,
00349 0x689a, 0x689c, 0x68a1, 0x68a3, 0x68a5, 0x68a9, 0x68aa, 0x68ae,

```



```
00350 0x68b2, 0x68bb, 0x68c5, 0x68c8, 0x68cc, 0x68cf,
00351 /* 0x44 */
00352 0x68d0, 0x68d1, 0x68d3, 0x68d6, 0x68d9, 0x68dc, 0x68dd, 0x68e5,
00353 0x68e8, 0x68ea, 0x68eb, 0x68ec, 0x68ed, 0x68f0, 0x68f1, 0x68f5,
00354 0x68f6, 0x68f8, 0x68fc, 0x68fd, 0x6906, 0x6909, 0x690a, 0x6910,
00355 0x6911, 0x6913, 0x6916, 0x6917, 0x691f, 0x6933, 0x6935, 0x6938,
00356 0x693b, 0x6942, 0x6945, 0x6949, 0x694e, 0x6957, 0x695b, 0x6963,
00357 0x6964, 0x6965, 0x6966, 0x6968, 0x6969, 0x696c, 0x6970, 0x6971,
00358 0x6972, 0x697a, 0x697b, 0x697f, 0x6980, 0x698d, 0x6992, 0x6996,
00359 0x6998, 0x69a1, 0x69a5, 0x69a6, 0x69a8, 0x69ab, 0x69ad, 0x69af,
00360 0x69b7, 0x69b8, 0x69ba, 0x69bc, 0x69c5, 0x69c8, 0x69d1, 0x69d6,
00361 0x69d7, 0x69e2, 0x69e5, 0x69ee, 0x69ef, 0x69f1, 0x69f3, 0x69f5,
00362 0x69fe, 0x6a00, 0x6a01, 0x6a03, 0x6a0f, 0x6a11, 0x6a15, 0x6a1a,
00363 0x6a1d, 0x6a20, 0x6a24, 0x6a28, 0x6a30, 0x6a32,
00364 /* 0x45 */
00365 0x6a34, 0x6a37, 0x6a3b, 0x6a3e, 0x6a3f, 0x6a45, 0x6a46, 0x6a49,
00366 0x6a4a, 0x6a4e, 0x6a50, 0x6a51, 0x6a52, 0x6a55, 0x6a56, 0x6a5b,
00367 0x6a64, 0x6a67, 0x6a6a, 0x6a71, 0x6a73, 0x6a7e, 0x6a81, 0x6a83,
00368 0x6a86, 0x6a87, 0x6a89, 0x6a8b, 0x6a91, 0x6a9b, 0x6a9d, 0x6a9e,
00369 0x6a9f, 0x6aa5, 0x6aab, 0x6aaf, 0x6ab0, 0x6ab1, 0x6ab4, 0x6abd,
00370 0x6abe, 0x6abf, 0x6ac6, 0x6ac9, 0x6ac8, 0x6acc, 0x6ad0, 0x6ad4,
00371 0x6ad5, 0x6ad6, 0x6adc, 0x6add, 0x6ae4, 0x6ae7, 0x6aec, 0x6af0,
00372 0x6af1, 0x6af2, 0x6afc, 0x6afd, 0x6b02, 0x6b03, 0x6b06, 0x6b07,
00373 0x6b09, 0x6b0f, 0x6b10, 0x6b11, 0x6b17, 0x6b1b, 0x6b1e, 0x6b24,
00374 0x6b28, 0x6b2b, 0x6b2c, 0x6b2f, 0x6b35, 0x6b36, 0x6b3b, 0x6b3f,
00375 0x6b46, 0x6b4a, 0x6b4d, 0x6b52, 0x6b56, 0x6b58, 0x6b5d, 0x6b60,
00376 0x6b67, 0x6b6b, 0x6b6e, 0x6b70, 0x6b75, 0x6b7d,
00377 /* 0x46 */
00378 0x6b7e, 0x6b82, 0x6b85, 0x6b97, 0x6b9b, 0x6b9f, 0x6ba0, 0x6ba2,
00379 0x6ba3, 0x6ba8, 0x6ba9, 0x6bac, 0x6bad, 0x6bae, 0x6bb0, 0x6bb8,
00380 0x6bb9, 0x6bbd, 0x6bbe, 0x6bc3, 0x6bc4, 0x6bc9, 0x6bcc, 0x6bd6,
00381 0x6bda, 0x6be1, 0x6be3, 0x6be6, 0x6be7, 0x6bee, 0x6bf1, 0x6bf7,
00382 0x6bf9, 0x6bff, 0x6c02, 0x6c04, 0x6c05, 0x6c09, 0x6c0d, 0x6c0e,
00383 0x6c10, 0x6c12, 0x6c19, 0x6c1f, 0x6c26, 0x6c27, 0x6c28, 0x6c2c,
00384 0x6c2e, 0x6c33, 0x6c35, 0x6c36, 0x6c3a, 0x6c3b, 0x6c3f, 0x6c4a,
00385 0x6c4b, 0x6c4d, 0x6c4f, 0x6c52, 0x6c54, 0x6c59, 0x6c5b, 0x6c5c,
00386 0x6c6b, 0x6c6d, 0x6c6f, 0x6c74, 0x6c76, 0x6c78, 0x6c79, 0x6c7b,
00387 0x6c85, 0x6c86, 0x6c87, 0x6c89, 0x6c94, 0x6c95, 0x6c97, 0x6c98,
00388 0x6c9c, 0x6c9f, 0x6cb0, 0x6cb2, 0x6cb4, 0x6cc2, 0x6cc6, 0x6ccd,
00389 0x6ccf, 0x6cd0, 0x6cd1, 0x6cd2, 0x6cd4, 0x6cd6,
00390 /* 0x47 */
00391 0x6cda, 0x6cdc, 0x6ce0, 0x6ce7, 0x6ce9, 0x6ceb, 0x6cec, 0x6cee,
00392 0x6cf2, 0x6cf4, 0x6d04, 0x6d07, 0x6d0a, 0x6d0e, 0x6d0f, 0x6d11,
00393 0x6d13, 0x6d1a, 0x6d26, 0x6d27, 0x6d28, 0x6d67, 0x6d2e, 0x6d2f,
00394 0x6d31, 0x6d39, 0x6d3c, 0x6d3f, 0x6d57, 0x6d5e, 0x6d5f, 0x6d61,
00395 0x6d65, 0x6d67, 0x6d6f, 0x6d70, 0x6d7c, 0x6d82, 0x6d87, 0x6d91,
00396 0x6d92, 0x6d94, 0x6d96, 0x6d97, 0x6d98, 0x6daa, 0x6dac, 0x6db4,
00397 0x6db7, 0x6db9, 0x6dbd, 0x6dbf, 0x6dc4, 0x6dc8, 0x6dca, 0x6dce,
00398 0x6dcf, 0x6dd6, 0x6ddb, 0x6ddd, 0x6ddf, 0x6de0, 0x6de2, 0x6de5,
00399 0x6de9, 0x6def, 0x6df0, 0x6df4, 0x6df6, 0x6dfc, 0x6e00, 0x6e04,
00400 0x6e1e, 0x6e22, 0x6e27, 0x6e32, 0x6e36, 0x6e39, 0x6e3b, 0x6e3c,
00401 0x6e44, 0x6e45, 0x6e48, 0x6e49, 0x6e4b, 0x6e4f, 0x6e51, 0x6e52,
00402 0x6e53, 0x6e54, 0x6e57, 0x6e5c, 0x6e5d, 0x6e5e,
00403 /* 0x48 */
00404 0x6e62, 0x6e63, 0x6e68, 0x6e73, 0x6e7b, 0x6e7d, 0x6e8d, 0x6e93,
00405 0x6e99, 0x6ea0, 0x6ea7, 0x6ead, 0x6eae, 0x6eb1, 0x6eb3, 0x6ebb,
00406 0x6ebf, 0x6ec0, 0x6ec1, 0x6ec3, 0x6ec7, 0x6ec8, 0x6eca, 0x6ecd,
00407 0x6ece, 0x6ecf, 0x6eeb, 0x6eed, 0x6eee, 0x6ef9, 0x6efb, 0x6efd,
00408 0x6f04, 0x6f08, 0x6f0a, 0x6f0c, 0x6f0d, 0x6f16, 0x6f18, 0x6f1a,
00409 0x6f1b, 0x6f26, 0x6f29, 0x6f2a, 0x6f2f, 0x6f30, 0x6f33, 0x6f36,
00410 0x6f3b, 0x6f3c, 0x6f2d, 0x6f4f, 0x6f51, 0x6f52, 0x6f53, 0x6f57,
00411 0x6f59, 0x6f5a, 0x6f5d, 0x6f5e, 0x6f61, 0x6f62, 0x6f68, 0x6f6c,
00412 0x6f7d, 0x6f7e, 0x6f83, 0x6f87, 0x6f88, 0x6f8b, 0x6f8c, 0x6f8d,
00413 0x6f90, 0x6f92, 0x6f93, 0x6f94, 0x6f96, 0x6f9a, 0x6f9f, 0x6fa0,
00414 0x6fa5, 0x6fa6, 0x6fa7, 0x6fa8, 0x6fae, 0x6faf, 0x6fb0, 0x6fb5,
00415 0x6fb6, 0x6fbc, 0x6fc5, 0x6fc7, 0x6fc8, 0x6fca,
00416 /* 0x49 */
00417 0x6fda, 0x6fde, 0x6fe8, 0x6fe9, 0x6ff0, 0x6ff5, 0x6ff9, 0x6ffc,
00418 0x6ffd, 0x7000, 0x7005, 0x7006, 0x7007, 0x700d, 0x7017, 0x7020,
00419 0x7023, 0x702f, 0x7034, 0x7037, 0x7039, 0x703c, 0x7043, 0x7044,
00420 0x7048, 0x7049, 0x704a, 0x704b, 0x7054, 0x7055, 0x705d, 0x705e,
00421 0x704e, 0x7064, 0x7065, 0x706c, 0x706e, 0x7075, 0x7076, 0x707e,
00422 0x7081, 0x7085, 0x7086, 0x7094, 0x7095, 0x7096, 0x7097, 0x7098,
00423 0x709b, 0x70a4, 0x70ab, 0x70b0, 0x70b1, 0x70b4, 0x70b7, 0x70ca,
00424 0x70d1, 0x70d3, 0x70d4, 0x70d5, 0x70d6, 0x70d8, 0x70dc, 0x70e4,
00425 0x70fa, 0x7103, 0x7104, 0x7105, 0x7106, 0x7107, 0x710b, 0x710c,
00426 0x710f, 0x711e, 0x7120, 0x712b, 0x712d, 0x712f, 0x7130, 0x7131,
00427 0x7138, 0x7141, 0x7145, 0x7146, 0x7147, 0x714a, 0x714b, 0x7150,
00428 0x7152, 0x7157, 0x715a, 0x715c, 0x715e, 0x7160,
00429 /* 0x4a */
00430 0x7168, 0x7179, 0x7180, 0x7185, 0x7187, 0x718c, 0x7192, 0x719a,
00431 0x719b, 0x71a0, 0x71a2, 0x71af, 0x71b0, 0x71b2, 0x71b3, 0x71ba,
00432 0x71bf, 0x71c0, 0x71c1, 0x71c4, 0x71cb, 0x71cc, 0x71d3, 0x71d6,
00433 0x71d9, 0x71da, 0x71dc, 0x71f8, 0x71fe, 0x7200, 0x7207, 0x7208,
00434 0x7209, 0x7213, 0x7217, 0x721a, 0x721d, 0x721f, 0x722a, 0x722b,
00435 0x722f, 0x7234, 0x7238, 0x7239, 0x7241, 0x7242, 0x7243, 0x7245,
00436 0x724e, 0x724f, 0x7250, 0x7253, 0x7255, 0x7256, 0x725a, 0x725c,
```

```

00437 0x725e, 0x7260, 0x7263, 0x7268, 0x726b, 0x726e, 0x726f, 0x7271,
00438 0x7277, 0x7278, 0x727b, 0x727c, 0x727f, 0x7284, 0x7289, 0x728d,
00439 0x728e, 0x7293, 0x729b, 0x72a8, 0x72ad, 0x72ae, 0x72b1, 0x72b4,
00440 0x72be, 0x72c1, 0x72c7, 0x72c9, 0x72cc, 0x72d5, 0x72d6, 0x72d8,
00441 0x72df, 0x72e5, 0x72f3, 0x72f4, 0x72fa, 0x72fb,
00442 /* 0x4b */
00443 0x72fe, 0x7302, 0x7304, 0x7305, 0x7307, 0x730b, 0x730d, 0x7312,
00444 0x7313, 0x7318, 0x7319, 0x731e, 0x7322, 0x7324, 0x7327, 0x7328,
00445 0x732c, 0x7331, 0x7332, 0x7335, 0x733a, 0x733b, 0x733d, 0x7343,
00446 0x734d, 0x7350, 0x7352, 0x7356, 0x7358, 0x735d, 0x735e, 0x735f,
00447 0x7360, 0x7366, 0x7367, 0x7369, 0x736b, 0x736c, 0x736e, 0x736f,
00448 0x7371, 0x7377, 0x7379, 0x737c, 0x7380, 0x7381, 0x7383, 0x7385,
00449 0x7386, 0x738e, 0x7390, 0x7393, 0x7395, 0x7397, 0x7398, 0x739c,
00450 0x739e, 0x739f, 0x73a0, 0x73a2, 0x73a5, 0x73a6, 0x73aa, 0x73ab,
00451 0x73ad, 0x73b5, 0x73b7, 0x73b9, 0x73bc, 0x73bd, 0x73bf, 0x73c5,
00452 0x73c6, 0x73c9, 0x73cb, 0x73cc, 0x73cf, 0x73d2, 0x73d3, 0x73d6,
00453 0x73d9, 0x73dd, 0x73e1, 0x73e3, 0x73e6, 0x73e7, 0x73e9, 0x73f4,
00454 0x73f5, 0x73f7, 0x73f9, 0x73fa, 0x73fb, 0x73fd,
00455 /* 0x4c */
00456 0x73ff, 0x7400, 0x7401, 0x7404, 0x7407, 0x740a, 0x7411, 0x741a,
00457 0x741b, 0x7424, 0x7426, 0x7428, 0x7429, 0x742a, 0x742b, 0x742c,
00458 0x742d, 0x742e, 0x742f, 0x7430, 0x7431, 0x7439, 0x7440, 0x7443,
00459 0x7444, 0x7446, 0x7447, 0x744b, 0x744d, 0x7451, 0x7452, 0x7457,
00460 0x745d, 0x7462, 0x7466, 0x7467, 0x7468, 0x746b, 0x746d, 0x746e,
00461 0x7471, 0x7472, 0x7480, 0x7481, 0x7485, 0x7486, 0x7487, 0x7489,
00462 0x748f, 0x7490, 0x7491, 0x7492, 0x7498, 0x7499, 0x749a, 0x749c,
00463 0x749f, 0x74a0, 0x74a1, 0x74a3, 0x74a6, 0x74a8, 0x74a9, 0x74aa,
00464 0x74ab, 0x74ae, 0x74af, 0x74b1, 0x74b2, 0x74b5, 0x74b9, 0x74bb,
00465 0x74bf, 0x74c8, 0x74c9, 0x74cc, 0x74d0, 0x74d3, 0x74d8, 0x74da,
00466 0x74db, 0x74de, 0x74df, 0x74e4, 0x74e8, 0x74ea, 0x74eb, 0x74ef,
00467 0x74f4, 0x74fa, 0x74fb, 0x74fc, 0x74ff, 0x7506,
00468 /* 0x4d */
00469 0x7512, 0x7516, 0x7517, 0x7520, 0x7521, 0x7524, 0x7527, 0x7529,
00470 0x752a, 0x752f, 0x7536, 0x7539, 0x753d, 0x753e, 0x753f, 0x7540,
00471 0x7543, 0x7547, 0x7548, 0x754e, 0x7550, 0x7552, 0x7557, 0x755e,
00472 0x755f, 0x7561, 0x756f, 0x7571, 0x7579, 0x757a, 0x757b, 0x757c,
00473 0x757d, 0x757e, 0x7581, 0x7585, 0x7590, 0x7592, 0x7593, 0x7595,
00474 0x7599, 0x759c, 0x75a2, 0x75a4, 0x75b4, 0x75ba, 0x75bf, 0x75c0,
00475 0x75c1, 0x75c4, 0x75c6, 0x75cc, 0x75ce, 0x75cf, 0x75d7, 0x75dc,
00476 0x75df, 0x75e0, 0x75e1, 0x75e4, 0x75e7, 0x75ec, 0x75ee, 0x75ef,
00477 0x75f1, 0x75f9, 0x7600, 0x7602, 0x7603, 0x7604, 0x7607, 0x7608,
00478 0x760a, 0x760c, 0x760f, 0x7612, 0x7613, 0x7615, 0x7616, 0x7619,
00479 0x761b, 0x761c, 0x761d, 0x761e, 0x7623, 0x7625, 0x7626, 0x7629,
00480 0x762d, 0x7632, 0x7633, 0x7635, 0x7638, 0x7639,
00481 /* 0x4e */
00482 0x763a, 0x763c, 0x764a, 0x7640, 0x7641, 0x7643, 0x7644, 0x7645,
00483 0x7649, 0x764b, 0x7655, 0x7659, 0x765f, 0x7664, 0x7665, 0x766d,
00484 0x766e, 0x766f, 0x7671, 0x7674, 0x7681, 0x7685, 0x768c, 0x768d,
00485 0x7695, 0x769b, 0x769c, 0x769d, 0x769f, 0x76a0, 0x76a2, 0x76a3,
00486 0x76a4, 0x76a5, 0x76a6, 0x76a7, 0x76a8, 0x76aa, 0x76ad, 0x76bd,
00487 0x76c1, 0x76c5, 0x76c9, 0x76cb, 0x76cc, 0x76ce, 0x76d4, 0x76d9,
00488 0x76e0, 0x76e6, 0x76e8, 0x76ec, 0x76f0, 0x76f1, 0x76f6, 0x76f9,
00489 0x76fc, 0x7700, 0x7706, 0x770a, 0x770e, 0x7712, 0x7714, 0x7715,
00490 0x7717, 0x7719, 0x771a, 0x771c, 0x7722, 0x7728, 0x772d, 0x772e,
00491 0x772f, 0x7734, 0x7735, 0x7736, 0x7739, 0x773d, 0x773e, 0x7742,
00492 0x7745, 0x7746, 0x774a, 0x774d, 0x774e, 0x774f, 0x7752, 0x7756,
00493 0x7757, 0x775c, 0x775e, 0x775f, 0x7760, 0x7762,
00494 /* 0x4f */
00495 0x7764, 0x7767, 0x776a, 0x776c, 0x7770, 0x7772, 0x7773, 0x7774,
00496 0x777a, 0x777d, 0x7780, 0x7784, 0x778c, 0x778d, 0x7794, 0x7795,
00497 0x7796, 0x779a, 0x779f, 0x77a2, 0x77a7, 0x77aa, 0x77ae, 0x77af,
00498 0x77b1, 0x77b5, 0x77be, 0x77c3, 0x77c9, 0x77d1, 0x77d2, 0x77d5,
00499 0x77d9, 0x77de, 0x77df, 0x77e0, 0x77e4, 0x77e6, 0x77ea, 0x77ec,
00500 0x77f0, 0x77f1, 0x77f4, 0x77f8, 0x77fb, 0x7805, 0x7806, 0x7809,
00501 0x780d, 0x780e, 0x7811, 0x781d, 0x7821, 0x7822, 0x7823, 0x782d,
00502 0x782e, 0x7830, 0x7835, 0x7837, 0x7843, 0x7844, 0x7847, 0x7848,
00503 0x784c, 0x784e, 0x7852, 0x785c, 0x785e, 0x7860, 0x7861, 0x7863,
00504 0x7864, 0x7868, 0x786a, 0x786e, 0x787a, 0x787e, 0x788a, 0x788f,
00505 0x7894, 0x7898, 0x78a1, 0x789d, 0x789e, 0x789f, 0x78a4, 0x78a8,
00506 0x78ac, 0x78ad, 0x78b0, 0x78b1, 0x78b2, 0x78b3,
00507 /* 0x50 */
00508 0x78bb, 0x78bd, 0x78bf, 0x78c7, 0x78c8, 0x78c9, 0x78cc, 0x78ce,
00509 0x78d2, 0x78d3, 0x78d5, 0x78d6, 0x78e4, 0x78db, 0x78df, 0x78e0,
00510 0x78e1, 0x78e6, 0x78ea, 0x78f2, 0x78f3, 0x7900, 0x78f6, 0x78f7,
00511 0x78fa, 0x78fb, 0x78ff, 0x7906, 0x790c, 0x7910, 0x791a, 0x791c,
00512 0x791e, 0x791f, 0x7920, 0x7925, 0x7927, 0x7929, 0x792d, 0x7931,
00513 0x7934, 0x7935, 0x793b, 0x793d, 0x793f, 0x7944, 0x7945, 0x7946,
00514 0x794a, 0x794b, 0x794f, 0x7951, 0x7954, 0x7958, 0x795b, 0x795c,
00515 0x7967, 0x7969, 0x796b, 0x7972, 0x7979, 0x797b, 0x797c, 0x797e,
00516 0x798b, 0x798c, 0x7991, 0x7993, 0x7994, 0x7995, 0x7996, 0x7998,
00517 0x799b, 0x799c, 0x79a1, 0x79a8, 0x79a9, 0x79ab, 0x79af, 0x79b1,
00518 0x79b4, 0x79b8, 0x79bb, 0x79c2, 0x79c4, 0x79c7, 0x79c8, 0x79ca,
00519 0x79cf, 0x79d4, 0x79d6, 0x79da, 0x79dd, 0x79de,
00520 /* 0x51 */
00521 0x79e0, 0x79e2, 0x79e5, 0x79ea, 0x79eb, 0x79ed, 0x79f1, 0x79f8,
00522 0x79fc, 0x79fd, 0x79fa, 0x79fb, 0x79fc, 0x79fd, 0x79fe, 0x79ff,
00523 0x7a15, 0x7a1b, 0x7a1e, 0x7a21, 0x7a27, 0x7a2b, 0x7a2d, 0x7a2f,

```

```
00524 0x7a30, 0x7a34, 0x7a35, 0x7a38, 0x7a39, 0x7a3a, 0x7a44, 0x7a45,
00525 0x7a47, 0x7a48, 0x7a4c, 0x7a55, 0x7a56, 0x7a59, 0x7a5c, 0x7a5d,
00526 0x7a5f, 0x7a60, 0x7a65, 0x7a67, 0x7a6a, 0x7a6d, 0x7a75, 0x7a78,
00527 0x7a7e, 0x7a80, 0x7a82, 0x7a85, 0x7a86, 0x7a8a, 0x7a8b, 0x7a90,
00528 0x7a91, 0x7a94, 0x7a9e, 0x7aa0, 0x7aa3, 0x7aac, 0x7ab3, 0x7ab5,
00529 0x7ab9, 0x7abb, 0x7abc, 0x7ac6, 0x7ac9, 0x7acc, 0x7ace, 0x7ad1,
00530 0x7adb, 0x7ae8, 0x7ae9, 0x7aeb, 0x7aec, 0x7af1, 0x7af4, 0x7afb,
00531 0x7afd, 0x7afe, 0x7b07, 0x7b14, 0x7b1f, 0x7b23, 0x7b27, 0x7b29,
00532 0x7b2a, 0x7b2b, 0x7b2d, 0x7b2e, 0x7b2f, 0x7b30,
00533 /* 0x52 */
00534 0x7b31, 0x7b34, 0x7b3d, 0x7b3f, 0x7b40, 0x7b41, 0x7b47, 0x7b4e,
00535 0x7b55, 0x7b60, 0x7b64, 0x7b66, 0x7b69, 0x7b6a, 0x7b6d, 0x7b6f,
00536 0x7b72, 0x7b73, 0x7b77, 0x7b84, 0x7b89, 0x7b8e, 0x7b90, 0x7b91,
00537 0x7b96, 0x7b9b, 0x7b9e, 0x7ba0, 0x7ba5, 0x7bac, 0x7baF, 0x7bb0,
00538 0x7bb2, 0x7bb5, 0x7bb6, 0x7bba, 0x7bbb, 0x7bbc, 0x7bbd, 0x7bc2,
00539 0x7bc5, 0x7bc8, 0x7bca, 0x7bd4, 0x7bd6, 0x7bd7, 0x7bd9, 0x7bda,
00540 0x7bdb, 0x7be8, 0x7bea, 0x7bf2, 0x7bf4, 0x7bf5, 0x7bf8, 0x7bf9,
00541 0x7bfa, 0x7bfc, 0x7bfe, 0x7c01, 0x7c02, 0x7c03, 0x7c04, 0x7c06,
00542 0x7c09, 0x7c0b, 0x7c0c, 0x7c0e, 0x7c0f, 0x7c19, 0x7c1b, 0x7c20,
00543 0x7c25, 0x7c26, 0x7c28, 0x7c2c, 0x7c31, 0x7c33, 0x7c34, 0x7c36,
00544 0x7c39, 0x7c3a, 0x7c46, 0x7c4a, 0x7c55, 0x7c51, 0x7c52, 0x7c53,
00545 0x7c59, 0x7c5a, 0x7c5b, 0x7c5c, 0x7c5d, 0x7c5e,
00546 /* 0x53 */
00547 0x7c61, 0x7c63, 0x7c67, 0x7c69, 0x7c6d, 0x7c6e, 0x7c70, 0x7c72,
00548 0x7c79, 0x7c7c, 0x7c7d, 0x7c86, 0x7c87, 0x7c8f, 0x7c94, 0x7c9e,
00549 0x7ca0, 0x7ca6, 0x7cb0, 0x7cb6, 0x7cb7, 0x7cba, 0x7cbb, 0x7cbc,
00550 0x7cbf, 0x7cc4, 0x7cc7, 0x7cc8, 0x7cc9, 0x7ccd, 0x7ccf, 0x7cd3,
00551 0x7cd4, 0x7cd5, 0x7cd7, 0x7cd9, 0x7cda, 0x7cdd, 0x7ce6, 0x7ce9,
00552 0x7ceb, 0x7cf5, 0x7d03, 0x7d07, 0x7d08, 0x7d09, 0x7d0f, 0x7d11,
00553 0x7d12, 0x7d13, 0x7d16, 0x7d1d, 0x7d1e, 0x7d23, 0x7d26, 0x7d2a,
00554 0x7d2d, 0x7d31, 0x7d3c, 0x7d3d, 0x7d3e, 0x7d40, 0x7d41, 0x7d47,
00555 0x7d48, 0x7d4d, 0x7d51, 0x7d53, 0x7d57, 0x7d59, 0x7d5a, 0x7d5c,
00556 0x7d5d, 0x7d65, 0x7d67, 0x7d6a, 0x7d70, 0x7d78, 0x7d7a, 0x7d7b,
00557 0x7d7f, 0x7d81, 0x7d82, 0x7d83, 0x7d85, 0x7d86, 0x7d88, 0x7d8b,
00558 0x7d8c, 0x7d8d, 0x7d91, 0x7d96, 0x7d97, 0x7d9d,
00559 /* 0x54 */
00560 0x7d9e, 0x7da6, 0x7da7, 0x7daa, 0x7db3, 0x7db6, 0x7db7, 0x7db9,
00561 0x7dc2, 0x7dc3, 0x7dc4, 0x7dc5, 0x7dc6, 0x7dcc, 0x7dcd, 0x7dce,
00562 0x7dd7, 0x7dd9, 0x7de0, 0x7de2, 0x7de5, 0x7de6, 0x7dea, 0x7deb,
00563 0x7ded, 0x7df1, 0x7df5, 0x7df6, 0x7df9, 0x7dfa, 0x7e08, 0x7e10,
00564 0x7e11, 0x7e15, 0x7e17, 0x7e1c, 0x7e1d, 0x7e20, 0x7e27, 0x7e28,
00565 0x7e2c, 0x7e2d, 0x7e2f, 0x7e33, 0x7e36, 0x7e3f, 0x7e44, 0x7e45,
00566 0x7e47, 0x7e4e, 0x7e50, 0x7e52, 0x7e58, 0x7e5f, 0x7e61, 0x7e62,
00567 0x7e65, 0x7e6b, 0x7e6e, 0x7e6f, 0x7e73, 0x7e78, 0x7e7e, 0x7e81,
00568 0x7e86, 0x7e87, 0x7e8a, 0x7e8d, 0x7e91, 0x7e95, 0x7e98, 0x7e9a,
00569 0x7e9d, 0x7e9e, 0x7f3c, 0x7f3b, 0x7f3d, 0x7f3e, 0x7f3f, 0x7f43,
00570 0x7f44, 0x7f47, 0x7f4f, 0x7f52, 0x7f53, 0x7f5b, 0x7f5c, 0x7f5d,
00571 0x7f61, 0x7f63, 0x7f64, 0x7f65, 0x7f66, 0x7f6d,
00572 /* 0x55 */
00573 0x7f71, 0x7f7d, 0x7f7e, 0x7f7f, 0x7f80, 0x7f8b, 0x7f8d, 0x7f8f,
00574 0x7f90, 0x7f91, 0x7f96, 0x7f97, 0x7f9c, 0x7fa1, 0x7fa2, 0x7fa6,
00575 0x7faa, 0x7fad, 0x7fb4, 0x7fbc, 0x7fbf, 0x7fc0, 0x7fc3, 0x7fc8,
00576 0x7fce, 0x7fcf, 0x7fdb, 0x7fdf, 0x7fe3, 0x7fe5, 0x7fe8, 0x7fec,
00577 0x7fee, 0x7fef, 0x7ff2, 0x7ffa, 0x7ffd, 0x7ffe, 0x7fff, 0x8007,
00578 0x8008, 0x800a, 0x800d, 0x800e, 0x800f, 0x8011, 0x8013, 0x8014,
00579 0x8016, 0x801d, 0x801e, 0x801f, 0x8020, 0x8024, 0x8026, 0x802c,
00580 0x802e, 0x8030, 0x8034, 0x8035, 0x8037, 0x8039, 0x803a, 0x803c,
00581 0x803e, 0x8040, 0x8044, 0x8060, 0x8064, 0x8066, 0x806d, 0x8071,
00582 0x8075, 0x8081, 0x8088, 0x808e, 0x809c, 0x809e, 0x80a6, 0x80a7,
00583 0x80ab, 0x80b8, 0x80b9, 0x80c8, 0x80cd, 0x80cf, 0x80d2, 0x80d4,
00584 0x80d5, 0x80d7, 0x80d8, 0x80e0, 0x80ed, 0x80ee,
00585 /* 0x56 */
00586 0x80f0, 0x80f2, 0x80f3, 0x80f6, 0x80f9, 0x80fa, 0x80fe, 0x8103,
00587 0x810b, 0x8116, 0x8117, 0x8118, 0x811c, 0x811e, 0x8120, 0x8124,
00588 0x8127, 0x812c, 0x8130, 0x8135, 0x813a, 0x813c, 0x8145, 0x8147,
00589 0x814a, 0x814c, 0x8152, 0x8157, 0x8160, 0x8161, 0x8167, 0x8168,
00590 0x8169, 0x816d, 0x816f, 0x8177, 0x8181, 0x8190, 0x8184, 0x8185,
00591 0x8186, 0x818b, 0x818e, 0x8196, 0x8198, 0x819b, 0x819e, 0x81a2,
00592 0x81ae, 0x81b2, 0x81b4, 0x81bb, 0x81cb, 0x81c3, 0x81c5, 0x81ca,
00593 0x81ce, 0x81cf, 0x81d5, 0x81d7, 0x81db, 0x81dd, 0x81de, 0x81e1,
00594 0x81e4, 0x81eb, 0x81ec, 0x81f0, 0x81f1, 0x81f2, 0x81f5, 0x81fe,
00595 0x81f8, 0x81f9, 0x81fd, 0x81ff, 0x8200, 0x8203, 0x820f, 0x8213,
00596 0x8214, 0x8219, 0x821a, 0x821d, 0x8221, 0x8222, 0x8228, 0x8232,
00597 0x8234, 0x823a, 0x8243, 0x8244, 0x8245, 0x8246,
00598 /* 0x57 */
00599 0x824b, 0x824e, 0x824f, 0x8251, 0x8256, 0x825c, 0x8260, 0x8263,
00600 0x8267, 0x826d, 0x8274, 0x827b, 0x827d, 0x827f, 0x8280, 0x8281,
00601 0x8283, 0x8284, 0x8287, 0x8289, 0x828a, 0x828e, 0x8291, 0x8294,
00602 0x8296, 0x8298, 0x829a, 0x829b, 0x82a0, 0x82a1, 0x82a4, 0x82a4,
00603 0x82a7, 0x82a8, 0x82a9, 0x82aa, 0x82ae, 0x82b0, 0x82b2, 0x82b4,
00604 0x82b7, 0x82ba, 0x82bc, 0x82be, 0x82bf, 0x82c6, 0x82d0, 0x82d5,
00605 0x82da, 0x82e0, 0x82e2, 0x82e4, 0x82e8, 0x82ea, 0x82ed, 0x82ef,
00606 0x82f6, 0x82f7, 0x82fd, 0x82fe, 0x8300, 0x8301, 0x8307, 0x8308,
00607 0x830a, 0x830b, 0x8354, 0x831b, 0x831d, 0x831e, 0x831f, 0x8321,
00608 0x8322, 0x832c, 0x832d, 0x832e, 0x8330, 0x8333, 0x8337, 0x833a,
00609 0x833c, 0x833d, 0x8342, 0x8343, 0x8344, 0x8347, 0x834d, 0x834e,
00610 0x8351, 0x8355, 0x8356, 0x8357, 0x8370, 0x8378,
```

```

00611  /* 0x58 */
00612  0x837d, 0x837f, 0x8380, 0x8382, 0x8384, 0x8386, 0x838d, 0x8392,
00613  0x8394, 0x8395, 0x8398, 0x8399, 0x839b, 0x839c, 0x839d, 0x83a6,
00614  0x83a7, 0x83a9, 0x83ac, 0x83be, 0x83bf, 0x83c0, 0x83c7, 0x83c9,
00615  0x83cf, 0x83d0, 0x83d1, 0x83d4, 0x83dd, 0x8353, 0x83e8, 0x83ea,
00616  0x83f6, 0x83f8, 0x83f9, 0x83fc, 0x8401, 0x8406, 0x840a, 0x840f,
00617  0x8411, 0x8415, 0x8419, 0x83ad, 0x842f, 0x8439, 0x8445, 0x8447,
00618  0x8448, 0x844a, 0x844d, 0x844f, 0x8451, 0x8452, 0x8456, 0x8458,
00619  0x8459, 0x845a, 0x845c, 0x8460, 0x8464, 0x8465, 0x8467, 0x846a,
00620  0x8470, 0x8473, 0x8474, 0x8476, 0x8478, 0x847c, 0x847d, 0x8481,
00621  0x8485, 0x8492, 0x8493, 0x8495, 0x849e, 0x84a6, 0x84a8, 0x84a9,
00622  0x84aa, 0x84af, 0x84b1, 0x84b4, 0x84ba, 0x84bd, 0x84be, 0x84c0,
00623  0x84c2, 0x84c7, 0x84c8, 0x84cc, 0x84cf, 0x84d3,
00624  /* 0x59 */
00625  0x84dc, 0x84e7, 0x84ea, 0x84ef, 0x84f0, 0x84f1, 0x84f2, 0x84f7,
00626  0x8532, 0x84fa, 0x84fb, 0x84fd, 0x8502, 0x8503, 0x8507, 0x850c,
00627  0x850e, 0x8510, 0x851c, 0x851e, 0x8522, 0x8523, 0x8524, 0x8525,
00628  0x8527, 0x852a, 0x852b, 0x852f, 0x8533, 0x8534, 0x8536, 0x853f,
00629  0x8546, 0x854f, 0x8550, 0x8551, 0x8552, 0x8553, 0x8556, 0x8559,
00630  0x855c, 0x855d, 0x855e, 0x855f, 0x8560, 0x8561, 0x8562, 0x8564,
00631  0x856b, 0x856f, 0x8579, 0x857a, 0x857b, 0x857d, 0x857f, 0x8581,
00632  0x8585, 0x8586, 0x8589, 0x858b, 0x858c, 0x858f, 0x8593, 0x8598,
00633  0x859d, 0x859f, 0x85a0, 0x85a2, 0x85a5, 0x85a7, 0x85b4, 0x85b6,
00634  0x85b7, 0x85b8, 0x85bc, 0x85bd, 0x85be, 0x85bf, 0x85c2, 0x85c7,
00635  0x85ca, 0x85cb, 0x85ce, 0x85ad, 0x85d8, 0x85da, 0x85df, 0x85e0,
00636  0x85e6, 0x85e8, 0x85ed, 0x85f3, 0x85f6, 0x85fc,
00637  /* 0x5a */
00638  0x85ff, 0x8600, 0x8604, 0x8605, 0x860d, 0x860e, 0x8610, 0x8611,
00639  0x8612, 0x8618, 0x8619, 0x861b, 0x861e, 0x8621, 0x8627, 0x8629,
00640  0x8636, 0x8638, 0x863a, 0x863c, 0x863d, 0x8640, 0x8642, 0x8646,
00641  0x8652, 0x8653, 0x8656, 0x8657, 0x8658, 0x8659, 0x865d, 0x8660,
00642  0x8661, 0x8662, 0x8663, 0x8664, 0x8669, 0x866c, 0x866f, 0x8675,
00643  0x8676, 0x8677, 0x867a, 0x868d, 0x8691, 0x8696, 0x8698, 0x869a,
00644  0x869c, 0x86a1, 0x86a6, 0x86a7, 0x86a8, 0x86ad, 0x86b1, 0x86b3,
00645  0x86b4, 0x86b5, 0x86b7, 0x86b8, 0x86b9, 0x86bf, 0x86c0, 0x86c1,
00646  0x86c3, 0x86c5, 0x86cd, 0x86d1, 0x86d2, 0x86d5, 0x86d7, 0x86da,
00647  0x86e0, 0x86e3, 0x86e5, 0x86e7, 0x86e8, 0x86fa, 0x86fc, 0x86fd,
00648  0x8704, 0x8705, 0x8707, 0x870b, 0x870e, 0x870f, 0x8710, 0x8713,
00649  0x8714, 0x8719, 0x871e, 0x871f, 0x8721, 0x8723,
00650  /* 0x5b */
00651  0x8728, 0x872e, 0x872f, 0x8731, 0x8732, 0x8739, 0x873a, 0x873c,
00652  0x873d, 0x873e, 0x8740, 0x8743, 0x8745, 0x874d, 0x8758, 0x875d,
00653  0x8761, 0x8764, 0x8765, 0x876f, 0x8771, 0x8772, 0x877b, 0x8783,
00654  0x8784, 0x8785, 0x8786, 0x8787, 0x8788, 0x8789, 0x878b, 0x878c,
00655  0x8790, 0x8793, 0x8795, 0x8797, 0x8798, 0x8799, 0x879e, 0x87a0,
00656  0x87a3, 0x87a7, 0x87ac, 0x87ad, 0x87ae, 0x87b1, 0x87b5, 0x87be,
00657  0x87bf, 0x87c1, 0x87c8, 0x87c9, 0x87ca, 0x87ce, 0x87d5, 0x87d6,
00658  0x87d9, 0x87da, 0x87dc, 0x87df, 0x87e2, 0x87e3, 0x87e4, 0x87ea,
00659  0x87eb, 0x87ed, 0x87f1, 0x87f3, 0x87f8, 0x87fa, 0x87ff, 0x8801,
00660  0x8803, 0x8806, 0x8809, 0x880a, 0x880b, 0x8810, 0x8819, 0x8812,
00661  0x8813, 0x8814, 0x8818, 0x881a, 0x881b, 0x881c, 0x881e, 0x881f,
00662  0x8828, 0x882d, 0x882e, 0x8830, 0x8832, 0x8835,
00663  /* 0x5c */
00664  0x883a, 0x883c, 0x8841, 0x8843, 0x8845, 0x8848, 0x8849, 0x884a,
00665  0x884b, 0x884e, 0x8851, 0x8855, 0x8856, 0x8858, 0x885a, 0x885c,
00666  0x885f, 0x8860, 0x8864, 0x8869, 0x8871, 0x8879, 0x887b, 0x8880,
00667  0x8898, 0x889a, 0x889b, 0x889c, 0x889f, 0x88a0, 0x88a8, 0x88aa,
00668  0x88ba, 0x88bd, 0x88be, 0x88c0, 0x88ca, 0x88cb, 0x88cc, 0x88cd,
00669  0x88ce, 0x88d1, 0x88d2, 0x88d3, 0x88db, 0x88de, 0x88e7, 0x88ef,
00670  0x88f0, 0x88f1, 0x88f5, 0x88f7, 0x8901, 0x8906, 0x890d, 0x890e,
00671  0x890f, 0x8915, 0x8916, 0x8918, 0x8919, 0x891a, 0x891c, 0x8920,
00672  0x8926, 0x8927, 0x8928, 0x8930, 0x8931, 0x8932, 0x8935, 0x8939,
00673  0x893a, 0x893e, 0x8940, 0x8942, 0x8945, 0x8946, 0x8949, 0x894f,
00674  0x8952, 0x8957, 0x895a, 0x895b, 0x895c, 0x8961, 0x8962, 0x8963,
00675  0x896b, 0x896e, 0x8970, 0x8973, 0x8975, 0x897a,
00676  /* 0x5d */
00677  0x897b, 0x897c, 0x897d, 0x8989, 0x898d, 0x8990, 0x8994, 0x8995,
00678  0x899b, 0x899c, 0x899f, 0x89a0, 0x89a5, 0x89b0, 0x89b4, 0x89b5,
00679  0x89b6, 0x89b7, 0x89bc, 0x89d4, 0x89d5, 0x89d6, 0x89d7, 0x89d8,
00680  0x89e5, 0x89e9, 0x89eb, 0x89ed, 0x89f1, 0x89f3, 0x89f6, 0x89f9,
00681  0x89fd, 0x89ff, 0x8a04, 0x8a05, 0x8a07, 0x8a0f, 0x8a11, 0x8a12,
00682  0x8a14, 0x8a15, 0x8a1e, 0x8a20, 0x8a22, 0x8a24, 0x8a26, 0x8a2b,
00683  0x8a2c, 0x8a2f, 0x8a35, 0x8a37, 0x8a3d, 0x8a3e, 0x8a40, 0x8a43,
00684  0x8a45, 0x8a47, 0x8a49, 0x8a4d, 0x8a4e, 0x8a53, 0x8a56, 0x8a57,
00685  0x8a58, 0x8a5c, 0x8a5d, 0x8a61, 0x8a65, 0x8a67, 0x8a75, 0x8a76,
00686  0x8a77, 0x8a79, 0x8a7a, 0x8a7b, 0x8a7e, 0x8a7f, 0x8a80, 0x8a83,
00687  0x8a86, 0x8a8b, 0x8a8f, 0x8a90, 0x8a92, 0x8a96, 0x8a97, 0x8a99,
00688  0x8a9f, 0x8aa7, 0x8aa9, 0x8aae, 0x8aaf, 0x8ab3,
00689  /* 0x5e */
00690  0x8ab6, 0x8ab7, 0x8abb, 0x8abe, 0x8ac3, 0x8ac6, 0x8ac8, 0x8ac9,
00691  0x8aca, 0x8ad1, 0x8ad3, 0x8ad4, 0x8ad5, 0x8ad7, 0x8add, 0x8adf,
00692  0x8aec, 0x8af0, 0x8af4, 0x8af5, 0x8af6, 0x8afc, 0x8aff, 0x8b05,
00693  0x8b06, 0x8b0b, 0x8b11, 0x8b1c, 0x8b1e, 0x8b1f, 0x8b0a, 0x8b2d,
00694  0x8b30, 0x8b37, 0x8b3c, 0x8b42, 0x8b43, 0x8b44, 0x8b45, 0x8b46,
00695  0x8b48, 0x8b52, 0x8b53, 0x8b54, 0x8b59, 0x8b4d, 0x8b56, 0x8b63,
00696  0x8b6d, 0x8b76, 0x8b78, 0x8b79, 0x8b7c, 0x8b7e, 0x8b81, 0x8b84,
00697  0x8b85, 0x8b8b, 0x8b8d, 0x8b8f, 0x8b94, 0x8b95, 0x8b9c, 0x8b9e,

```

```
00698 0x8b9f, 0x8c38, 0x8c39, 0x8c3d, 0x8c3e, 0x8c45, 0x8c47, 0x8c49,
00699 0x8c4b, 0x8c4f, 0x8c51, 0x8c53, 0x8c54, 0x8c57, 0x8c58, 0x8c5b,
00700 0x8c5d, 0x8c59, 0x8c63, 0x8c64, 0x8c66, 0x8c68, 0x8c69, 0x8c6d,
00701 0x8c73, 0x8c75, 0x8c76, 0x8c7b, 0x8c7e, 0x8c86,
00702 /* 0x5f */
00703 0x8c87, 0x8c8b, 0x8c90, 0x8c92, 0x8c93, 0x8c99, 0x8c9b, 0x8c9c,
00704 0x8ca4, 0x8cb9, 0x8cba, 0x8cc5, 0x8cc6, 0x8cc9, 0x8ccb, 0x8ccf,
00705 0x8cd6, 0x8cd5, 0x8cd9, 0x8cdd, 0x8ce1, 0x8ce8, 0x8cec, 0x8cef,
00706 0x8cf0, 0x8cf2, 0x8cf5, 0x8cf7, 0x8cf8, 0x8cfe, 0x8cff, 0x8d01,
00707 0x8d03, 0x8d09, 0x8d12, 0x8d17, 0x8d1b, 0x8d65, 0x8d69, 0x8d6c,
00708 0x8d6e, 0x8d7f, 0x8d82, 0x8d84, 0x8d88, 0x8d8d, 0x8d90, 0x8d91,
00709 0x8d95, 0x8d9e, 0x8d9f, 0x8da0, 0x8da6, 0x8dab, 0x8dac, 0x8daf,
00710 0x8db2, 0x8db5, 0x8db7, 0x8db9, 0x8dbb, 0x8dc0, 0x8dc5, 0x8dc6,
00711 0x8dc7, 0x8dc8, 0x8dca, 0x8dce, 0x8dd1, 0x8dd4, 0x8dd5, 0x8dd7,
00712 0x8dd9, 0x8de4, 0x8de5, 0x8de7, 0x8dec, 0x8df0, 0x8dbc, 0x8df1,
00713 0x8df2, 0x8df4, 0x8dfd, 0x8e01, 0x8e04, 0x8e05, 0x8e06, 0x8e0b,
00714 0x8e11, 0x8e14, 0x8e16, 0x8e20, 0x8e21, 0x8e22,
00715 /* 0x60 */
00716 0x8e23, 0x8e26, 0x8e27, 0x8e31, 0x8e33, 0x8e36, 0x8e37, 0x8e38,
00717 0x8e39, 0x8e3d, 0x8e40, 0x8e41, 0x8e4b, 0x8e4d, 0x8e4e, 0x8e4f,
00718 0x8e54, 0x8e5b, 0x8e5c, 0x8e5d, 0x8e5e, 0x8e61, 0x8e62, 0x8e69,
00719 0x8e6c, 0x8e6d, 0x8e6f, 0x8e70, 0x8e71, 0x8e79, 0x8e7a, 0x8e7b,
00720 0x8e82, 0x8e83, 0x8e89, 0x8e90, 0x8e92, 0x8e95, 0x8e9a, 0x8e9b,
00721 0x8e9d, 0x8e9e, 0x8ea2, 0x8ea7, 0x8ea9, 0x8ead, 0x8eae, 0x8eb3,
00722 0x8eb5, 0x8eba, 0x8ebb, 0x8ec0, 0x8ec1, 0x8ec3, 0x8ec4, 0x8ec7,
00723 0x8ecf, 0x8ed1, 0x8ed4, 0x8edc, 0x8ee8, 0x8eee, 0x8ef0, 0x8ef1,
00724 0x8ef7, 0x8ef9, 0x8efa, 0x8eed, 0x8f00, 0x8f02, 0x8f07, 0x8f08,
00725 0x8f0f, 0x8f10, 0x8f16, 0x8f17, 0x8f18, 0x8f1e, 0x8f20, 0x8f21,
00726 0x8f23, 0x8f25, 0x8f27, 0x8f28, 0x8f2c, 0x8f2d, 0x8f2e, 0x8f34,
00727 0x8f35, 0x8f36, 0x8f37, 0x8f3a, 0x8f40, 0x8f41,
00728 /* 0x61 */
00729 0x8f43, 0x8f47, 0x8f4f, 0x8f51, 0x8f52, 0x8f53, 0x8f54, 0x8f55,
00730 0x8f58, 0x8f5d, 0x8f5e, 0x8f65, 0x8f6d, 0x8fa0, 0x8fa1, 0x8fa4,
00731 0x8fa5, 0x8fa6, 0x8fb5, 0x8fb6, 0x8fb8, 0x8fbe, 0x8fc0, 0x8fc1,
00732 0x8fc6, 0x8fca, 0x8fcb, 0x8fcd, 0x8fd0, 0x8fd2, 0x8fd3, 0x8fd5,
00733 0x8fe0, 0x8fe3, 0x8fe4, 0x8fee, 0x8ff1, 0x8ff5, 0x8ff6,
00734 0x8ffb, 0x8ffe, 0x9002, 0x9004, 0x9008, 0x900c, 0x9018, 0x901b,
00735 0x9028, 0x9029, 0x902f, 0x902a, 0x902c, 0x902d, 0x9033, 0x9034,
00736 0x9037, 0x903f, 0x9043, 0x9044, 0x904c, 0x905b, 0x905d, 0x9062,
00737 0x9066, 0x9067, 0x906c, 0x9070, 0x9074, 0x9079, 0x9085, 0x9088,
00738 0x908b, 0x908c, 0x908e, 0x9090, 0x9095, 0x9097, 0x9098, 0x9099,
00739 0x909b, 0x90a0, 0x90a1, 0x90a2, 0x90a5, 0x90b0, 0x90b2, 0x90b3,
00740 0x90b4, 0x90b6, 0x90bd, 0x90cc, 0x90be, 0x90c3,
00741 /* 0x62 */
00742 0x90c4, 0x90c5, 0x90c7, 0x90c8, 0x90d5, 0x90d7, 0x90d8, 0x90d9,
00743 0x90dc, 0x90dd, 0x90df, 0x90e5, 0x90d2, 0x90f6, 0x90eb, 0x90ef,
00744 0x90f0, 0x90f4, 0x90fe, 0x90ff, 0x9100, 0x9104, 0x9105, 0x9106,
00745 0x9108, 0x910d, 0x9110, 0x9114, 0x9116, 0x9117, 0x9118, 0x911a,
00746 0x911c, 0x911e, 0x9120, 0x9125, 0x9122, 0x9123, 0x9127, 0x9129,
00747 0x912e, 0x912f, 0x9131, 0x9134, 0x9136, 0x9137, 0x9139, 0x913a,
00748 0x913c, 0x913d, 0x9143, 0x9147, 0x9148, 0x914f, 0x9153, 0x9157,
00749 0x9159, 0x915a, 0x915b, 0x9161, 0x9164, 0x9167, 0x916d, 0x9174,
00750 0x9179, 0x917a, 0x917b, 0x9181, 0x9183, 0x9185, 0x9186, 0x918a,
00751 0x918e, 0x9191, 0x9193, 0x9194, 0x9195, 0x9198, 0x919e, 0x91a1,
00752 0x91a6, 0x91a8, 0x91ac, 0x91ad, 0x91ae, 0x91b0, 0x91b1, 0x91b2,
00753 0x91b3, 0x91b6, 0x91bb, 0x91bc, 0x91bd, 0x91bf,
00754 /* 0x63 */
00755 0x91c2, 0x91c3, 0x91c5, 0x91d3, 0x91d4, 0x91d7, 0x91d9, 0x91da,
00756 0x91de, 0x91e4, 0x91e5, 0x91e9, 0x91ea, 0x91ec, 0x91ed, 0x91ee,
00757 0x91ef, 0x91f0, 0x91f1, 0x91f7, 0x91f9, 0x91fb, 0x91fd, 0x9200,
00758 0x9201, 0x9204, 0x9205, 0x9206, 0x9207, 0x9209, 0x920a, 0x920c,
00759 0x9210, 0x9212, 0x9213, 0x9216, 0x9218, 0x921c, 0x921d, 0x9223,
00760 0x9224, 0x9225, 0x9226, 0x9228, 0x922e, 0x922f, 0x9230, 0x9233,
00761 0x9235, 0x9236, 0x9238, 0x9239, 0x923a, 0x923c, 0x923e, 0x9240,
00762 0x9242, 0x9243, 0x9246, 0x9247, 0x924a, 0x924d, 0x924e, 0x924f,
00763 0x9251, 0x9258, 0x9259, 0x925c, 0x925d, 0x9260, 0x9261, 0x9265,
00764 0x9267, 0x9268, 0x9269, 0x926e, 0x926f, 0x9270, 0x9275, 0x9276,
00765 0x9277, 0x9278, 0x9279, 0x927b, 0x927c, 0x927d, 0x927f, 0x9288,
00766 0x9289, 0x928a, 0x928d, 0x928e, 0x9292, 0x9297,
00767 /* 0x64 */
00768 0x9299, 0x929f, 0x92a0, 0x92a4, 0x92a5, 0x92a7, 0x92a8, 0x92ab,
00769 0x92af, 0x92b2, 0x92b6, 0x92b8, 0x92ba, 0x92bb, 0x92bc, 0x92bd,
00770 0x92bf, 0x92c0, 0x92c1, 0x92c2, 0x92c3, 0x92c5, 0x92c6, 0x92c7,
00771 0x92c8, 0x92cb, 0x92cc, 0x92cd, 0x92ce, 0x92d0, 0x92d3, 0x92d5,
00772 0x92d7, 0x92d8, 0x92d9, 0x92dc, 0x92dd, 0x92df, 0x92e0, 0x92e1,
00773 0x92e3, 0x92e5, 0x92e7, 0x92e8, 0x92ec, 0x92ee, 0x92f0, 0x92f9,
00774 0x92fb, 0x92ff, 0x9300, 0x9302, 0x9308, 0x930d, 0x9311, 0x9314,
00775 0x9315, 0x931c, 0x931d, 0x931e, 0x931f, 0x9321, 0x9324, 0x9325,
00776 0x9327, 0x9329, 0x932a, 0x9333, 0x9334, 0x9336, 0x9337, 0x9347,
00777 0x9348, 0x9349, 0x9350, 0x9351, 0x9352, 0x9355, 0x9357, 0x9358,
00778 0x935a, 0x935e, 0x9364, 0x9365, 0x9367, 0x9369, 0x936a, 0x936d,
00779 0x936f, 0x9370, 0x9371, 0x9373, 0x9374, 0x9376,
00780 /* 0x65 */
00781 0x937a, 0x937d, 0x937f, 0x9380, 0x9381, 0x9382, 0x9388, 0x938a,
00782 0x938b, 0x938d, 0x938f, 0x9392, 0x9395, 0x9398, 0x939b, 0x939e,
00783 0x93a1, 0x93a3, 0x93a4, 0x93a6, 0x93a8, 0x93ab, 0x93b4, 0x93b5,
00784 0x93b6, 0x93ba, 0x93a9, 0x93c1, 0x93c4, 0x93c5, 0x93c6, 0x93c7,
```

```

00785 0x93c9, 0x93ca, 0x93cb, 0x93cc, 0x93cd, 0x93d3, 0x93d9, 0x93dc,
00786 0x93de, 0x93df, 0x93e2, 0x93e6, 0x93e7, 0x93f9, 0x93f7, 0x93f8,
00787 0x93fa, 0x93fb, 0x93fd, 0x9401, 0x9402, 0x9404, 0x9408, 0x9409,
00788 0x940d, 0x940e, 0x940f, 0x9415, 0x9416, 0x9417, 0x941f, 0x942e,
00789 0x942f, 0x9431, 0x9432, 0x9433, 0x9434, 0x943b, 0x943c, 0x943d,
00790 0x9443, 0x9445, 0x9448, 0x944a, 0x944c, 0x9455, 0x9459, 0x945c,
00791 0x945f, 0x9461, 0x9463, 0x9468, 0x946b, 0x946d, 0x946e, 0x946f,
00792 0x9471, 0x9472, 0x9484, 0x9483, 0x9578, 0x9579,
00793 /* 0x66 */
00794 0x957e, 0x9584, 0x9588, 0x958c, 0x958d, 0x958e, 0x959d, 0x959e,
00795 0x959f, 0x95a1, 0x95a6, 0x95a9, 0x95ab, 0x95ac, 0x95b4, 0x95b6,
00796 0x95ba, 0x95bd, 0x95bf, 0x95c6, 0x95c8, 0x95c9, 0x95cb, 0x95d0,
00797 0x95d1, 0x95d2, 0x95d3, 0x95d9, 0x95da, 0x95dd, 0x95de, 0x95df,
00798 0x95e0, 0x95e4, 0x95e6, 0x961d, 0x961e, 0x9622, 0x9624, 0x9625,
00799 0x9626, 0x962c, 0x9631, 0x9633, 0x9637, 0x9638, 0x9639, 0x963a,
00800 0x963c, 0x963d, 0x9641, 0x9652, 0x9654, 0x9656, 0x9657, 0x9658,
00801 0x9661, 0x966e, 0x9674, 0x967b, 0x967c, 0x967e, 0x967f, 0x9681,
00802 0x9682, 0x9683, 0x9684, 0x9688, 0x9689, 0x9691, 0x9696, 0x969a,
00803 0x969f, 0x96a4, 0x96a5, 0x96a6, 0x96a9, 0x96ae, 0x96af, 0x96b3,
00804 0x96ba, 0x96ca, 0x96d2, 0x5db2, 0x96d8, 0x96da, 0x96dd, 0x96de,
00805 0x96df, 0x96e9, 0x96ef, 0x96f1, 0x96fa, 0x9702,
00806 /* 0x67 */
00807 0x9703, 0x9705, 0x9709, 0x971a, 0x971b, 0x971d, 0x9721, 0x9722,
00808 0x9723, 0x9728, 0x9731, 0x9733, 0x9741, 0x9743, 0x974a, 0x974e,
00809 0x974f, 0x9755, 0x9757, 0x9758, 0x975a, 0x975b, 0x9763, 0x9767,
00810 0x976a, 0x976e, 0x9773, 0x9776, 0x9777, 0x9778, 0x977b, 0x977d,
00811 0x977f, 0x9780, 0x9789, 0x9795, 0x9796, 0x9797, 0x9799, 0x979a,
00812 0x979e, 0x979f, 0x97a2, 0x97ac, 0x97ae, 0x97b1, 0x97b2, 0x97b5,
00813 0x97b6, 0x97b8, 0x97b9, 0x97ba, 0x97bc, 0x97be, 0x97bf, 0x97c1,
00814 0x97c4, 0x97c5, 0x97c7, 0x97c9, 0x97ca, 0x97cc, 0x97cd, 0x97ce,
00815 0x97d0, 0x97d1, 0x97d4, 0x97d7, 0x97d8, 0x97d9, 0x97dd, 0x97de,
00816 0x97e0, 0x97db, 0x97e1, 0x97e4, 0x97ef, 0x97f1, 0x97f4, 0x97f7,
00817 0x97f8, 0x97fa, 0x9807, 0x980a, 0x9819, 0x980d, 0x980e, 0x9814,
00818 0x9816, 0x981c, 0x981e, 0x9820, 0x9823, 0x9826,
00819 /* 0x68 */
00820 0x982b, 0x982e, 0x982f, 0x9830, 0x9832, 0x9833, 0x9835, 0x9825,
00821 0x983e, 0x9844, 0x9847, 0x984a, 0x9851, 0x9852, 0x9853, 0x9856,
00822 0x9857, 0x9859, 0x985a, 0x9862, 0x9863, 0x9865, 0x9866, 0x986a,
00823 0x986c, 0x98ab, 0x98ad, 0x98ae, 0x98b0, 0x98b4, 0x98b7, 0x98b8,
00824 0x98ba, 0x98bb, 0x98bf, 0x98c2, 0x98c5, 0x98c8, 0x98cc, 0x98e1,
00825 0x98e3, 0x98e5, 0x98e6, 0x98e7, 0x98ea, 0x98f3, 0x98f6, 0x9902,
00826 0x9907, 0x9908, 0x9911, 0x9915, 0x9916, 0x9917, 0x991a, 0x991b,
00827 0x991c, 0x991f, 0x9922, 0x9926, 0x9927, 0x992b, 0x9931, 0x9932,
00828 0x9933, 0x9934, 0x9935, 0x9939, 0x993a, 0x993b, 0x993c, 0x9940,
00829 0x9941, 0x9946, 0x9947, 0x9948, 0x994d, 0x994e, 0x9954, 0x9958,
00830 0x9959, 0x995b, 0x995c, 0x995e, 0x995f, 0x9960, 0x999b, 0x999d,
00831 0x999f, 0x99a6, 0x99b0, 0x99b1, 0x99b2, 0x99b5,
00832 /* 0x69 */
00833 0x99b9, 0x99ba, 0x99bd, 0x99bf, 0x99c3, 0x99c9, 0x99d3, 0x99d4,
00834 0x99d9, 0x99da, 0x99dc, 0x99de, 0x99e7, 0x99ea, 0x99eb, 0x99ec,
00835 0x99f0, 0x99f4, 0x99f5, 0x99f9, 0x99fd, 0x99fe, 0x9a02, 0x9a03,
00836 0x9a04, 0x9a0b, 0x9a0c, 0x9a10, 0x9a11, 0x9a16, 0x9a1e, 0x9a20,
00837 0x9a22, 0x9a23, 0x9a24, 0x9a27, 0x9a2d, 0x9a2e, 0x9a33, 0x9a35,
00838 0x9a36, 0x9a38, 0x9a47, 0x9a41, 0x9a44, 0x9a4a, 0x9a4b, 0x9a4c,
00839 0x9a4e, 0x9a51, 0x9a54, 0x9a56, 0x9a5d, 0x9aaa, 0x9aac, 0x9aae,
00840 0x9aaf, 0x9ab2, 0x9ab4, 0x9ab5, 0x9ab6, 0x9ab9, 0x9abb, 0x9abe,
00841 0x9abf, 0x9ac1, 0x9ac3, 0x9ac6, 0x9ac8, 0x9ace, 0x9ad0, 0x9ad2,
00842 0x9ad5, 0x9ad6, 0x9ad7, 0x9adb, 0x9adc, 0x9ae0, 0x9ae4, 0x9ae5,
00843 0x9ae7, 0x9ae9, 0x9aec, 0x9af2, 0x9af3, 0x9af5, 0x9af9, 0x9afa,
00844 0x9afd, 0x9aff, 0x9b00, 0x9b01, 0x9b02, 0x9b03,
00845 /* 0x6a */
00846 0x9b04, 0x9b05, 0x9b08, 0x9b09, 0x9b0b, 0x9b0c, 0x9b0d, 0x9b0e,
00847 0x9b10, 0x9b12, 0x9b16, 0x9b19, 0x9b1b, 0x9b1c, 0x9b20, 0x9b26,
00848 0x9b2b, 0x9b2d, 0x9b33, 0x9b34, 0x9b35, 0x9b37, 0x9b39, 0x9b3a,
00849 0x9b3d, 0x9b48, 0x9b4b, 0x9b4c, 0x9b55, 0x9b56, 0x9b57, 0x9b5b,
00850 0x9b5e, 0x9b61, 0x9b63, 0x9b65, 0x9b66, 0x9b68, 0x9b6a, 0x9b6b,
00851 0x9b6c, 0x9b6d, 0x9b6e, 0x9b73, 0x9b75, 0x9b77, 0x9b78, 0x9b79,
00852 0x9b7f, 0x9b80, 0x9b84, 0x9b85, 0x9b86, 0x9b87, 0x9b89, 0x9b8a,
00853 0x9b8b, 0x9b8d, 0x9b8f, 0x9b90, 0x9b94, 0x9b9a, 0x9b9d, 0x9b9e,
00854 0x9ba6, 0x9ba7, 0x9ba9, 0x9bac, 0x9bb0, 0x9bb1, 0x9bb2, 0x9bb7,
00855 0x9bb8, 0x9bbb, 0x9bbc, 0x9bbf, 0x9bc1, 0x9bc7, 0x9bc8, 0x9bc9,
00856 0x9bce, 0x9bd0, 0x9bd7, 0x9bdd, 0x9bdf, 0x9be5, 0x9be7,
00857 0x9bea, 0x9beb, 0x9bef, 0x9bf3, 0x9bf7, 0x9bf8,
00858 /* 0x6b */
00859 0x9bf9, 0x9bfa, 0x9bfd, 0x9bff, 0x9c00, 0x9c02, 0x9c0b, 0x9c0f,
00860 0x9c11, 0x9c16, 0x9c18, 0x9c19, 0x9c1a, 0x9c1c, 0x9c1e, 0x9c22,
00861 0x9c23, 0x9c26, 0x9c27, 0x9c28, 0x9c29, 0x9c2a, 0x9c31, 0x9c35,
00862 0x9c36, 0x9c37, 0x9c3d, 0x9c41, 0x9c43, 0x9c44, 0x9c45, 0x9c49,
00863 0x9c4a, 0x9c4e, 0x9c4f, 0x9c50, 0x9c53, 0x9c54, 0x9c56, 0x9c58,
00864 0x9c5b, 0x9c5d, 0x9c5e, 0x9c5f, 0x9c63, 0x9c69, 0x9c6a, 0x9c5c,
00865 0x9c6b, 0x9c68, 0x9c6e, 0x9c70, 0x9c72, 0x9c75, 0x9c77, 0x9c7b,
00866 0x9c6c, 0x9cf2, 0x9cf7, 0x9cf9, 0x9d0b, 0x9d02, 0x9d11, 0x9d17,
00867 0x9d18, 0x9d1c, 0x9d1d, 0x9d1e, 0x9d2f, 0x9d30, 0x9d32, 0x9d33,
00868 0x9d34, 0x9d3a, 0x9d3c, 0x9d45, 0x9d3d, 0x9d42, 0x9d43, 0x9d47,
00869 0x9d4a, 0x9d53, 0x9d54, 0x9d5f, 0x9d63, 0x9d62, 0x9d65, 0x9d69,
00870 0x9d6a, 0x9d6b, 0x9d70, 0x9d76, 0x9d77, 0x9d7b,
00871 /* 0x6c */

```

```
00872 0x9d7c, 0x9d7e, 0x9d83, 0x9d84, 0x9d86, 0x9d8a, 0x9d8d, 0x9d8e,
00873 0x9d92, 0x9d93, 0x9d95, 0x9d96, 0x9d97, 0x9d98, 0x9da1, 0x9daa,
00874 0x9dac, 0x9dae, 0x9db1, 0x9db5, 0x9db9, 0x9dbc, 0x9dbf, 0x9dc3,
00875 0x9dc7, 0x9dc9, 0x9dca, 0x9dd4, 0x9dd5, 0x9dd6, 0x9dd7, 0x9dda,
00876 0x9dde, 0x9ddf, 0x9de0, 0x9de5, 0x9de7, 0x9de9, 0x9deb, 0x9dee,
00877 0x9df0, 0x9df3, 0x9df4, 0x9dfe, 0x9e0a, 0x9e02, 0x9e07, 0x9e0e,
00878 0x9e10, 0x9e11, 0x9e12, 0x9e15, 0x9e16, 0x9e19, 0x9e1c, 0x9e1d,
00879 0x9e7a, 0x9e7b, 0x9e7c, 0x9e80, 0x9e82, 0x9e83, 0x9e84, 0x9e85,
00880 0x9e87, 0x9e8e, 0x9e8f, 0x9e96, 0x9e98, 0x9e9b, 0x9e9e, 0x9ea4,
00881 0x9ea8, 0x9eac, 0x9eae, 0x9eaf, 0x9eb0, 0x9eb3, 0x9eb4, 0x9eb5,
00882 0x9ec6, 0x9ec8, 0x9ecb, 0x9ed5, 0x9edf, 0x9ee4, 0x9ee7, 0x9eec,
00883 0x9eed, 0x9eee, 0x9ef0, 0x9ef1, 0x9ef2, 0x9ef5,
00884 /* 0x6d */
00885 0x9ef8, 0x9eff, 0x9f02, 0x9f03, 0x9f09, 0x9f0f, 0x9f10, 0x9f11,
00886 0x9f12, 0x9f14, 0x9f16, 0x9f17, 0x9f19, 0x9f1a, 0x9f1b, 0x9f1f,
00887 0x9f22, 0x9f26, 0x9f2a, 0x9f2b, 0x9f2f, 0x9f31, 0x9f32, 0x9f34,
00888 0x9f37, 0x9f39, 0x9f3a, 0x9f3c, 0x9f3d, 0x9f3f, 0x9f41, 0x9f43,
00889 0x9f44, 0x9f45, 0x9f46, 0x9f47, 0x9f53, 0x9f55, 0x9f56, 0x9f57,
00890 0x9f58, 0x9f5a, 0x9f5d, 0x9f5e, 0x9f68, 0x9f69, 0x9f6d, 0x9f6e,
00891 0x9f6f, 0x9f70, 0x9f71, 0x9f73, 0x9f75, 0x9f7a, 0x9f7d, 0x9f8f,
00892 0x9f90, 0x9f91, 0x9f92, 0x9f94, 0x9f96, 0x9f97, 0x9f9e, 0x9fa1,
00893 0x9fa2, 0x9fa3, 0x9fa5,
00894 };
00895
00896 static int
00897 jisx0212_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00898 {
00899     unsigned char c1 = (s[0] & 0x7F);
00900     if ((c1 == 0x22) || (c1 >= 0x26 && c1 <= 0x27) || (c1 >= 0x29 && c1 <= 0x2b) || (c1 >= 0x30 && c1 <=
00901         0x6d)) {
00902         if (n >= 2) {
00903             unsigned char c2 = (s[1] & 0x7F);
00904             if (c2 >= 0x21 && c2 < 0x7f) {
00905                 unsigned int i = 94 * (c1 - 0x21) + (c2 - 0x21);
00906                 unsigned short wc = 0xffff;
00907                 if (i < 470) {
00908                     if (i < 175)
00909                         wc = jisx0212_2uni_page22[i-94];
00910                     } else if (i < 752) {
00911                         if (i < 658)
00912                             wc = jisx0212_2uni_page26[i-470];
00913                     } else if (i < 1410) {
00914                         if (i < 1027)
00915                             wc = jisx0212_2uni_page29[i-752];
00916                     } else {
00917                         if (i < 7211)
00918                             wc = jisx0212_2uni_page30[i-1410];
00919                     }
00920                     if (wc != 0xffff) {
00921                         *pwc = (ucs4_t) wc;
00922                         return 2;
00923                     }
00924                 }
00925                 return RET_ILSEQ;
00926             }
00927             return RET_TOOFEW(0);
00928         }
00929     }
00930     #endif /* NEED_TOWC */
00931
00932     #ifndef NEED_TOMB
00933     static const unsigned short jisx0212_2charset[6067] = {
00934         0x2237, 0x2242, 0x2270, 0x2243, 0x226d, 0x226c, 0x226e, 0x2234,
00935         0x2231, 0x226b, 0x224e, 0x2a22, 0x2a21, 0x2a24, 0x2a2a, 0x2a23,
00936         0x2a29, 0x2921, 0x2a2e, 0x2a32, 0x2a31, 0x2a34, 0x2a33, 0x2a40,
00937         0x2a3f, 0x2a42, 0x2a41, 0x2a50, 0x2a52, 0x2a51, 0x2a54, 0x2a58,
00938         0x2a53, 0x292c, 0x2a63, 0x2a62, 0x2a65, 0x2a64, 0x2a72, 0x2930,
00939         0x294e, 0x2b22, 0x2b21, 0x2b24, 0x2b23, 0x2b29, 0x2941,
00940         0x2b2e, 0x2b32, 0x2b31, 0x2b34, 0x2b33, 0x2b40, 0x2b3f, 0x2b42,
00941         0x2b41, 0x2943, 0x2b50, 0x2b52, 0x2b51, 0x2b54, 0x2b58, 0x2b53,
00942         0x294c, 0x2b63, 0x2b62, 0x2b65, 0x2b64, 0x2b72, 0x2950, 0x2b73,
00943         0x2a27, 0x2b27, 0x2a25, 0x2b25, 0x2a28, 0x2b28, 0x2a2b, 0x2b2b,
00944         0x2a2c, 0x2b2c, 0x2a2f, 0x2b2f, 0x2a2d, 0x2b2d, 0x2a30, 0x2b30,
00945         0x2922, 0x2942, 0x2a37, 0x2b37, 0x2a36, 0x2b36, 0x2a38, 0x2b38,
00946         0x2a35, 0x2b35, 0x2a3a, 0x2b3a, 0x2a3b, 0x2b3b, 0x2a3d, 0x2b3d,
00947         0x2a3c, 0x2a3e, 0x2b3e, 0x2924, 0x2944, 0x2a47, 0x2b47, 0x2a45,
00948         0x2b45, 0x2a46, 0x2b46, 0x2a44, 0x2945, 0x2926, 0x2946, 0x2a48,
00949         0x2b48, 0x2a49, 0x2b49, 0x2947, 0x2a4a, 0x2b4a, 0x2a4c, 0x2b4c,
00950         0x2a4b, 0x2b4b, 0x2929, 0x2949, 0x2928, 0x2948, 0x2a4d, 0x2b4d,
00951         0x2a4f, 0x2b4f, 0x2a4e, 0x2b4e, 0x294a, 0x292b, 0x294b, 0x2a57,
00952         0x2b57, 0x2a56, 0x2b56, 0x292d, 0x294d, 0x2a59, 0x2b59, 0x2a5b,
00953         0x2b5b, 0x2a5a, 0x2b5a, 0x2a5c, 0x2b5c, 0x2a5d, 0x2b5d, 0x2a5f,
00954         0x2b5f, 0x2a5e, 0x2b5e, 0x2a61, 0x2b61, 0x2a60, 0x2b60, 0x292f,
00955         0x294f, 0x2a6c, 0x2b6c, 0x2a69, 0x2b69, 0x2a66, 0x2b66, 0x2a6b,
00956         0x2b6b, 0x2a68, 0x2b68, 0x2a6a, 0x2b6a, 0x2a71, 0x2b71, 0x2a74,
00957         0x2b74, 0x2a73, 0x2a75, 0x2b75, 0x2a77, 0x2b77, 0x2a76, 0x2b76,
```

```
00958 0x2a26, 0x2b26, 0x2a43, 0x2b43, 0x2a55, 0x2b55, 0x2a67, 0x2b67,
00959 0x2a70, 0x2b70, 0x2a6d, 0x2b6d, 0x2a6f, 0x2b6f, 0x2a6e, 0x2b6e,
00960 0x2b39, 0x2230, 0x222f, 0x2232, 0x2236, 0x2235, 0x2233, 0x2238,
00961 0x2239, 0x2661, 0x2662, 0x2663, 0x2664, 0x2667, 0x2669, 0x266c,
00962 0x2676, 0x2665, 0x266a, 0x2671, 0x2672, 0x2673, 0x2674, 0x267b,
00963 0x2678, 0x2675, 0x267a, 0x2677, 0x2679, 0x267c, 0x2742, 0x2743,
00964 0x2744, 0x2745, 0x2746, 0x2747, 0x2748, 0x2749, 0x274a, 0x274b,
00965 0x274c, 0x274d, 0x274e, 0x2772, 0x2773, 0x2774, 0x2775, 0x2776,
00966 0x2777, 0x2778, 0x2779, 0x277a, 0x277b, 0x277c, 0x277d, 0x277e,
00967 0x2271, 0x226f, 0x3021, 0x3022, 0x3023, 0x3024, 0x3025, 0x3026,
00968 0x3027, 0x3028, 0x3029, 0x302a, 0x302b, 0x302c, 0x302d, 0x302e,
00969 0x302f, 0x3030, 0x3031, 0x3032, 0x3033, 0x3034, 0x3035, 0x3036,
00970 0x3037, 0x3038, 0x3039, 0x303a, 0x303b, 0x303c, 0x303d, 0x303e,
00971 0x303f, 0x3040, 0x3041, 0x3042, 0x3043, 0x3044, 0x3045, 0x3046,
00972 0x3047, 0x3048, 0x3049, 0x304a, 0x304b, 0x304c, 0x304d, 0x304e,
00973 0x304f, 0x3050, 0x3051, 0x3052, 0x3053, 0x3054, 0x3055, 0x3056,
00974 0x3057, 0x3058, 0x3059, 0x305a, 0x305b, 0x305c, 0x305d, 0x305e,
00975 0x3060, 0x3061, 0x3062, 0x3063, 0x3064, 0x3065, 0x3066, 0x3067,
00976 0x3068, 0x3069, 0x306a, 0x306b, 0x306c, 0x306d, 0x306e, 0x306f,
00977 0x3070, 0x305f, 0x3071, 0x3072, 0x3073, 0x3074, 0x3075, 0x3076,
00978 0x3077, 0x3078, 0x3079, 0x307a, 0x307b, 0x307c, 0x307d, 0x307e,
00979 0x3121, 0x3122, 0x3123, 0x3124, 0x3125, 0x3126, 0x3127, 0x3128,
00980 0x3129, 0x312a, 0x312b, 0x312c, 0x312d, 0x312e, 0x312f, 0x3130,
00981 0x3131, 0x3132, 0x3133, 0x3134, 0x3135, 0x3136, 0x3137, 0x3138,
00982 0x3139, 0x313a, 0x313b, 0x313c, 0x313d, 0x313e, 0x313f, 0x3140,
00983 0x3141, 0x3142, 0x3143, 0x3144, 0x3145, 0x3146, 0x3147, 0x3148,
00984 0x3149, 0x314a, 0x314b, 0x314c, 0x314d, 0x314e, 0x314f, 0x3150,
00985 0x3151, 0x3152, 0x3153, 0x3154, 0x3155, 0x3156, 0x3157, 0x3158,
00986 0x3159, 0x315a, 0x315b, 0x315c, 0x315d, 0x315e, 0x315f, 0x3160,
00987 0x3161, 0x3162, 0x3163, 0x3164, 0x3165, 0x3166, 0x3167,
00988 0x3168, 0x3169, 0x316a, 0x316b, 0x316c, 0x316d, 0x316e, 0x316f,
00989 0x3170, 0x3171, 0x3172, 0x3173, 0x3174, 0x3175, 0x3176, 0x3177,
00990 0x3178, 0x3179, 0x317a, 0x317b, 0x317c, 0x317d, 0x317e, 0x3221,
00991 0x3222, 0x3223, 0x3224, 0x3225, 0x3226, 0x3227, 0x3228, 0x3229,
00992 0x322a, 0x322b, 0x322c, 0x322d, 0x322e, 0x322f, 0x3230, 0x3231,
00993 0x3232, 0x3233, 0x3234, 0x3235, 0x3236, 0x3237, 0x3238, 0x3239,
00994 0x323a, 0x323b, 0x323c, 0x323d, 0x323e, 0x323f, 0x3240, 0x3241,
00995 0x3242, 0x3243, 0x3244, 0x3245, 0x3246, 0x3247, 0x3248, 0x3249,
00996 0x324a, 0x324b, 0x324c, 0x324d, 0x324e, 0x324f, 0x3250, 0x3251,
00997 0x3252, 0x3253, 0x3254, 0x3255, 0x3256, 0x3257, 0x3258, 0x3259,
00998 0x325a, 0x325b, 0x325c, 0x325d, 0x325e, 0x325f, 0x3260, 0x3261,
00999 0x3262, 0x3263, 0x3264, 0x3265, 0x3266, 0x3267, 0x3268, 0x3269,
01000 0x326a, 0x326b, 0x326c, 0x326d, 0x326e, 0x326f, 0x3270, 0x3271,
01001 0x3272, 0x3273, 0x3274, 0x3275, 0x3276, 0x3277, 0x3278, 0x3279,
01002 0x327a, 0x327b, 0x327c, 0x327d, 0x327e, 0x327f, 0x3280, 0x3281,
01003 0x3282, 0x3283, 0x3284, 0x3285, 0x3286, 0x3287, 0x3288, 0x3289,
01004 0x328a, 0x328b, 0x328c, 0x328d, 0x328e, 0x328f, 0x3290, 0x3291,
01005 0x3292, 0x3293, 0x3294, 0x3295, 0x3296, 0x3297, 0x3298, 0x3299,
01006 0x329a, 0x329b, 0x329c, 0x329d, 0x329e, 0x329f, 0x3300, 0x3301,
01007 0x3302, 0x3303, 0x3304, 0x3305, 0x3306, 0x3307, 0x3308, 0x3309,
01008 0x330a, 0x330b, 0x330c, 0x330d, 0x330e, 0x330f, 0x3310, 0x3311,
01009 0x3312, 0x3313, 0x3314, 0x3315, 0x3316, 0x3317, 0x3318, 0x3319,
01010 0x331a, 0x331b, 0x331c, 0x331d, 0x331e, 0x331f, 0x3320, 0x3321,
01011 0x3322, 0x3323, 0x3324, 0x3325, 0x3326, 0x3327, 0x3328, 0x3329,
01012 0x332a, 0x332b, 0x332c, 0x332d, 0x332e, 0x332f, 0x3330, 0x3331,
01013 0x3332, 0x3333, 0x3334, 0x3335, 0x3336, 0x3337, 0x3338, 0x3339,
01014 0x333a, 0x333b, 0x333c, 0x333d, 0x333e, 0x333f, 0x3340, 0x3341,
01015 0x3342, 0x3343, 0x3344, 0x3345, 0x3346, 0x3347, 0x3348, 0x3349,
01016 0x334a, 0x334b, 0x334c, 0x334d, 0x334e, 0x334f, 0x3350, 0x3351,
01017 0x3352, 0x3353, 0x3354, 0x3355, 0x3356, 0x3357, 0x3358, 0x3359,
01018 0x335a, 0x335b, 0x335c, 0x335d, 0x335e, 0x335f, 0x3360, 0x3361,
01019 0x3362, 0x3363, 0x3364, 0x3365, 0x3366, 0x3367, 0x3368, 0x3369,
01020 0x336a, 0x336b, 0x336c, 0x336d, 0x336e, 0x336f, 0x3370, 0x3371,
01021 0x3372, 0x3373, 0x3374, 0x3375, 0x3376, 0x3377, 0x3378, 0x3379,
01022 0x337a, 0x337b, 0x337c, 0x337d, 0x337e, 0x337f, 0x3380, 0x3381,
01023 0x3382, 0x3383, 0x3384, 0x3385, 0x3386, 0x3387, 0x3388, 0x3389,
01024 0x338a, 0x338b, 0x338c, 0x338d, 0x338e, 0x338f, 0x3390, 0x3391,
01025 0x3392, 0x3393, 0x3394, 0x3395, 0x3396, 0x3397, 0x3398, 0x3399,
01026 0x339a, 0x339b, 0x339c, 0x339d, 0x339e, 0x339f, 0x3400, 0x3401,
01027 0x3402, 0x3403, 0x3404, 0x3405, 0x3406, 0x3407, 0x3408, 0x3409,
01028 0x340a, 0x340b, 0x340c, 0x340d, 0x340e, 0x340f, 0x3410, 0x3411,
01029 0x3412, 0x3413, 0x3414, 0x3415, 0x3416, 0x3417, 0x3418, 0x3419,
01030 0x341a, 0x341b, 0x341c, 0x341d, 0x341e, 0x341f, 0x3420, 0x3421,
01031 0x3422, 0x3423, 0x3424, 0x3425, 0x3426, 0x3427, 0x3428, 0x3429,
01032 0x342a, 0x342b, 0x342c, 0x342d, 0x342e, 0x342f, 0x3430, 0x3431,
01033 0x3432, 0x3433, 0x3434, 0x3435, 0x3436, 0x3437, 0x3438, 0x3439,
01034 0x343a, 0x343b, 0x343c, 0x343d, 0x343e, 0x343f, 0x3440, 0x3441,
01035 0x3442, 0x3443, 0x3444, 0x3445, 0x3446, 0x3447, 0x3448, 0x3449,
01036 0x344a, 0x344b, 0x344c, 0x344d, 0x344e, 0x344f, 0x3450, 0x3451,
01037 0x3452, 0x3453, 0x3454, 0x3455, 0x3456, 0x3457, 0x3458, 0x3459,
01038 0x345a, 0x345b, 0x345c, 0x345d, 0x345e, 0x345f, 0x3460, 0x3461,
01039 0x3462, 0x3463, 0x3464, 0x3465, 0x3466, 0x3467, 0x3468, 0x3469,
01040 0x346a, 0x346b, 0x346c, 0x346d, 0x346e, 0x346f, 0x3470, 0x3471,
01041 0x3472, 0x3473, 0x3474, 0x3475, 0x3476, 0x3477, 0x3478, 0x3479,
01042 0x347a, 0x347b, 0x347c, 0x347d, 0x347e, 0x347f, 0x3480, 0x3481,
01043 0x3482, 0x3483, 0x3484, 0x3485, 0x3486, 0x3487, 0x3488, 0x3489,
01044 0x348a, 0x348b, 0x348c, 0x348d, 0x348e, 0x348f, 0x3490, 0x3491,
```


01045 0x365b, 0x365c, 0x365d, 0x365e, 0x365f, 0x3660, 0x3661, 0x3662,
01046 0x3663, 0x3664, 0x3665, 0x3666, 0x3667, 0x3668, 0x3669, 0x366a,
01047 0x366b, 0x366c, 0x366d, 0x3670, 0x3671, 0x366e, 0x366f, 0x3672,
01048 0x3673, 0x3674, 0x3675, 0x3676, 0x3677, 0x3678, 0x3679, 0x367a,
01049 0x367b, 0x367d, 0x367e, 0x367f, 0x367c, 0x3721, 0x3722, 0x3723, 0x3724,
01050 0x3725, 0x3726, 0x3727, 0x3728, 0x3729, 0x372a, 0x372b, 0x372c,
01051 0x372d, 0x372e, 0x372f, 0x3730, 0x3731, 0x3732, 0x3733, 0x3734,
01052 0x3735, 0x3736, 0x3737, 0x3738, 0x3739, 0x373a, 0x373b, 0x373c,
01053 0x373d, 0x373e, 0x373f, 0x3740, 0x3741, 0x3742, 0x3743, 0x3744,
01054 0x3745, 0x3746, 0x3747, 0x3748, 0x3749, 0x374a, 0x374b, 0x374c,
01055 0x374d, 0x374e, 0x374f, 0x3750, 0x3751, 0x3752, 0x3753, 0x3754,
01056 0x3755, 0x3756, 0x3757, 0x3758, 0x3759, 0x375a, 0x375b,
01057 0x375c, 0x375d, 0x375e, 0x375f, 0x3761, 0x3762, 0x3763, 0x3764,
01058 0x3765, 0x3766, 0x3767, 0x3768, 0x3769, 0x376a, 0x376b, 0x376c,
01059 0x376d, 0x377e, 0x377e, 0x376e, 0x376f, 0x3770, 0x3771, 0x3772, 0x3773,
01060 0x3774, 0x3775, 0x3776, 0x3777, 0x3778, 0x3779, 0x377a, 0x377b,
01061 0x377c, 0x377d, 0x3821, 0x3822, 0x3823, 0x3824, 0x3825, 0x3826,
01062 0x3827, 0x3828, 0x3829, 0x382a, 0x382b, 0x382c, 0x382d, 0x382e,
01063 0x382f, 0x3830, 0x3831, 0x3832, 0x3833, 0x3834, 0x3835, 0x3836,
01064 0x3837, 0x3838, 0x3839, 0x383a, 0x383b, 0x383c, 0x383d, 0x383e,
01065 0x383f, 0x3840, 0x3841, 0x3842, 0x3843, 0x3844, 0x3845, 0x3846,
01066 0x3847, 0x3848, 0x3849, 0x384a, 0x384b, 0x384c, 0x384d, 0x384e,
01067 0x3850, 0x3851, 0x384f, 0x3852, 0x3853, 0x3854, 0x3855, 0x3856,
01068 0x3857, 0x3858, 0x3859, 0x385a, 0x385b, 0x385c, 0x385d, 0x385e,
01069 0x385f, 0x3860, 0x3861, 0x3862, 0x3863, 0x3864, 0x3865, 0x3867,
01070 0x3868, 0x3869, 0x386a, 0x386b, 0x386c, 0x386d, 0x386e, 0x386f,
01071 0x3870, 0x3871, 0x3872, 0x3873, 0x3874, 0x3875, 0x3876, 0x3877,
01072 0x3878, 0x3879, 0x387a, 0x387b, 0x387c, 0x387d, 0x387e, 0x3921,
01073 0x3922, 0x3923, 0x3924, 0x3925, 0x3926, 0x3927, 0x3928, 0x3929,
01074 0x392a, 0x392b, 0x392c, 0x392d, 0x392e, 0x392f, 0x3930, 0x3931,
01075 0x3932, 0x3933, 0x3934, 0x3935, 0x3936, 0x3937, 0x3938, 0x3939,
01076 0x393a, 0x393b, 0x393c, 0x393d, 0x393e, 0x393f, 0x3940, 0x3941,
01077 0x3942, 0x3943, 0x3944, 0x3945, 0x3946, 0x3947, 0x3948, 0x3949,
01078 0x394a, 0x394b, 0x394c, 0x394d, 0x394e, 0x394f, 0x3950, 0x3951,
01079 0x3952, 0x3953, 0x3954, 0x3955, 0x3956, 0x3957, 0x3958, 0x3959,
01080 0x395a, 0x395b, 0x395c, 0x395d, 0x395e, 0x395f, 0x3960, 0x3961,
01081 0x3962, 0x3963, 0x3964, 0x3965, 0x3966, 0x3967, 0x3968, 0x3969,
01082 0x396a, 0x396b, 0x396c, 0x396d, 0x396e, 0x396f, 0x3970, 0x3971,
01083 0x3972, 0x3973, 0x3974, 0x3975, 0x3976, 0x3977, 0x3978, 0x3979,
01084 0x397a, 0x397b, 0x397c, 0x397d, 0x397e, 0x3a21, 0x3a22, 0x3a23,
01085 0x3a24, 0x3a25, 0x3a26, 0x3a27, 0x3a28, 0x3a29, 0x3a2a, 0x3a2b,
01086 0x3a2c, 0x3a2d, 0x3a2e, 0x3a2f, 0x3a30, 0x3a31, 0x3a32, 0x3a33,
01087 0x3a34, 0x3a35, 0x3a36, 0x3a37, 0x3a38, 0x3a39, 0x3a3a, 0x3a3b,
01088 0x3a3c, 0x3a3d, 0x3a3e, 0x3a3f, 0x3a40, 0x3a41, 0x3a42, 0x3a43,
01089 0x3a44, 0x3a45, 0x3a46, 0x3a47, 0x3a48, 0x3a49, 0x3a4a, 0x3a4b,
01090 0x3a4c, 0x3a4d, 0x3a4e, 0x3a4f, 0x3a50, 0x3a51, 0x3a52, 0x3a53,
01091 0x3a54, 0x3a55, 0x3a56, 0x3a57, 0x3a58, 0x3a59, 0x3a5a, 0x3a5b,
01092 0x3a5c, 0x3a5d, 0x3a5e, 0x3a5f, 0x3a60, 0x3a61, 0x3a62, 0x3a63,
01093 0x3a64, 0x3a65, 0x3a66, 0x3a67, 0x3a68, 0x3a69, 0x3a6a, 0x3a6b,
01094 0x3a6c, 0x3a6d, 0x3a6e, 0x3a6f, 0x3a70, 0x3a71, 0x3a72, 0x3a73,
01095 0x3a74, 0x3a75, 0x3a76, 0x3a77, 0x3a78, 0x3a79, 0x3a7a, 0x3a7b,
01096 0x3a7c, 0x3a7d, 0x3a7e, 0x3b21, 0x3b22, 0x3b23, 0x3b24, 0x3b25,
01097 0x3b26, 0x3b27, 0x3b28, 0x3b29, 0x3b2a, 0x3b2b, 0x3b2c, 0x3b2d,
01098 0x3b2e, 0x3b2f, 0x3b30, 0x3b31, 0x3b32, 0x3b33, 0x3b34, 0x3b35,
01099 0x3b36, 0x3b37, 0x3b38, 0x3b39, 0x3b3a, 0x3b3b, 0x3b3c, 0x3b3d,
01100 0x3b3e, 0x3b3f, 0x3b40, 0x3b41, 0x3b42, 0x3b43, 0x3b44, 0x3b45,
01101 0x3b47, 0x3b48, 0x3b49, 0x3b4a, 0x3b4b, 0x3b4c, 0x3b4d, 0x3b4e,
01102 0x3b4f, 0x3b50, 0x3b51, 0x3b52, 0x3b53, 0x3b54, 0x3b55, 0x3b56,
01103 0x3b57, 0x3b58, 0x3b59, 0x3b5a, 0x3b5b, 0x3b5c, 0x3b5d, 0x3b5e,
01104 0x3b5f, 0x3b60, 0x3b61, 0x3b62, 0x3b63, 0x3b64, 0x3b65, 0x3b66,
01105 0x3b67, 0x3b68, 0x3b69, 0x3b6a, 0x3b6b, 0x3b6c, 0x3b6d, 0x3b6e,
01106 0x3b6f, 0x3b70, 0x3b71, 0x3b72, 0x3b73, 0x3b74, 0x3b75, 0x3b76,
01107 0x3b77, 0x3b78, 0x3b79, 0x3b7a, 0x3b7b, 0x3b7c, 0x3b7d, 0x3b7e,
01108 0x3b7f, 0x3b80, 0x3c21, 0x3c22, 0x3c23, 0x3c24, 0x3c25, 0x3c26,
01109 0x3c27, 0x3c28, 0x3c29, 0x3c2a, 0x3c2b, 0x3c2c, 0x3c2d, 0x3c2e,
01110 0x3c2f, 0x3c30, 0x3c31, 0x3c32, 0x3c33, 0x3c34, 0x3c35, 0x3c36,
01111 0x3c37, 0x3c38, 0x3c39, 0x3c3a, 0x3c3b, 0x3c3c, 0x3c3d, 0x3c3e,
01112 0x3c3f, 0x3c40, 0x3c41, 0x3c42, 0x3c43, 0x3c44, 0x3c45, 0x3c46,
01113 0x3c47, 0x3c48, 0x3c49, 0x3c4a, 0x3c4b, 0x3c4c, 0x3c4d, 0x3c4e,
01114 0x3c4f, 0x3c50, 0x3c51, 0x3c52, 0x3c53, 0x3c54, 0x3c55, 0x3c56,
01115 0x3c57, 0x3c58, 0x3c59, 0x3c5a, 0x3c5b, 0x3c5c, 0x3c5d, 0x3c5e,
01116 0x3c5f, 0x3c60, 0x3c61, 0x3c62, 0x3c63, 0x3c64, 0x3c65, 0x3c66,
01117 0x3c67, 0x3c68, 0x3c69, 0x3c6a, 0x3c6b, 0x3c6c, 0x3c6d, 0x3c6e,
01118 0x3c6f, 0x3c70, 0x3c71, 0x3c72, 0x3c73, 0x3c74, 0x3c75, 0x3c76,
01119 0x3c77, 0x3c78, 0x3c79, 0x3c7a, 0x3c7b, 0x3c7c, 0x3c7d, 0x3c7e,
01120 0x3d21, 0x3d22, 0x3d23, 0x3d24, 0x3d25, 0x3d26, 0x3d27, 0x3d28,
01121 0x3d29, 0x3d2a, 0x3d2b, 0x3d2c, 0x3d2d, 0x3d2e, 0x3d2f, 0x3d30,
01122 0x3d31, 0x3d32, 0x3d33, 0x3d34, 0x3d35, 0x3d36, 0x3d37, 0x3d38,
01123 0x3d39, 0x3d3a, 0x3d3b, 0x3d3c, 0x3d3d, 0x3d3e, 0x3d3f, 0x3d40,
01124 0x3d41, 0x3d42, 0x3d43, 0x3d44, 0x3d45, 0x3d46, 0x3d47, 0x3d48,
01125 0x3d49, 0x3d4a, 0x3d4b, 0x3d4c, 0x3d4d, 0x3d4e, 0x3d4f, 0x3d50,
01126 0x3d51, 0x3d52, 0x3d53, 0x3d54, 0x3d55, 0x3d56, 0x3d57, 0x3d58,
01127 0x3d59, 0x3d5a, 0x3d5b, 0x3d5c, 0x3d5d, 0x3d5e, 0x3d5f, 0x3d60,
01128 0x3d61, 0x3d62, 0x3d63, 0x3d64, 0x3d65, 0x3d66, 0x3d67, 0x3d68,
01129 0x3d69, 0x3d6a, 0x3d6b, 0x3d6c, 0x3d6d, 0x3d6e, 0x3d6f, 0x3d70,
01130 0x3d71, 0x3d72, 0x3d73, 0x3d74, 0x3d75, 0x3d76, 0x3d77, 0x3d78,
01131 0x3d79, 0x3d7a, 0x3d7b, 0x3d7c, 0x3d7d, 0x3d7e, 0x3d7f, 0x3e21,

```
01132 0x3e23, 0x3e24, 0x3e25, 0x3e26, 0x3e27, 0x3e28, 0x3e29, 0x3e2a,
01133 0x3e2b, 0x3e2c, 0x3e2d, 0x3e2e, 0x3e2f, 0x3e30, 0x3e31, 0x3e32,
01134 0x3e33, 0x3e34, 0x3e35, 0x3e36, 0x3e37, 0x3e38, 0x3e39, 0x3e3a,
01135 0x3e3b, 0x3e3c, 0x3e3d, 0x3e3e, 0x3e3f, 0x3e40, 0x3e41, 0x3e42,
01136 0x3e43, 0x3e44, 0x3e45, 0x3e46, 0x3e47, 0x3e48, 0x3e49, 0x3e4a,
01137 0x3e4b, 0x3e4c, 0x3e4d, 0x3e4e, 0x3e4f, 0x3e50, 0x3e51, 0x3e52,
01138 0x3e53, 0x3e54, 0x3e55, 0x3e56, 0x3e57, 0x3e58, 0x3e59, 0x3e5a,
01139 0x3e5b, 0x3e5c, 0x3e5d, 0x3e5e, 0x3e5f, 0x3e60, 0x3e61, 0x3e62,
01140 0x3e63, 0x3e64, 0x3e65, 0x3e66, 0x3e67, 0x3e68, 0x3e69, 0x3e6a,
01141 0x3e6b, 0x3e6c, 0x3e6d, 0x3e6e, 0x3e6f, 0x3e70, 0x3e71, 0x3e72,
01142 0x3e73, 0x3e74, 0x3e75, 0x3e76, 0x3e77, 0x3e78, 0x3e79, 0x3e7a,
01143 0x3e7b, 0x3e7c, 0x3e7d, 0x3e7e, 0x3e7f, 0x3e80, 0x3e81, 0x3e82,
01144 0x3f25, 0x3f26, 0x3f27, 0x3f28, 0x3f29, 0x3f2a, 0x3f2b, 0x3f2c,
01145 0x3f2d, 0x3f2e, 0x3f2f, 0x3f30, 0x3f31, 0x3f32, 0x3f33, 0x3f34,
01146 0x3f35, 0x3f36, 0x3f37, 0x3f38, 0x3f39, 0x3f3a, 0x3f3b, 0x3f3c,
01147 0x3f3d, 0x3f3e, 0x3f3f, 0x3f40, 0x3f41, 0x3f42, 0x3f43, 0x3f44,
01148 0x3f45, 0x3f46, 0x3f47, 0x3f48, 0x3f49, 0x3f4a, 0x3f4b, 0x3f4c,
01149 0x3f4d, 0x3f4e, 0x3f4f, 0x3f50, 0x3f51, 0x3f52, 0x3f53, 0x3f54,
01150 0x3f55, 0x3f56, 0x3f57, 0x3f58, 0x3f59, 0x3f5a, 0x3f5b, 0x3f5c,
01151 0x3f5d, 0x3f5e, 0x3f5f, 0x3f60, 0x3f61, 0x3f62, 0x3f63, 0x3f64,
01152 0x3f65, 0x3f66, 0x3f67, 0x3f68, 0x3f69, 0x3f6a, 0x3f6b, 0x3f6c,
01153 0x3f6d, 0x3f6e, 0x3f6f, 0x3f70, 0x3f71, 0x3f72, 0x3f73, 0x3f74,
01154 0x3f75, 0x3f76, 0x3f77, 0x3f78, 0x3f79, 0x3f7a, 0x3f7b, 0x3f7c,
01155 0x3f7d, 0x3f7e, 0x4021, 0x4022, 0x4023, 0x4024, 0x4025, 0x4026,
01156 0x4027, 0x4028, 0x4029, 0x402a, 0x402b, 0x402c, 0x402d, 0x402e,
01157 0x402f, 0x4030, 0x4031, 0x4032, 0x4033, 0x4034, 0x4035, 0x4036,
01158 0x4037, 0x4038, 0x4039, 0x403a, 0x403b, 0x403c, 0x403d, 0x403e,
01159 0x403f, 0x4040, 0x4041, 0x4042, 0x4043, 0x4044, 0x4045, 0x4046,
01160 0x4047, 0x4048, 0x4049, 0x404a, 0x404b, 0x404c, 0x404d, 0x404e,
01161 0x404f, 0x4050, 0x4051, 0x4052, 0x4053, 0x4054, 0x4055, 0x4056,
01162 0x4057, 0x4058, 0x4059, 0x405a, 0x405b, 0x405c, 0x405d, 0x405e,
01163 0x405f, 0x4060, 0x4061, 0x4062, 0x4063, 0x4064, 0x4065, 0x4066,
01164 0x4067, 0x4068, 0x4069, 0x406a, 0x406b, 0x406c, 0x406d, 0x406e,
01165 0x406f, 0x4070, 0x4071, 0x4072, 0x4073, 0x4074, 0x4075, 0x4076,
01166 0x4077, 0x4078, 0x4079, 0x407a, 0x407b, 0x407c, 0x407d, 0x407e,
01167 0x4121, 0x4122, 0x4123, 0x4124, 0x4125, 0x4126, 0x4127, 0x4128,
01168 0x4129, 0x412a, 0x412b, 0x412c, 0x412d, 0x412e, 0x412f, 0x4130,
01169 0x4131, 0x4132, 0x4133, 0x4134, 0x4135, 0x4136, 0x4137, 0x4138,
01170 0x4139, 0x413a, 0x413b, 0x413c, 0x413d, 0x413e, 0x413f, 0x4140,
01171 0x4141, 0x4142, 0x4143, 0x4144, 0x4145, 0x4146, 0x4147, 0x4148,
01172 0x4149, 0x414a, 0x414b, 0x414c, 0x414d, 0x414e, 0x414f, 0x4150,
01173 0x4151, 0x4152, 0x4153, 0x4154, 0x4155, 0x4156, 0x4157, 0x4158,
01174 0x4159, 0x415a, 0x415b, 0x415c, 0x415d, 0x415e, 0x415f, 0x4160,
01175 0x4161, 0x4162, 0x4163, 0x4164, 0x4165, 0x4166, 0x4167, 0x4168,
01176 0x4169, 0x416a, 0x416b, 0x416c, 0x416d, 0x416e, 0x416f, 0x4170,
01177 0x4171, 0x4172, 0x4173, 0x4174, 0x4175, 0x4176, 0x4177, 0x4178,
01178 0x4179, 0x417a, 0x417b, 0x417c, 0x417d, 0x417e, 0x4221, 0x4222,
01179 0x4223, 0x4224, 0x4225, 0x4226, 0x4227, 0x4228, 0x4229, 0x422a,
01180 0x422b, 0x422c, 0x422d, 0x422e, 0x422f, 0x4230, 0x4231, 0x4232,
01181 0x4233, 0x4234, 0x4235, 0x4236, 0x4237, 0x4238, 0x4239, 0x423a,
01182 0x423b, 0x423c, 0x423d, 0x423e, 0x423f, 0x4240, 0x4241, 0x4242,
01183 0x4243, 0x4244, 0x4245, 0x4246, 0x4247, 0x4248, 0x4249, 0x424a,
01184 0x424b, 0x424c, 0x424d, 0x424e, 0x424f, 0x4250, 0x4251, 0x4252,
01185 0x4253, 0x4254, 0x4255, 0x4256, 0x4257, 0x4258, 0x4259, 0x425a,
01186 0x425b, 0x425c, 0x425d, 0x425e, 0x425f, 0x4260, 0x4261, 0x4262,
01187 0x4263, 0x4264, 0x4265, 0x4266, 0x4267, 0x4268, 0x4269, 0x426a,
01188 0x426b, 0x426c, 0x426d, 0x426e, 0x426f, 0x4270, 0x4271, 0x4272,
01189 0x4273, 0x4274, 0x4275, 0x4276, 0x4277, 0x4278, 0x4279, 0x427a,
01190 0x427b, 0x427c, 0x427d, 0x427e, 0x4321, 0x4322, 0x4323, 0x4324,
01191 0x4325, 0x4326, 0x4327, 0x4328, 0x4329, 0x432a, 0x432b, 0x432c,
01192 0x432d, 0x432e, 0x432f, 0x4330, 0x4331, 0x4332, 0x4333, 0x4334,
01193 0x4335, 0x4336, 0x4337, 0x4338, 0x4339, 0x433a, 0x433b, 0x433c, 0x433d,
01194 0x433e, 0x433f, 0x4340, 0x4341, 0x4342, 0x4343, 0x4344, 0x4345,
01195 0x4346, 0x4347, 0x4348, 0x4349, 0x434a, 0x434b, 0x434c, 0x434d,
01196 0x434e, 0x434f, 0x4350, 0x4351, 0x4352, 0x4353, 0x4354, 0x4355,
01197 0x4356, 0x4357, 0x4358, 0x4359, 0x435a, 0x435b, 0x435c, 0x435d,
01198 0x435e, 0x435f, 0x4360, 0x4361, 0x4362, 0x4363, 0x4364, 0x4365,
01199 0x4366, 0x4367, 0x4368, 0x4369, 0x436a, 0x436b, 0x436c, 0x436d,
01200 0x436e, 0x436f, 0x4370, 0x4371, 0x4372, 0x4373, 0x4374, 0x4375,
01201 0x4376, 0x4377, 0x4378, 0x4379, 0x437a, 0x437b, 0x437c, 0x437d,
01202 0x437e, 0x437f, 0x4421, 0x4422, 0x4423, 0x4424, 0x4425, 0x4426,
01203 0x4427, 0x4428, 0x4429, 0x442a, 0x442b, 0x442c, 0x442d, 0x442e,
01204 0x442f, 0x4430, 0x4431, 0x4432, 0x4433, 0x4434, 0x4435, 0x4436,
01205 0x4437, 0x4438, 0x4439, 0x443a, 0x443b, 0x443c, 0x443d, 0x443e,
01206 0x443f, 0x4440, 0x4441, 0x4442, 0x4443, 0x4444, 0x4445, 0x4446,
01207 0x4447, 0x4448, 0x4449, 0x444a, 0x444b, 0x444c, 0x444d, 0x444e,
01208 0x444f, 0x4450, 0x4451, 0x4452, 0x4453, 0x4454, 0x4455, 0x4456,
01209 0x4457, 0x4458, 0x4459, 0x445a, 0x445b, 0x445c, 0x445d, 0x445e,
01210 0x445f, 0x4460, 0x4461, 0x4462, 0x4463, 0x4464, 0x4465, 0x4466,
01211 0x4467, 0x4468, 0x4469, 0x446a, 0x446b, 0x446c, 0x446d, 0x446e,
01212 0x446f, 0x4470, 0x4471, 0x4472, 0x4473, 0x4474, 0x4475, 0x4476,
01213 0x4477, 0x4478, 0x4479, 0x447a, 0x447b, 0x447c, 0x447d, 0x447e,
01214 0x447f, 0x4480, 0x4481, 0x4482, 0x4483, 0x4484, 0x4485, 0x4486,
01215 0x4487, 0x4488, 0x4489, 0x448a, 0x448b, 0x448c, 0x448d, 0x448e,
01216 0x448f, 0x4490, 0x4491, 0x4492, 0x4493, 0x4494, 0x4495, 0x4496,
01217 0x4497, 0x4498, 0x4499, 0x449a, 0x449b, 0x449c, 0x449d, 0x449e,
01218 0x449f, 0x4500, 0x4501, 0x4502, 0x4503, 0x4504, 0x4505, 0x4506,
```

```
01219 0x4549, 0x454a, 0x454b, 0x454d, 0x454c, 0x454e, 0x454f, 0x4550,
01220 0x4551, 0x4552, 0x4553, 0x4554, 0x4555, 0x4556, 0x4557, 0x4558,
01221 0x4559, 0x455a, 0x455b, 0x455c, 0x455d, 0x455e, 0x455f, 0x4560,
01222 0x4561, 0x4562, 0x4563, 0x4564, 0x4565, 0x4566, 0x4567, 0x4568,
01223 0x4569, 0x456a, 0x456b, 0x456c, 0x456d, 0x456e, 0x456f, 0x4570,
01224 0x4571, 0x4572, 0x4573, 0x4574, 0x4575, 0x4576, 0x4577, 0x4578,
01225 0x4579, 0x457a, 0x457b, 0x457c, 0x457d, 0x457e, 0x4621, 0x4622,
01226 0x4623, 0x4624, 0x4625, 0x4626, 0x4627, 0x4628, 0x4629, 0x462a,
01227 0x462b, 0x462c, 0x462d, 0x462e, 0x462f, 0x4630, 0x4631, 0x4632,
01228 0x4633, 0x4634, 0x4635, 0x4636, 0x4637, 0x4638, 0x4639, 0x463a,
01229 0x463b, 0x463c, 0x463d, 0x463e, 0x463f, 0x4640, 0x4641, 0x4642,
01230 0x4643, 0x4644, 0x4645, 0x4646, 0x4647, 0x4648, 0x4649, 0x464a,
01231 0x464b, 0x464c, 0x464d, 0x464e, 0x464f, 0x4650, 0x4651, 0x4652,
01232 0x4653, 0x4654, 0x4655, 0x4656, 0x4657, 0x4658, 0x4659, 0x465a,
01233 0x465b, 0x465c, 0x465d, 0x465e, 0x465f, 0x4660, 0x4736, 0x4661,
01234 0x4662, 0x4663, 0x4664, 0x4665, 0x4666, 0x4667, 0x4668, 0x4669,
01235 0x466a, 0x466b, 0x466c, 0x466d, 0x466e, 0x466f, 0x4670, 0x4671,
01236 0x4672, 0x4673, 0x4674, 0x4675, 0x4676, 0x4677, 0x4678, 0x4679,
01237 0x467a, 0x467b, 0x467c, 0x467d, 0x467e, 0x4721, 0x4722, 0x4723,
01238 0x4724, 0x4725, 0x4726, 0x4727, 0x4728, 0x4729, 0x472a, 0x472b,
01239 0x472c, 0x472d, 0x472e, 0x472f, 0x4730, 0x4731, 0x4732, 0x4733,
01240 0x4734, 0x4735, 0x4737, 0x4738, 0x4739, 0x473a, 0x473b, 0x473c,
01241 0x473d, 0x473e, 0x473f, 0x4740, 0x4741, 0x4742, 0x4743, 0x4744,
01242 0x4745, 0x4746, 0x4747, 0x4748, 0x4749, 0x474a, 0x474b, 0x474c,
01243 0x474d, 0x474e, 0x474f, 0x4750, 0x4751, 0x4752, 0x4753, 0x4754,
01244 0x4755, 0x4756, 0x4757, 0x4758, 0x4759, 0x475a, 0x475b, 0x475c,
01245 0x475d, 0x475e, 0x475f, 0x4760, 0x4761, 0x4762, 0x4763, 0x4764,
01246 0x4765, 0x4766, 0x4767, 0x4768, 0x4769, 0x476a, 0x476b, 0x476c,
01247 0x476d, 0x476e, 0x476f, 0x4770, 0x4771, 0x4772, 0x4773, 0x4774,
01248 0x4775, 0x4776, 0x4777, 0x4778, 0x4779, 0x477a, 0x477b, 0x477c,
01249 0x477d, 0x477e, 0x4821, 0x4822, 0x4823, 0x4824, 0x4825, 0x4826,
01250 0x4827, 0x4828, 0x4829, 0x482a, 0x482b, 0x482c, 0x482d, 0x482e,
01251 0x482f, 0x4830, 0x4831, 0x4832, 0x4833, 0x4834, 0x4835, 0x4836,
01252 0x4837, 0x4838, 0x4839, 0x483a, 0x483b, 0x483c, 0x483d, 0x483e,
01253 0x483f, 0x4840, 0x4841, 0x4842, 0x4843, 0x4844, 0x4845, 0x4846,
01254 0x4847, 0x4848, 0x4849, 0x484a, 0x484b, 0x484c, 0x4853, 0x484d,
01255 0x484e, 0x484f, 0x4850, 0x4851, 0x4852, 0x4854, 0x4855, 0x4856,
01256 0x4857, 0x4858, 0x4859, 0x485a, 0x485b, 0x485c, 0x485d, 0x485e,
01257 0x485f, 0x4860, 0x4861, 0x4862, 0x4863, 0x4864, 0x4865, 0x4866,
01258 0x4867, 0x4868, 0x4869, 0x486a, 0x486b, 0x486c, 0x486d, 0x486e,
01259 0x486f, 0x4870, 0x4871, 0x4872, 0x4873, 0x4874, 0x4875, 0x4876,
01260 0x4877, 0x4878, 0x4879, 0x487a, 0x487b, 0x487c, 0x487d, 0x487e,
01261 0x4921, 0x4922, 0x4923, 0x4924, 0x4925, 0x4926, 0x4927, 0x4928,
01262 0x4929, 0x492a, 0x492b, 0x492c, 0x492d, 0x492e, 0x492f, 0x4930,
01263 0x4931, 0x4932, 0x4933, 0x4934, 0x4935, 0x4936, 0x4937, 0x4938,
01264 0x4939, 0x493a, 0x493b, 0x493c, 0x4941, 0x493d, 0x493e, 0x493f,
01265 0x4940, 0x4942, 0x4943, 0x4944, 0x4945, 0x4946, 0x4947, 0x4948,
01266 0x4949, 0x494a, 0x494b, 0x494c, 0x494d, 0x494e, 0x494f, 0x4950,
01267 0x4951, 0x4952, 0x4953, 0x4954, 0x4955, 0x4956, 0x4957, 0x4958,
01268 0x4959, 0x495a, 0x495b, 0x495c, 0x495d, 0x495e, 0x495f, 0x4960,
01269 0x4961, 0x4962, 0x4963, 0x4964, 0x4965, 0x4966, 0x4967, 0x4968,
01270 0x4969, 0x496a, 0x496b, 0x496c, 0x496d, 0x496e, 0x496f, 0x4970,
01271 0x4971, 0x4972, 0x4973, 0x4974, 0x4975, 0x4976, 0x4977, 0x4978,
01272 0x4979, 0x497a, 0x497b, 0x497c, 0x497d, 0x497e, 0x4a21, 0x4a22,
01273 0x4a23, 0x4a24, 0x4a25, 0x4a26, 0x4a27, 0x4a28, 0x4a29, 0x4a2a,
01274 0x4a2b, 0x4a2c, 0x4a2d, 0x4a2e, 0x4a2f, 0x4a30, 0x4a31, 0x4a32,
01275 0x4a33, 0x4a34, 0x4a35, 0x4a36, 0x4a37, 0x4a38, 0x4a39, 0x4a3a,
01276 0x4a3b, 0x4a3c, 0x4a3d, 0x4a3e, 0x4a3f, 0x4a40, 0x4a41, 0x4a42,
01277 0x4a43, 0x4a44, 0x4a45, 0x4a46, 0x4a47, 0x4a48, 0x4a49, 0x4a4a,
01278 0x4a4b, 0x4a4c, 0x4a4d, 0x4a4e, 0x4a4f, 0x4a50, 0x4a51, 0x4a52,
01279 0x4a53, 0x4a54, 0x4a55, 0x4a56, 0x4a57, 0x4a58, 0x4a59, 0x4a5a,
01280 0x4a5b, 0x4a5c, 0x4a5d, 0x4a5e, 0x4a5f, 0x4a60, 0x4a61, 0x4a62,
01281 0x4a63, 0x4a64, 0x4a65, 0x4a66, 0x4a67, 0x4a68, 0x4a69, 0x4a6a,
01282 0x4a6b, 0x4a6c, 0x4a6d, 0x4a6e, 0x4a6f, 0x4a70, 0x4a71, 0x4a72,
01283 0x4a73, 0x4a74, 0x4a75, 0x4a76, 0x4a77, 0x4a78, 0x4a79, 0x4a7a,
01284 0x4a7b, 0x4a7c, 0x4a7d, 0x4a7e, 0x4b21, 0x4b22, 0x4b23, 0x4b24,
01285 0x4b25, 0x4b26, 0x4b27, 0x4b28, 0x4b29, 0x4b2a, 0x4b2b, 0x4b2c,
01286 0x4b2d, 0x4b2e, 0x4b2f, 0x4b30, 0x4b31, 0x4b32, 0x4b33, 0x4b34,
01287 0x4b35, 0x4b36, 0x4b37, 0x4b38, 0x4b39, 0x4b3a, 0x4b3b, 0x4b3c,
01288 0x4b3d, 0x4b3e, 0x4b3f, 0x4b40, 0x4b41, 0x4b42, 0x4b43, 0x4b44,
01289 0x4b45, 0x4b46, 0x4b47, 0x4b48, 0x4b49, 0x4b4a, 0x4b4b, 0x4b4c,
01290 0x4b4d, 0x4b4e, 0x4b4f, 0x4b50, 0x4b51, 0x4b52, 0x4b53, 0x4b54,
01291 0x4b55, 0x4b56, 0x4b57, 0x4b58, 0x4b59, 0x4b5a, 0x4b5b, 0x4b5c,
01292 0x4b5d, 0x4b5e, 0x4b5f, 0x4b60, 0x4b61, 0x4b62, 0x4b63, 0x4b64,
01293 0x4b65, 0x4b66, 0x4b67, 0x4b68, 0x4b69, 0x4b6a, 0x4b6b, 0x4b6c,
01294 0x4b6d, 0x4b6e, 0x4b6f, 0x4b70, 0x4b71, 0x4b72, 0x4b73, 0x4b74,
01295 0x4b75, 0x4b76, 0x4b77, 0x4b78, 0x4b79, 0x4b7a, 0x4b7b, 0x4b7c,
01296 0x4b7d, 0x4b7e, 0x4c21, 0x4c22, 0x4c23, 0x4c24, 0x4c25, 0x4c26,
01297 0x4c27, 0x4c28, 0x4c29, 0x4c2a, 0x4c2b, 0x4c2c, 0x4c2d, 0x4c2e,
01298 0x4c2f, 0x4c30, 0x4c31, 0x4c32, 0x4c33, 0x4c34, 0x4c35, 0x4c36,
01299 0x4c37, 0x4c38, 0x4c39, 0x4c3a, 0x4c3b, 0x4c3c, 0x4c3d, 0x4c3e,
01300 0x4c3f, 0x4c40, 0x4c41, 0x4c42, 0x4c43, 0x4c44, 0x4c45, 0x4c46,
01301 0x4c47, 0x4c48, 0x4c49, 0x4c4a, 0x4c4b, 0x4c4c, 0x4c4d, 0x4c4e,
01302 0x4c4f, 0x4c50, 0x4c51, 0x4c52, 0x4c53, 0x4c54, 0x4c55, 0x4c56,
01303 0x4c57, 0x4c58, 0x4c59, 0x4c5a, 0x4c5b, 0x4c5c, 0x4c5d, 0x4c5e,
01304 0x4c5f, 0x4c60, 0x4c61, 0x4c62, 0x4c63, 0x4c64, 0x4c65, 0x4c66,
01305 0x4c67, 0x4c68, 0x4c69, 0x4c6a, 0x4c6b, 0x4c6c, 0x4c6d, 0x4c6e,
```

```
01306 0x4c6f, 0x4c70, 0x4c71, 0x4c72, 0x4c73, 0x4c74, 0x4c75, 0x4c76,
01307 0x4c77, 0x4c78, 0x4c79, 0x4c7a, 0x4c7b, 0x4c7c, 0x4c7d, 0x4c7e,
01308 0x4d21, 0x4d22, 0x4d23, 0x4d24, 0x4d25, 0x4d26, 0x4d27, 0x4d28,
01309 0x4d29, 0x4d2a, 0x4d2b, 0x4d2c, 0x4d2d, 0x4d2e, 0x4d2f, 0x4d30,
01310 0x4d31, 0x4d32, 0x4d33, 0x4d34, 0x4d35, 0x4d36, 0x4d37, 0x4d38,
01311 0x4d39, 0x4d3a, 0x4d3b, 0x4d3c, 0x4d3d, 0x4d3e, 0x4d3f, 0x4d40,
01312 0x4d41, 0x4d42, 0x4d43, 0x4d44, 0x4d45, 0x4d46, 0x4d47, 0x4d48,
01313 0x4d49, 0x4d4a, 0x4d4b, 0x4d4c, 0x4d4d, 0x4d4e, 0x4d4f, 0x4d50,
01314 0x4d51, 0x4d52, 0x4d53, 0x4d54, 0x4d55, 0x4d56, 0x4d57, 0x4d58,
01315 0x4d59, 0x4d5a, 0x4d5b, 0x4d5c, 0x4d5d, 0x4d5e, 0x4d5f, 0x4d60,
01316 0x4d61, 0x4d62, 0x4d63, 0x4d64, 0x4d65, 0x4d66, 0x4d67, 0x4d68,
01317 0x4d69, 0x4d6a, 0x4d6b, 0x4d6c, 0x4d6d, 0x4d6e, 0x4d6f, 0x4d70,
01318 0x4d71, 0x4d72, 0x4d73, 0x4d74, 0x4d75, 0x4d76, 0x4d77, 0x4d78,
01319 0x4d79, 0x4d7a, 0x4d7b, 0x4d7c, 0x4d7d, 0x4d7e, 0x4e21, 0x4e22,
01320 0x4e24, 0x4e25, 0x4e26, 0x4e27, 0x4e28, 0x4e29, 0x4e23, 0x4e2a,
01321 0x4e2b, 0x4e2c, 0x4e2d, 0x4e2e, 0x4e2f, 0x4e30, 0x4e31, 0x4e32,
01322 0x4e33, 0x4e34, 0x4e35, 0x4e36, 0x4e37, 0x4e38, 0x4e39, 0x4e3a,
01323 0x4e3b, 0x4e3c, 0x4e3d, 0x4e3e, 0x4e3f, 0x4e40, 0x4e41, 0x4e42,
01324 0x4e43, 0x4e44, 0x4e45, 0x4e46, 0x4e47, 0x4e48, 0x4e49, 0x4e4a,
01325 0x4e4b, 0x4e4c, 0x4e4d, 0x4e4e, 0x4e4f, 0x4e50, 0x4e51, 0x4e52,
01326 0x4e53, 0x4e54, 0x4e55, 0x4e56, 0x4e57, 0x4e58, 0x4e59, 0x4e5a,
01327 0x4e5b, 0x4e5c, 0x4e5d, 0x4e5e, 0x4e5f, 0x4e60, 0x4e61, 0x4e62,
01328 0x4e63, 0x4e64, 0x4e65, 0x4e66, 0x4e67, 0x4e68, 0x4e69, 0x4e6a,
01329 0x4e6b, 0x4e6c, 0x4e6d, 0x4e6e, 0x4e6f, 0x4e70, 0x4e71, 0x4e72,
01330 0x4e73, 0x4e74, 0x4e75, 0x4e76, 0x4e77, 0x4e78, 0x4e79, 0x4e7a,
01331 0x4e7b, 0x4e7c, 0x4e7d, 0x4e7e, 0x4f21, 0x4f22, 0x4f23, 0x4f24,
01332 0x4f25, 0x4f26, 0x4f27, 0x4f28, 0x4f29, 0x4f2a, 0x4f2b, 0x4f2c,
01333 0x4f2d, 0x4f2e, 0x4f2f, 0x4f30, 0x4f31, 0x4f32, 0x4f33, 0x4f34,
01334 0x4f35, 0x4f36, 0x4f37, 0x4f38, 0x4f39, 0x4f3a, 0x4f3b, 0x4f3c,
01335 0x4f3d, 0x4f3e, 0x4f3f, 0x4f40, 0x4f41, 0x4f42, 0x4f43, 0x4f44,
01336 0x4f45, 0x4f46, 0x4f47, 0x4f48, 0x4f49, 0x4f4a, 0x4f4b, 0x4f4c,
01337 0x4f4d, 0x4f4e, 0x4f4f, 0x4f50, 0x4f51, 0x4f52, 0x4f53, 0x4f54,
01338 0x4f55, 0x4f56, 0x4f57, 0x4f58, 0x4f59, 0x4f5a, 0x4f5b, 0x4f5c,
01339 0x4f5d, 0x4f5e, 0x4f5f, 0x4f60, 0x4f61, 0x4f62, 0x4f63, 0x4f64,
01340 0x4f65, 0x4f66, 0x4f67, 0x4f68, 0x4f69, 0x4f6a, 0x4f6b, 0x4f6c,
01341 0x4f6d, 0x4f6e, 0x4f6f, 0x4f70, 0x4f71, 0x4f72, 0x4f73, 0x4f74,
01342 0x4f75, 0x4f76, 0x4f77, 0x4f78, 0x4f79, 0x4f7a, 0x4f7b, 0x4f7c,
01343 0x4f7d, 0x4f7e, 0x5021, 0x5022, 0x5023, 0x5024, 0x5025, 0x5026,
01344 0x5027, 0x5028, 0x5029, 0x502a, 0x502b, 0x502c, 0x502d, 0x502e,
01345 0x5030, 0x5031, 0x5032, 0x5033, 0x5034, 0x5035, 0x5037,
01346 0x5038, 0x5039, 0x503a, 0x503b, 0x503c, 0x503d, 0x503e,
01347 0x503f, 0x5040, 0x5041, 0x5042, 0x5043, 0x5044, 0x5045, 0x5046,
01348 0x5047, 0x5048, 0x5049, 0x504a, 0x504b, 0x504c, 0x504d, 0x504e,
01349 0x504f, 0x5050, 0x5051, 0x5052, 0x5053, 0x5054, 0x5055, 0x5056,
01350 0x5057, 0x5058, 0x5059, 0x505a, 0x505b, 0x505c, 0x505d, 0x505e,
01351 0x505f, 0x5060, 0x5061, 0x5062, 0x5063, 0x5064, 0x5065, 0x5066,
01352 0x5067, 0x5068, 0x5069, 0x506a, 0x506b, 0x506c, 0x506d, 0x506e,
01353 0x506f, 0x5070, 0x5071, 0x5072, 0x5073, 0x5074, 0x5075, 0x5076,
01354 0x5077, 0x5078, 0x5079, 0x507a, 0x507b, 0x507c, 0x507d, 0x507e,
01355 0x5121, 0x5122, 0x5123, 0x5124, 0x5125, 0x5126, 0x5127, 0x5128,
01356 0x5129, 0x512a, 0x512b, 0x512c, 0x512d, 0x512e, 0x512f, 0x5130,
01357 0x5131, 0x5132, 0x5133, 0x5134, 0x5135, 0x5136, 0x5137, 0x5138,
01358 0x5139, 0x513a, 0x513b, 0x513c, 0x513d, 0x513e, 0x513f, 0x5140,
01359 0x5141, 0x5142, 0x5143, 0x5144, 0x5145, 0x5146, 0x5147, 0x5148,
01360 0x5149, 0x514a, 0x514b, 0x514c, 0x514d, 0x514e, 0x514f, 0x5150,
01361 0x5151, 0x5152, 0x5153, 0x5154, 0x5155, 0x5156, 0x5157, 0x5158,
01362 0x5159, 0x515a, 0x515b, 0x515c, 0x515d, 0x515e, 0x515f, 0x5160,
01363 0x5161, 0x5162, 0x5163, 0x5164, 0x5165, 0x5166, 0x5167, 0x5168,
01364 0x5169, 0x516a, 0x516b, 0x516c, 0x516d, 0x516e, 0x516f, 0x5170,
01365 0x5171, 0x5172, 0x5173, 0x5174, 0x5175, 0x5176, 0x5177, 0x5178,
01366 0x5179, 0x517a, 0x517b, 0x517c, 0x517d, 0x517e, 0x5221, 0x5222,
01367 0x5223, 0x5224, 0x5225, 0x5226, 0x5227, 0x5228, 0x5229, 0x522a,
01368 0x522b, 0x522c, 0x522d, 0x522e, 0x522f, 0x5230, 0x5231, 0x5232,
01369 0x5233, 0x5234, 0x5235, 0x5236, 0x5237, 0x5238, 0x5239, 0x523a,
01370 0x523b, 0x523c, 0x523d, 0x523e, 0x523f, 0x5240, 0x5241, 0x5242,
01371 0x5243, 0x5244, 0x5245, 0x5246, 0x5247, 0x5248, 0x5249, 0x524a,
01372 0x524b, 0x524c, 0x524d, 0x524e, 0x524f, 0x5250, 0x5251, 0x5252,
01373 0x5253, 0x5254, 0x5255, 0x5256, 0x5257, 0x5258, 0x5259, 0x525a,
01374 0x525b, 0x525c, 0x525d, 0x525e, 0x525f, 0x5260, 0x5261, 0x5262,
01375 0x5263, 0x5264, 0x5265, 0x5266, 0x5267, 0x5268, 0x5269, 0x526a,
01376 0x526b, 0x526c, 0x526d, 0x526e, 0x526f, 0x5270, 0x5271, 0x5272,
01377 0x5273, 0x5274, 0x5275, 0x5276, 0x5277, 0x5278, 0x5279, 0x527a,
01378 0x527b, 0x527c, 0x527d, 0x527e, 0x5321, 0x5322, 0x5323, 0x5324,
01379 0x5325, 0x5326, 0x5327, 0x5328, 0x5329, 0x532a, 0x532b, 0x532c,
01380 0x532d, 0x532e, 0x532f, 0x5330, 0x5331, 0x5332, 0x5333, 0x5334,
01381 0x5335, 0x5336, 0x5337, 0x5338, 0x5339, 0x533a, 0x533b, 0x533c,
01382 0x533d, 0x533e, 0x533f, 0x5340, 0x5341, 0x5342, 0x5343, 0x5344,
01383 0x5345, 0x5346, 0x5347, 0x5348, 0x5349, 0x534a, 0x534b, 0x534c,
01384 0x534d, 0x534e, 0x534f, 0x5350, 0x5351, 0x5352, 0x5353, 0x5354,
01385 0x5355, 0x5356, 0x5357, 0x5358, 0x5359, 0x535a, 0x535b, 0x535c,
01386 0x535d, 0x535e, 0x535f, 0x5360, 0x5361, 0x5362, 0x5363, 0x5364,
01387 0x5365, 0x5366, 0x5367, 0x5368, 0x5369, 0x536a, 0x536b, 0x536c,
01388 0x536d, 0x536e, 0x536f, 0x5370, 0x5371, 0x5372, 0x5373, 0x5374,
01389 0x5375, 0x5376, 0x5377, 0x5378, 0x5379, 0x537a, 0x537b, 0x537c,
01390 0x537d, 0x537e, 0x5421, 0x5422, 0x5423, 0x5424, 0x5425, 0x5426,
01391 0x5427, 0x5428, 0x5429, 0x542a, 0x542b, 0x542c, 0x542d, 0x542e,
01392 0x542f, 0x5430, 0x5431, 0x5432, 0x5433, 0x5434, 0x5435, 0x5436, 0x5437,
```

01393 0x5438, 0x5439, 0x543a, 0x543b, 0x543c, 0x543d, 0x543e, 0x5433,
01394 0x543f, 0x5440, 0x5441, 0x5442, 0x5443, 0x5444, 0x5445, 0x5446,
01395 0x5447, 0x5448, 0x5449, 0x544a, 0x544b, 0x544c, 0x544d, 0x544e,
01396 0x544f, 0x5450, 0x5451, 0x5452, 0x5453, 0x5454, 0x5455, 0x5456,
01397 0x5457, 0x5458, 0x5459, 0x545a, 0x545b, 0x545c, 0x545d, 0x545e,
01398 0x545f, 0x5460, 0x5461, 0x5462, 0x5463, 0x5464, 0x5465, 0x5466,
01399 0x5467, 0x5468, 0x5469, 0x546a, 0x546c, 0x546b, 0x546d, 0x546e,
01400 0x546f, 0x5470, 0x5471, 0x5472, 0x5473, 0x5474, 0x5475, 0x5476,
01401 0x5477, 0x5478, 0x5479, 0x547a, 0x547b, 0x547c, 0x547d, 0x547e,
01402 0x5521, 0x5522, 0x5523, 0x5524, 0x5525, 0x5526, 0x5527, 0x5528,
01403 0x5529, 0x552a, 0x552b, 0x552c, 0x552d, 0x552e, 0x552f, 0x5530,
01404 0x5531, 0x5532, 0x5533, 0x5534, 0x5535, 0x5536, 0x5537, 0x5538,
01405 0x5539, 0x553a, 0x553b, 0x553c, 0x553d, 0x553e, 0x553f, 0x5540,
01406 0x5541, 0x5542, 0x5543, 0x5544, 0x5545, 0x5546, 0x5547, 0x5548,
01407 0x5549, 0x554a, 0x554b, 0x554c, 0x554d, 0x554e, 0x554f, 0x5550,
01408 0x5551, 0x5552, 0x5553, 0x5554, 0x5555, 0x5556, 0x5557, 0x5558,
01409 0x5559, 0x555a, 0x555b, 0x555c, 0x555d, 0x555e, 0x555f, 0x5560,
01410 0x5561, 0x5562, 0x5563, 0x5564, 0x5565, 0x5566, 0x5567, 0x5568,
01411 0x5569, 0x556a, 0x556b, 0x556c, 0x556d, 0x556e, 0x556f, 0x5570,
01412 0x5571, 0x5572, 0x5573, 0x5574, 0x5575, 0x5576, 0x5577, 0x5578,
01413 0x5579, 0x557a, 0x557b, 0x557c, 0x557d, 0x557e, 0x5621, 0x5622,
01414 0x5623, 0x5624, 0x5625, 0x5626, 0x5627, 0x5628, 0x5629, 0x562a,
01415 0x562b, 0x562c, 0x562d, 0x562e, 0x562f, 0x5630, 0x5631, 0x5632,
01416 0x5633, 0x5634, 0x5635, 0x5636, 0x5637, 0x5638, 0x5639, 0x563a,
01417 0x563b, 0x563c, 0x563d, 0x563e, 0x563f, 0x5640, 0x5641, 0x5642,
01418 0x5643, 0x5644, 0x5645, 0x5647, 0x5648, 0x5649, 0x564a, 0x564b,
01419 0x5646, 0x564c, 0x564d, 0x564e, 0x564f, 0x5650, 0x5651, 0x5652,
01420 0x5653, 0x5654, 0x5656, 0x5657, 0x5658, 0x5655, 0x5659, 0x565a,
01421 0x565b, 0x565c, 0x565d, 0x565e, 0x565f, 0x5660, 0x5661, 0x5662,
01422 0x5663, 0x5664, 0x5665, 0x5666, 0x5667, 0x5668, 0x5669, 0x566a,
01423 0x566b, 0x566c, 0x566d, 0x566e, 0x566f, 0x5670, 0x5671, 0x5672,
01424 0x5673, 0x5674, 0x5675, 0x5676, 0x5677, 0x5678, 0x5679, 0x567a,
01425 0x567b, 0x567c, 0x567d, 0x567e, 0x5721, 0x5722, 0x5723, 0x5724,
01426 0x5725, 0x5726, 0x5727, 0x5728, 0x5729, 0x572a, 0x572b, 0x572c,
01427 0x572d, 0x572e, 0x572f, 0x5730, 0x5731, 0x5732, 0x5733, 0x5734,
01428 0x5735, 0x5736, 0x5737, 0x5738, 0x5739, 0x573a, 0x573b, 0x573c,
01429 0x573d, 0x573e, 0x573f, 0x5740, 0x5741, 0x5742, 0x5743, 0x5744,
01430 0x5745, 0x5746, 0x5747, 0x5748, 0x5749, 0x574a, 0x574b, 0x574c,
01431 0x574d, 0x574e, 0x574f, 0x5750, 0x5751, 0x5752, 0x5753, 0x5754,
01432 0x5755, 0x5756, 0x5757, 0x5758, 0x5759, 0x575a, 0x575b, 0x575c,
01433 0x575d, 0x575e, 0x575f, 0x5760, 0x5761, 0x5762, 0x5764, 0x5765,
01434 0x5766, 0x5767, 0x5768, 0x5769, 0x576a, 0x576b, 0x576c, 0x576d,
01435 0x576e, 0x576f, 0x5770, 0x5771, 0x5772, 0x5773, 0x5774, 0x5775,
01436 0x5776, 0x5777, 0x5778, 0x5779, 0x583e, 0x5763, 0x577a, 0x577b,
01437 0x577c, 0x577d, 0x577e, 0x5821, 0x5822, 0x5823, 0x5824, 0x5825,
01438 0x5826, 0x5827, 0x5828, 0x5829, 0x582a, 0x582b, 0x582c, 0x582d,
01439 0x582e, 0x582f, 0x5830, 0x5831, 0x5832, 0x5833, 0x584c, 0x5834,
01440 0x5835, 0x5836, 0x5837, 0x5838, 0x5839, 0x583a, 0x583b, 0x583c,
01441 0x583d, 0x583f, 0x5840, 0x5841, 0x5842, 0x5843, 0x5844, 0x5845,
01442 0x5846, 0x5847, 0x5848, 0x5849, 0x584a, 0x584b, 0x584d, 0x584e,
01443 0x584f, 0x5850, 0x5851, 0x5852, 0x5853, 0x5854, 0x5855, 0x5856,
01444 0x5857, 0x5858, 0x5859, 0x585a, 0x585b, 0x585c, 0x585d, 0x585e,
01445 0x585f, 0x5860, 0x5861, 0x5862, 0x5863, 0x5864, 0x5865, 0x5866,
01446 0x5867, 0x5868, 0x5869, 0x586a, 0x586b, 0x586c, 0x586d, 0x586e,
01447 0x586f, 0x5870, 0x5871, 0x5872, 0x5873, 0x5874, 0x5875, 0x5876,
01448 0x5877, 0x5878, 0x5879, 0x587a, 0x587b, 0x587c, 0x587d, 0x587e,
01449 0x5921, 0x5922, 0x5923, 0x5924, 0x5925, 0x5926, 0x5927, 0x5928,
01450 0x592a, 0x592b, 0x592c, 0x592d, 0x592e, 0x592f, 0x5930, 0x5931,
01451 0x5932, 0x5933, 0x5934, 0x5935, 0x5936, 0x5937, 0x5938, 0x5939,
01452 0x593a, 0x593b, 0x593c, 0x5929, 0x593d, 0x593e, 0x593f, 0x5940,
01453 0x5941, 0x5942, 0x5943, 0x5944, 0x5945, 0x5946, 0x5947, 0x5948,
01454 0x5949, 0x594a, 0x594b, 0x594c, 0x594d, 0x594e, 0x594f, 0x5950,
01455 0x5951, 0x5952, 0x5953, 0x5954, 0x5955, 0x5956, 0x5957, 0x5958,
01456 0x5959, 0x595a, 0x595b, 0x595c, 0x595d, 0x595e, 0x595f, 0x5960,
01457 0x5961, 0x5962, 0x5963, 0x5964, 0x5965, 0x5966, 0x5974, 0x5967,
01458 0x5968, 0x5969, 0x596a, 0x596b, 0x596c, 0x596d, 0x596e, 0x596f,
01459 0x5970, 0x5971, 0x5972, 0x5973, 0x5975, 0x5976, 0x5977, 0x5978,
01460 0x5979, 0x597a, 0x597b, 0x597c, 0x597d, 0x597e, 0x5a21, 0x5a22,
01461 0x5a23, 0x5a24, 0x5a25, 0x5a26, 0x5a27, 0x5a28, 0x5a29, 0x5a2a,
01462 0x5a2b, 0x5a2c, 0x5a2d, 0x5a2e, 0x5a2f, 0x5a30, 0x5a31, 0x5a32,
01463 0x5a33, 0x5a34, 0x5a35, 0x5a36, 0x3866, 0x5a37, 0x5a38, 0x5a39,
01464 0x5a3a, 0x5a3b, 0x5a3c, 0x5a3d, 0x5a3e, 0x5a3f, 0x5a40, 0x5a41,
01465 0x5a42, 0x5a43, 0x5a44, 0x5a45, 0x5a46, 0x5a47, 0x5a48, 0x5a49,
01466 0x5a4a, 0x5a4b, 0x5a4c, 0x5a4d, 0x5a4e, 0x5a4f, 0x5a50, 0x5a51,
01467 0x5a52, 0x5a53, 0x5a54, 0x5a55, 0x5a56, 0x5a57, 0x5a58,
01468 0x5a59, 0x5a5a, 0x5a5b, 0x5a5c, 0x5a5d, 0x5a5e, 0x5a5f, 0x5a60,
01469 0x5a61, 0x5a62, 0x5a63, 0x5a64, 0x5a65, 0x5a66, 0x5a67, 0x5a68,
01470 0x5a69, 0x5a6a, 0x5a6b, 0x5a6c, 0x5a6e, 0x5a6f, 0x5a70, 0x5a71,
01471 0x5a72, 0x5a73, 0x5a74, 0x5a75, 0x5a76, 0x5a77, 0x5a78, 0x5a79,
01472 0x5a7a, 0x5a7b, 0x5a7c, 0x5a7d, 0x5a7e, 0x5b21, 0x5b22, 0x5b23,
01473 0x5b24, 0x5b25, 0x5b26, 0x5b27, 0x5b28, 0x5b29, 0x5b2a, 0x5b2b,
01474 0x5b2c, 0x5b2d, 0x5b2e, 0x5b2f, 0x5b30, 0x5b31, 0x5b32, 0x5b33,
01475 0x5b34, 0x5b35, 0x5b36, 0x5b37, 0x5b38, 0x5b39, 0x5b3a, 0x5b3b,
01476 0x5b3c, 0x5b3d, 0x5b3e, 0x5b3f, 0x5b40, 0x5b41, 0x5b42, 0x5b43,
01477 0x5b44, 0x5b45, 0x5b46, 0x5b47, 0x5b48, 0x5b49, 0x5b4a, 0x5b4b,
01478 0x5b4c, 0x5b4d, 0x5b4e, 0x5b4f, 0x5b50, 0x5b51, 0x5b52, 0x5b53,
01479 0x5b54, 0x5b55, 0x5b56, 0x5b57, 0x5b58, 0x5b59, 0x5b5a, 0x5b5b,

```
01480 0x5b5c, 0x5b5d, 0x5b5e, 0x5b5f, 0x5b60, 0x5b61, 0x5b62, 0x5b63,
01481 0x5b64, 0x5b65, 0x5b66, 0x5b67, 0x5b68, 0x5b69, 0x5b6a, 0x5b6b,
01482 0x5b6c, 0x5b6d, 0x5b6e, 0x5b6f, 0x5b70, 0x5b71, 0x5b72, 0x5b73, 0x5b74,
01483 0x5b75, 0x5b76, 0x5b77, 0x5b78, 0x5b79, 0x5b7a, 0x5b7b, 0x5b7c,
01484 0x5b7d, 0x5b7e, 0x5c21, 0x5c22, 0x5c23, 0x5c24, 0x5c25, 0x5c26,
01485 0x5c27, 0x5c28, 0x5c29, 0x5c2a, 0x5c2b, 0x5c2c, 0x5c2d, 0x5c2e,
01486 0x5c2f, 0x5c30, 0x5c31, 0x5c32, 0x5c33, 0x5c34, 0x5c35, 0x5c36,
01487 0x5c37, 0x5c38, 0x5c39, 0x5c3a, 0x5c3b, 0x5c3c, 0x5c3d, 0x5c3e,
01488 0x5c3f, 0x5c40, 0x5c41, 0x5c42, 0x5c43, 0x5c44, 0x5c45, 0x5c46,
01489 0x5c47, 0x5c48, 0x5c49, 0x5c4a, 0x5c4b, 0x5c4c, 0x5c4d, 0x5c4e,
01490 0x5c4f, 0x5c50, 0x5c51, 0x5c52, 0x5c53, 0x5c54, 0x5c55, 0x5c56,
01491 0x5c57, 0x5c58, 0x5c59, 0x5c5a, 0x5c5b, 0x5c5c, 0x5c5d, 0x5c5e,
01492 0x5c5f, 0x5c60, 0x5c61, 0x5c62, 0x5c63, 0x5c64, 0x5c65, 0x5c66,
01493 0x5c67, 0x5c68, 0x5c69, 0x5c6a, 0x5c6b, 0x5c6c, 0x5c6d, 0x5c6e,
01494 0x5c6f, 0x5c70, 0x5c71, 0x5c72, 0x5c73, 0x5c74, 0x5c75, 0x5c76,
01495 0x5c77, 0x5c78, 0x5c79, 0x5c7a, 0x5c7b, 0x5c7c, 0x5c7d, 0x5c7e,
01496 0x5c7f, 0x5d21, 0x5d22, 0x5d23, 0x5d24, 0x5d25, 0x5d26, 0x5d27, 0x5d28,
01497 0x5d29, 0x5d2a, 0x5d2b, 0x5d2c, 0x5d2d, 0x5d2e, 0x5d2f, 0x5d30,
01498 0x5d31, 0x5d32, 0x5d33, 0x5d34, 0x5d35, 0x5d36, 0x5d37, 0x5d38,
01499 0x5d39, 0x5d3a, 0x5d3b, 0x5d3c, 0x5d3d, 0x5d3e, 0x5d3f, 0x5d40,
01500 0x5d41, 0x5d42, 0x5d43, 0x5d44, 0x5d45, 0x5d46, 0x5d47, 0x5d48,
01501 0x5d49, 0x5d4a, 0x5d4b, 0x5d4c, 0x5d4d, 0x5d4e, 0x5d4f, 0x5d50,
01502 0x5d51, 0x5d52, 0x5d53, 0x5d54, 0x5d55, 0x5d56, 0x5d57, 0x5d58,
01503 0x5d59, 0x5d5a, 0x5d5b, 0x5d5c, 0x5d5d, 0x5d5e, 0x5d5f, 0x5d60,
01504 0x5d61, 0x5d62, 0x5d63, 0x5d64, 0x5d65, 0x5d66, 0x5d67, 0x5d68,
01505 0x5d69, 0x5d6a, 0x5d6b, 0x5d6c, 0x5d6d, 0x5d6e, 0x5d6f, 0x5d70,
01506 0x5d71, 0x5d72, 0x5d73, 0x5d74, 0x5d75, 0x5d76, 0x5d77, 0x5d78,
01507 0x5d79, 0x5d7a, 0x5d7b, 0x5d7c, 0x5d7d, 0x5d7e, 0x5d7f, 0x5e21,
01508 0x5e22, 0x5e23, 0x5e24, 0x5e25, 0x5e26, 0x5e27, 0x5e28, 0x5e29, 0x5e2a,
01509 0x5e2b, 0x5e2c, 0x5e2d, 0x5e2e, 0x5e2f, 0x5e30, 0x5e31, 0x5e32,
01510 0x5e33, 0x5e34, 0x5e35, 0x5e36, 0x5e37, 0x5e38, 0x5e39, 0x5e3a,
01511 0x5e3b, 0x5e3c, 0x5e3d, 0x5e3e, 0x5e3f, 0x5e40, 0x5e41, 0x5e42,
01512 0x5e43, 0x5e44, 0x5e45, 0x5e46, 0x5e47, 0x5e48, 0x5e49, 0x5e4a,
01513 0x5e4b, 0x5e4c, 0x5e4d, 0x5e4e, 0x5e4f, 0x5e50, 0x5e51, 0x5e52,
01514 0x5e53, 0x5e54, 0x5e55, 0x5e56, 0x5e57, 0x5e58, 0x5e59, 0x5e5a,
01515 0x5e5b, 0x5e5c, 0x5e5d, 0x5e5e, 0x5e5f, 0x5e60, 0x5e61, 0x5e62,
01516 0x5e63, 0x5e64, 0x5e65, 0x5e66, 0x5e67, 0x5e68, 0x5e69, 0x5e6a,
01517 0x5e6b, 0x5e6c, 0x5e6d, 0x5e6e, 0x5e6f, 0x5e70, 0x5e71, 0x5e72,
01518 0x5e73, 0x5e74, 0x5e75, 0x5e76, 0x5e77, 0x5e78, 0x5e79, 0x5e7a,
01519 0x5e7b, 0x5e7c, 0x5e7d, 0x5e7e, 0x5f21, 0x5f22, 0x5f23, 0x5f24,
01520 0x5f25, 0x5f26, 0x5f27, 0x5f28, 0x5f29, 0x5f2a, 0x5f2b, 0x5f2c,
01521 0x5f2d, 0x5f2e, 0x5f2f, 0x5f30, 0x5f31, 0x5f32, 0x5f33, 0x5f34,
01522 0x5f35, 0x5f36, 0x5f37, 0x5f38, 0x5f39, 0x5f3a, 0x5f3b, 0x5f3c,
01523 0x5f3d, 0x5f3e, 0x5f3f, 0x5f40, 0x5f41, 0x5f42, 0x5f43, 0x5f44,
01524 0x5f45, 0x5f46, 0x5f47, 0x5f48, 0x5f49, 0x5f4a, 0x5f4b, 0x5f4c,
01525 0x5f4d, 0x5f4e, 0x5f4f, 0x5f50, 0x5f51, 0x5f52, 0x5f53, 0x5f54,
01526 0x5f55, 0x5f56, 0x5f57, 0x5f58, 0x5f59, 0x5f5a, 0x5f5b, 0x5f5c,
01527 0x5f5d, 0x5f5e, 0x5f5f, 0x5f60, 0x5f61, 0x5f62, 0x5f63, 0x5f64,
01528 0x5f65, 0x5f66, 0x5f67, 0x5f68, 0x5f69, 0x5f6a, 0x5f6b, 0x5f6c,
01529 0x5f6d, 0x5f6e, 0x5f6f, 0x5f70, 0x5f71, 0x5f72, 0x5f73, 0x5f74,
01530 0x5f75, 0x5f76, 0x5f77, 0x5f78, 0x5f79, 0x5f7a, 0x5f7b, 0x5f7c,
01531 0x5f7d, 0x5f7e, 0x6021, 0x6022, 0x6023, 0x6024, 0x6025, 0x6026,
01532 0x6027, 0x6028, 0x6029, 0x602a, 0x602b, 0x602c, 0x602d, 0x602e,
01533 0x602f, 0x6030, 0x6031, 0x6032, 0x6033, 0x6034, 0x6035, 0x6036,
01534 0x6037, 0x6038, 0x6039, 0x603a, 0x603b, 0x603c, 0x603d, 0x603e,
01535 0x603f, 0x6040, 0x6041, 0x6042, 0x6043, 0x6044, 0x6045, 0x6046,
01536 0x6047, 0x6048, 0x6049, 0x604a, 0x604b, 0x604c, 0x604d, 0x604e,
01537 0x604f, 0x6050, 0x6051, 0x6052, 0x6053, 0x6054, 0x6055, 0x6056,
01538 0x6057, 0x6058, 0x6059, 0x605a, 0x605b, 0x605c, 0x605d, 0x605e,
01539 0x605f, 0x6060, 0x6061, 0x6062, 0x6063, 0x6064, 0x6065, 0x6066,
01540 0x6067, 0x6068, 0x6069, 0x606a, 0x606b, 0x606c, 0x606d, 0x606e,
01541 0x606f, 0x6070, 0x6071, 0x6072, 0x6073, 0x6074, 0x6075, 0x6076,
01542 0x6077, 0x6078, 0x6079, 0x607a, 0x607b, 0x607c, 0x607d, 0x607e,
01543 0x607f, 0x6121, 0x6122, 0x6123, 0x6124, 0x6125, 0x6126, 0x6127, 0x6128,
01544 0x6129, 0x612a, 0x612b, 0x612c, 0x612d, 0x612e, 0x612f, 0x6130,
01545 0x6131, 0x6132, 0x6133, 0x6134, 0x6135, 0x6136, 0x6137, 0x6138,
01546 0x6139, 0x613a, 0x613b, 0x613c, 0x613d, 0x613e, 0x613f, 0x6140,
01547 0x6141, 0x6142, 0x6143, 0x6144, 0x6145, 0x6146, 0x6147, 0x6148,
01548 0x6149, 0x614a, 0x614b, 0x614c, 0x614d, 0x614e, 0x614f, 0x6150,
01549 0x6151, 0x6152, 0x6153, 0x6154, 0x6155, 0x6156, 0x6157, 0x6158,
01550 0x6159, 0x615a, 0x615b, 0x615c, 0x615d, 0x615e, 0x615f, 0x6160,
01551 0x6161, 0x6162, 0x6163, 0x6164, 0x6165, 0x6166, 0x6167, 0x6168,
01552 0x6169, 0x616a, 0x616b, 0x616c, 0x616d, 0x616e, 0x616f, 0x6170,
01553 0x6171, 0x6172, 0x6173, 0x6174, 0x6175, 0x6176, 0x6177, 0x6178,
01554 0x6179, 0x617a, 0x617b, 0x617c, 0x617d, 0x617e, 0x6221, 0x6222,
01555 0x6223, 0x6224, 0x6225, 0x6226, 0x6227, 0x6228, 0x6229, 0x622a,
01556 0x622b, 0x622c, 0x622d, 0x622e, 0x622f, 0x6230, 0x6231, 0x6232,
01557 0x6233, 0x6234, 0x6235, 0x6236, 0x6237, 0x6238, 0x6239, 0x623a,
01558 0x623b, 0x623c, 0x623d, 0x623e, 0x623f, 0x6240, 0x6241, 0x6242,
01559 0x6243, 0x6244, 0x6245, 0x6246, 0x6247, 0x6248, 0x6249, 0x624a,
01560 0x624b, 0x624c, 0x624d, 0x624e, 0x624f, 0x6250, 0x6251, 0x6252,
01561 0x6253, 0x6254, 0x6255, 0x6256, 0x6257, 0x6258, 0x6259, 0x625a,
01562 0x625b, 0x625c, 0x625d, 0x625e, 0x625f, 0x6260, 0x6261, 0x6262,
01563 0x6263, 0x6264, 0x6265, 0x6266, 0x6267, 0x6268, 0x6269, 0x626a,
01564 0x626b, 0x626c, 0x626d, 0x626e, 0x626f, 0x6270, 0x6271, 0x6272,
01565 0x6273, 0x6274, 0x6275, 0x6276, 0x6277, 0x6278, 0x6279, 0x627a,
01566 0x627b, 0x627c, 0x627d, 0x627e, 0x627f, 0x6321, 0x6322, 0x6323,
```

01567 0x6324, 0x6325, 0x6326, 0x6327, 0x6328, 0x6329, 0x632a, 0x632b,
01568 0x632c, 0x632d, 0x632e, 0x632f, 0x6330, 0x6331, 0x6332, 0x6333,
01569 0x6334, 0x6335, 0x6336, 0x6337, 0x6338, 0x6339, 0x633a, 0x633b,
01570 0x633c, 0x633d, 0x633e, 0x633f, 0x6340, 0x6341, 0x6342, 0x6343,
01571 0x6344, 0x6345, 0x6346, 0x6347, 0x6348, 0x6349, 0x634a, 0x634b,
01572 0x634c, 0x634d, 0x634e, 0x634f, 0x6350, 0x6351, 0x6352, 0x6353,
01573 0x6354, 0x6355, 0x6356, 0x6357, 0x6358, 0x6359, 0x635a, 0x635b,
01574 0x635c, 0x635d, 0x635e, 0x635f, 0x6360, 0x6361, 0x6362, 0x6363,
01575 0x6364, 0x6365, 0x6366, 0x6367, 0x6368, 0x6369, 0x636a, 0x636b,
01576 0x636c, 0x636d, 0x636e, 0x636f, 0x6370, 0x6371, 0x6372, 0x6373,
01577 0x6374, 0x6375, 0x6376, 0x6377, 0x6378, 0x6379, 0x637a, 0x637b,
01578 0x637c, 0x637d, 0x637e, 0x6421, 0x6422, 0x6423, 0x6424, 0x6425,
01579 0x6426, 0x6427, 0x6428, 0x6429, 0x642a, 0x642b, 0x642c, 0x642d,
01580 0x642e, 0x642f, 0x6430, 0x6431, 0x6432, 0x6433, 0x6434, 0x6435,
01581 0x6436, 0x6437, 0x6438, 0x6439, 0x643a, 0x643b, 0x643c, 0x643d,
01582 0x643e, 0x643f, 0x6440, 0x6441, 0x6442, 0x6443, 0x6444, 0x6445,
01583 0x6446, 0x6447, 0x6448, 0x6449, 0x644a, 0x644b, 0x644c, 0x644d,
01584 0x644e, 0x644f, 0x6450, 0x6451, 0x6452, 0x6453, 0x6454, 0x6455,
01585 0x6456, 0x6457, 0x6458, 0x6459, 0x645a, 0x645b, 0x645c, 0x645d,
01586 0x645e, 0x645f, 0x6460, 0x6461, 0x6462, 0x6463, 0x6464, 0x6465,
01587 0x6466, 0x6467, 0x6468, 0x6469, 0x646a, 0x646b, 0x646c, 0x646d,
01588 0x646e, 0x646f, 0x6470, 0x6471, 0x6472, 0x6473, 0x6474, 0x6475,
01589 0x6476, 0x6477, 0x6478, 0x6479, 0x647a, 0x647b, 0x647c, 0x647d,
01590 0x647e, 0x6521, 0x6522, 0x6523, 0x6524, 0x6525, 0x6526, 0x6527,
01591 0x6528, 0x6529, 0x652a, 0x652b, 0x652c, 0x652d, 0x652e, 0x652f,
01592 0x6530, 0x6531, 0x6532, 0x6533, 0x6534, 0x6535, 0x6536, 0x6537,
01593 0x6538, 0x6539, 0x653a, 0x653b, 0x653c, 0x653d, 0x653e, 0x653f,
01594 0x6540, 0x6541, 0x6542, 0x6543, 0x6544, 0x6545, 0x6546, 0x6547,
01595 0x6548, 0x6549, 0x654a, 0x654b, 0x654c, 0x654d, 0x654e, 0x654f,
01596 0x6550, 0x6551, 0x6552, 0x6553, 0x6554, 0x6555, 0x6556, 0x6557,
01597 0x6558, 0x6559, 0x655a, 0x655b, 0x655c, 0x655d, 0x655e, 0x655f,
01598 0x6560, 0x6561, 0x6562, 0x6563, 0x6564, 0x6565, 0x6566, 0x6567,
01599 0x6568, 0x6569, 0x656a, 0x656b, 0x656c, 0x656d, 0x656e, 0x656f,
01600 0x6570, 0x6571, 0x6572, 0x6573, 0x6574, 0x6575, 0x6576, 0x6577,
01601 0x6578, 0x6579, 0x657a, 0x657b, 0x657c, 0x657d, 0x657e, 0x657f,
01602 0x6580, 0x6581, 0x6582, 0x6583, 0x6584, 0x6585, 0x6586, 0x6587,
01603 0x6588, 0x6589, 0x658a, 0x658b, 0x658c, 0x658d, 0x658e, 0x658f,
01604 0x6590, 0x6591, 0x6592, 0x6593, 0x6594, 0x6595, 0x6596, 0x6597,
01605 0x6598, 0x6599, 0x659a, 0x659b, 0x659c, 0x659d, 0x659e, 0x659f,
01606 0x6600, 0x6601, 0x6602, 0x6603, 0x6604, 0x6605, 0x6606, 0x6607,
01607 0x6608, 0x6609, 0x660a, 0x660b, 0x660c, 0x660d, 0x660e, 0x660f,
01608 0x6610, 0x6611, 0x6612, 0x6613, 0x6614, 0x6615, 0x6616, 0x6617,
01609 0x6618, 0x6619, 0x661a, 0x661b, 0x661c, 0x661d, 0x661e, 0x661f,
01610 0x6620, 0x6621, 0x6622, 0x6623, 0x6624, 0x6625, 0x6626, 0x6627,
01611 0x6628, 0x6629, 0x662a, 0x662b, 0x662c, 0x662d, 0x662e, 0x662f,
01612 0x6630, 0x6631, 0x6632, 0x6633, 0x6634, 0x6635, 0x6636, 0x6637,
01613 0x6638, 0x6639, 0x663a, 0x663b, 0x663c, 0x663d, 0x663e, 0x663f,
01614 0x6640, 0x6641, 0x6642, 0x6643, 0x6644, 0x6645, 0x6646, 0x6647,
01615 0x6648, 0x6649, 0x664a, 0x664b, 0x664c, 0x664d, 0x664e, 0x664f,
01616 0x6650, 0x6651, 0x6652, 0x6653, 0x6654, 0x6655, 0x6656, 0x6657,
01617 0x6658, 0x6659, 0x665a, 0x665b, 0x665c, 0x665d, 0x665e, 0x665f,
01618 0x6660, 0x6661, 0x6662, 0x6663, 0x6664, 0x6665, 0x6666, 0x6667,
01619 0x6668, 0x6669, 0x666a, 0x666b, 0x666c, 0x666d, 0x666e, 0x666f,
01620 0x6670, 0x6671, 0x6672, 0x6673, 0x6674, 0x6675, 0x6676, 0x6677,
01621 0x6678, 0x6679, 0x667a, 0x667b, 0x667c, 0x667d, 0x667e, 0x667f,
01622 0x6680, 0x6681, 0x6682, 0x6683, 0x6684, 0x6685, 0x6686, 0x6687,
01623 0x6688, 0x6689, 0x668a, 0x668b, 0x668c, 0x668d, 0x668e, 0x668f,
01624 0x6690, 0x6691, 0x6692, 0x6693, 0x6694, 0x6695, 0x6696, 0x6697,
01625 0x6698, 0x6699, 0x669a, 0x669b, 0x669c, 0x669d, 0x669e, 0x669f,
01626 0x6700, 0x6701, 0x6702, 0x6703, 0x6704, 0x6705, 0x6706, 0x6707,
01627 0x6708, 0x6709, 0x670a, 0x670b, 0x670c, 0x670d, 0x670e, 0x670f,
01628 0x6710, 0x6711, 0x6712, 0x6713, 0x6714, 0x6715, 0x6716, 0x6717,
01629 0x6718, 0x6719, 0x671a, 0x671b, 0x671c, 0x671d, 0x671e, 0x671f,
01630 0x6720, 0x6721, 0x6722, 0x6723, 0x6724, 0x6725, 0x6726, 0x6727,
01631 0x6728, 0x6729, 0x672a, 0x672b, 0x672c, 0x672d, 0x672e, 0x672f,
01632 0x6730, 0x6731, 0x6732, 0x6733, 0x6734, 0x6735, 0x6736, 0x6737,
01633 0x6738, 0x6739, 0x673a, 0x673b, 0x673c, 0x673d, 0x673e, 0x673f,
01634 0x6740, 0x6741, 0x6742, 0x6743, 0x6744, 0x6745, 0x6746, 0x6747,
01635 0x6748, 0x6749, 0x674a, 0x674b, 0x674c, 0x674d, 0x674e, 0x674f,
01636 0x6750, 0x6751, 0x6752, 0x6753, 0x6754, 0x6755, 0x6756, 0x6757,
01637 0x6758, 0x6759, 0x675a, 0x675b, 0x675c, 0x675d, 0x675e, 0x675f,
01638 0x6760, 0x6761, 0x6762, 0x6763, 0x6764, 0x6765, 0x6766, 0x6767,
01639 0x6768, 0x6769, 0x676a, 0x676b, 0x676c, 0x676d, 0x676e, 0x676f,
01640 0x6770, 0x6771, 0x6772, 0x6773, 0x6774, 0x6775, 0x6776, 0x6777,
01641 0x6778, 0x6779, 0x677a, 0x677b, 0x677c, 0x677d, 0x677e, 0x677f,
01642 0x6780, 0x6781, 0x6782, 0x6783, 0x6784, 0x6785, 0x6786, 0x6787,
01643 0x6788, 0x6789, 0x678a, 0x678b, 0x678c, 0x678d, 0x678e, 0x678f,
01644 0x6790, 0x6791, 0x6792, 0x6793, 0x6794, 0x6795, 0x6796, 0x6797,
01645 0x6798, 0x6799, 0x679a, 0x679b, 0x679c, 0x679d, 0x679e, 0x679f,
01646 0x6800, 0x6801, 0x6802, 0x6803, 0x6804, 0x6805, 0x6806, 0x6807,
01647 0x6808, 0x6809, 0x680a, 0x680b, 0x680c, 0x680d, 0x680e, 0x680f,
01648 0x6810, 0x6811, 0x6812, 0x6813, 0x6814, 0x6815, 0x6816, 0x6817,
01649 0x6818, 0x6819, 0x681a, 0x681b, 0x681c, 0x681d, 0x681e, 0x681f,
01650 0x6820, 0x6821, 0x6822, 0x6823, 0x6824, 0x6825, 0x6826, 0x6827,
01651 0x6828, 0x6829, 0x682a, 0x682b, 0x682c, 0x682d, 0x682e, 0x682f,
01652 0x6830, 0x6831, 0x6832, 0x6833, 0x6834, 0x6835, 0x6836, 0x6837,
01653 0x6838, 0x6839, 0x683a, 0x683b, 0x683c, 0x683d, 0x683e, 0x683f,

```

01654 0x6a4b, 0x6a4c, 0x6a4d, 0x6a4e, 0x6a4f, 0x6a50, 0x6a51, 0x6a52,
01655 0x6a53, 0x6a54, 0x6a55, 0x6a56, 0x6a57, 0x6a58, 0x6a59, 0x6a5a,
01656 0x6a5b, 0x6a5c, 0x6a5d, 0x6a5e, 0x6a5f, 0x6a60, 0x6a61, 0x6a62,
01657 0x6a63, 0x6a64, 0x6a65, 0x6a66, 0x6a67, 0x6a68, 0x6a69, 0x6a6a,
01658 0x6a6b, 0x6a6c, 0x6a6d, 0x6a6e, 0x6a6f, 0x6a70, 0x6a71, 0x6a72,
01659 0x6a73, 0x6a74, 0x6a75, 0x6a76, 0x6a77, 0x6a78, 0x6a79, 0x6a7a,
01660 0x6a7b, 0x6a7c, 0x6a7d, 0x6a7e, 0x6b21, 0x6b22, 0x6b23, 0x6b24,
01661 0x6b25, 0x6b26, 0x6b27, 0x6b28, 0x6b29, 0x6b2a, 0x6b2b, 0x6b2c,
01662 0x6b2d, 0x6b2e, 0x6b2f, 0x6b30, 0x6b31, 0x6b32, 0x6b33, 0x6b34,
01663 0x6b35, 0x6b36, 0x6b37, 0x6b38, 0x6b39, 0x6b3a, 0x6b3b, 0x6b3c,
01664 0x6b3d, 0x6b3e, 0x6b3f, 0x6b40, 0x6b41, 0x6b42, 0x6b43, 0x6b44,
01665 0x6b45, 0x6b46, 0x6b47, 0x6b48, 0x6b49, 0x6b50, 0x6b4a, 0x6b4b,
01666 0x6b4c, 0x6b4d, 0x6b52, 0x6b4e, 0x6b4f, 0x6b51, 0x6b53, 0x6b54,
01667 0x6b55, 0x6b56, 0x6b57, 0x6b58, 0x6b59, 0x6b5a, 0x6b5b, 0x6b5c,
01668 0x6b5e, 0x6b5d, 0x6b5f, 0x6b60, 0x6b61, 0x6b62, 0x6b63, 0x6b64,
01669 0x6b65, 0x6b66, 0x6b67, 0x6b68, 0x6b69, 0x6b6a, 0x6b6b, 0x6b6d,
01670 0x6b6e, 0x6b6f, 0x6b6c, 0x6b70, 0x6b71, 0x6b72, 0x6b73, 0x6b74,
01671 0x6b76, 0x6b75, 0x6b77, 0x6b78, 0x6b79, 0x6b7a, 0x6b7b, 0x6b7c,
01672 0x6b7d, 0x6b7e, 0x6c21, 0x6c22, 0x6c23, 0x6c24, 0x6c25, 0x6c26,
01673 0x6c27, 0x6c28, 0x6c29, 0x6c2a, 0x6c2b, 0x6c2c, 0x6c2d, 0x6c2e,
01674 0x6c2f, 0x6c30, 0x6c31, 0x6c32, 0x6c33, 0x6c34, 0x6c35, 0x6c36,
01675 0x6c37, 0x6c38, 0x6c39, 0x6c3a, 0x6c3b, 0x6c3c, 0x6c3d, 0x6c3e,
01676 0x6c3f, 0x6c40, 0x6c41, 0x6c42, 0x6c43, 0x6c44, 0x6c45, 0x6c46,
01677 0x6c47, 0x6c48, 0x6c49, 0x6c4a, 0x6c4b, 0x6c4c, 0x6c4e, 0x6c4f,
01678 0x6c4d, 0x6c50, 0x6c51, 0x6c52, 0x6c53, 0x6c54, 0x6c55, 0x6c56,
01679 0x6c57, 0x6c58, 0x6c59, 0x6c5a, 0x6c5b, 0x6c5c, 0x6c5d, 0x6c5e,
01680 0x6c5f, 0x6c60, 0x6c61, 0x6c62, 0x6c63, 0x6c64, 0x6c65, 0x6c66,
01681 0x6c67, 0x6c68, 0x6c69, 0x6c6a, 0x6c6b, 0x6c6c, 0x6c6d, 0x6c6e,
01682 0x6c6f, 0x6c70, 0x6c71, 0x6c72, 0x6c73, 0x6c74, 0x6c75, 0x6c76,
01683 0x6c77, 0x6c78, 0x6c79, 0x6c7a, 0x6c7b, 0x6c7c, 0x6c7d, 0x6c7e,
01684 0x6d21, 0x6d22, 0x6d23, 0x6d24, 0x6d25, 0x6d26, 0x6d27, 0x6d28,
01685 0x6d29, 0x6d2a, 0x6d2b, 0x6d2c, 0x6d2d, 0x6d2e, 0x6d2f, 0x6d30,
01686 0x6d31, 0x6d32, 0x6d33, 0x6d34, 0x6d35, 0x6d36, 0x6d37, 0x6d38,
01687 0x6d39, 0x6d3a, 0x6d3b, 0x6d3c, 0x6d3d, 0x6d3e, 0x6d3f, 0x6d40,
01688 0x6d41, 0x6d42, 0x6d43, 0x6d44, 0x6d45, 0x6d46, 0x6d47, 0x6d48,
01689 0x6d49, 0x6d4a, 0x6d4b, 0x6d4c, 0x6d4d, 0x6d4e, 0x6d4f, 0x6d50,
01690 0x6d51, 0x6d52, 0x6d53, 0x6d54, 0x6d55, 0x6d56, 0x6d57, 0x6d58,
01691 0x6d59, 0x6d5a, 0x6d5b, 0x6d5c, 0x6d5d, 0x6d5e, 0x6d5f, 0x6d60,
01692 0x6d61, 0x6d62, 0x6d63,
01693 };
01694
01695 static const Summary16 jisx0212_uni2indx_page00[70] = {
01696 /* 0x0000 */
01697 { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 },
01698 { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x4000 },
01699 { 1, 0x0000 }, { 1, 0x0000 }, { 1, 0xc652 }, { 8, 0x8500 },
01700 { 11, 0xffff }, { 27, 0xff7e }, { 41, 0xffff }, { 57, 0xff7f },
01701 /* 0x0100 */
01702 { 72, 0xffff }, { 88, 0xffcf }, { 102, 0xcff7 }, { 115, 0xffff },
01703 { 131, 0x3fff }, { 145, 0xffff }, { 161, 0xffff }, { 177, 0x7fff },
01704 { 192, 0x0000 }, { 192, 0x0000 }, { 192, 0x0000 }, { 192, 0x0000 },
01705 { 192, 0xe000 }, { 195, 0x1fff }, { 208, 0x0000 }, { 208, 0x0020 },
01706 /* 0x0200 */
01707 { 209, 0x0000 }, { 209, 0x0000 }, { 209, 0x0000 }, { 209, 0x0000 },
01708 { 209, 0x0000 }, { 209, 0x0000 }, { 209, 0x0000 }, { 209, 0x0000 },
01709 { 209, 0x0000 }, { 209, 0x0000 }, { 209, 0x0000 }, { 209, 0x0000 },
01710 { 209, 0x0080 }, { 210, 0x2f00 }, { 215, 0x0000 }, { 215, 0x0000 },
01711 /* 0x0300 */
01712 { 215, 0x0000 }, { 215, 0x0000 }, { 215, 0x0000 }, { 215, 0x0000 },
01713 { 215, 0x0000 }, { 215, 0x0000 }, { 215, 0x0000 }, { 215, 0x0000 },
01714 { 215, 0xd770 }, { 224, 0x0001 }, { 225, 0xfc00 }, { 231, 0x0001 },
01715 { 232, 0x7c04 }, { 238, 0x0000 }, { 238, 0x0000 }, { 238, 0x0000 },
01716 /* 0x0400 */
01717 { 238, 0xdffc }, { 251, 0x0000 }, { 251, 0x0000 }, { 251, 0x0000 },
01718 { 251, 0x0000 }, { 251, 0xdffc },
01719 };
01720 static const Summary16 jisx0212_uni2indx_page21[3] = {
01721 /* 0x2100 */
01722 { 264, 0x0000 }, { 264, 0x0040 }, { 265, 0x0004 },
01723 };
01724 static const Summary16 jisx0212_uni2indx_page4e[1307] = {
01725 /* 0x4e00 */
01726 { 266, 0x1034 }, { 270, 0x8004 }, { 272, 0xc918 }, { 278, 0x0021 },
01727 { 280, 0x0093 }, { 284, 0x1402 }, { 287, 0x0308 }, { 290, 0x8230 },
01728 { 294, 0x2000 }, { 295, 0x20c0 }, { 298, 0x8000 }, { 299, 0x0200 },
01729 { 300, 0x0008 }, { 301, 0x0c01 }, { 304, 0x8107 }, { 309, 0xe02a },
01730 /* 0x4f00 */
01731 { 315, 0x190d }, { 321, 0x02e4 }, { 326, 0x4000 }, { 327, 0x4aaa },
01732 { 334, 0x1b05 }, { 340, 0x8154 }, { 345, 0x5409 }, { 350, 0x6782 },
01733 { 357, 0x5636 }, { 365, 0xc69d }, { 374, 0x0000 }, { 374, 0x7a84 },
01734 { 381, 0xbb63 }, { 391, 0x1004 }, { 393, 0x0005 }, { 395, 0xb005 },
01735 /* 0x5000 */
01736 { 400, 0x5493 }, { 407, 0x7989 }, { 415, 0x4084 }, { 418, 0x082d },
01737 { 423, 0x5467 }, { 431, 0x828e }, { 437, 0x24cd }, { 444, 0x0003 },
01738 { 446, 0xc45a }, { 453, 0xd85d }, { 462, 0x8407 }, { 467, 0x2601 },
01739 { 471, 0x5099 }, { 477, 0xb119 }, { 484, 0x8354 }, { 490, 0x4446 },
01740 /* 0x5100 */

```



```
01741 { 495, 0x79c8 }, { 503, 0x7a81 }, { 510, 0xb188 }, { 516, 0x033a },
01742 { 522, 0x8404 }, { 525, 0x81a8 }, { 530, 0x0050 }, { 532, 0x4000 },
01743 { 533, 0x4818 }, { 537, 0x2100 }, { 539, 0x200a }, { 542, 0xd500 },
01744 { 547, 0x8104 }, { 550, 0x412e }, { 556, 0x4024 }, { 559, 0x009c },
01745 /* 0x5200 */
01746 { 563, 0x0026 }, { 566, 0x016c }, { 571, 0x0104 }, { 573, 0x1026 },
01747 { 577, 0x0220 }, { 579, 0x95a0 }, { 585, 0x4043 }, { 589, 0x0380 },
01748 { 592, 0x1425 }, { 597, 0x15e8 }, { 604, 0x80f0 }, { 609, 0x2dc1 },
01749 { 616, 0x9151 }, { 622, 0x1852 }, { 627, 0x1722 }, { 633, 0x00d3 },
01750 /* 0x5300 */
01751 { 638, 0x1c09 }, { 643, 0xd90a }, { 650, 0x3ba0 }, { 657, 0x7025 },
01752 { 663, 0x1804 }, { 666, 0x0a00 }, { 668, 0x302a }, { 673, 0x4204 },
01753 { 676, 0x4188 }, { 680, 0x2218 }, { 684, 0x8c12 }, { 689, 0x25b4 },
01754 { 696, 0x8021 }, { 699, 0x642c }, { 705, 0x00c1 }, { 708, 0x0020 },
01755 /* 0x5400 */
01756 { 709, 0x0004 }, { 710, 0x0408 }, { 712, 0x8582 }, { 717, 0x0032 },
01757 { 720, 0xa098 }, { 725, 0x4000 }, { 726, 0x6ad4 }, { 734, 0x8010 },
01758 { 736, 0x232a }, { 742, 0x9062 }, { 747, 0x66c2 }, { 754, 0x8e82 },
01759 { 760, 0x6440 }, { 764, 0x0000 }, { 764, 0x9401 }, { 768, 0xd040 },
01760 /* 0x5500 */
01761 { 772, 0x7323 }, { 780, 0x0020 }, { 781, 0x0c00 }, { 783, 0x3864 },
01762 { 789, 0x2682 }, { 794, 0x4d03 }, { 800, 0x0053 }, { 804, 0x8000 },
01763 { 805, 0xc146 }, { 811, 0x009e }, { 816, 0x2018 }, { 819, 0x8004 },
01764 { 821, 0x5a4a }, { 828, 0x498e }, { 835, 0x0204 }, { 837, 0x8040 },
01765 /* 0x5600 */
01766 { 839, 0xe520 }, { 845, 0x0207 }, { 849, 0x1000 }, { 850, 0xbaa9 },
01767 { 859, 0xaa5b }, { 868, 0x4010 }, { 870, 0xa24f }, { 878, 0x0026 },
01768 { 881, 0x1930 }, { 886, 0xe620 }, { 892, 0x3bc0 }, { 899, 0x408a },
01769 { 903, 0x8be20 }, { 910, 0xb201 }, { 915, 0x29f2 }, { 923, 0x00c2 },
01770 /* 0x5700 */
01771 { 926, 0x1486 }, { 931, 0x2c22 }, { 936, 0xd63d }, { 946, 0xe018 },
01772 { 951, 0x3060 }, { 955, 0x0004 }, { 956, 0xe9a4 }, { 964, 0x5ebb },
01773 { 975, 0x100a }, { 978, 0xf6b0 }, { 987, 0x1382 }, { 992, 0x2100 },
01774 { 994, 0x9180 }, { 998, 0x6020 }, { 1001, 0x22d2 }, { 1007, 0xe161 },
01775 /* 0x5800 */
01776 { 1014, 0x3318 }, { 1020, 0xc800 }, { 1023, 0x20c1 }, { 1027, 0x8204 },
01777 { 1030, 0xb200 }, { 1034, 0x8021 }, { 1037, 0x0192 }, { 1041, 0x9100 },
01778 { 1044, 0xb783 }, { 1053, 0x2051 }, { 1057, 0x0247 }, { 1062, 0x1006 },
01779 { 1065, 0x6114 }, { 1070, 0x2455 }, { 1076, 0x0206 }, { 1079, 0x0008 },
01780 /* 0x5900 */
01781 { 1080, 0x1860 }, { 1084, 0x201c }, { 1088, 0x811a }, { 1093, 0x8069 },
01782 { 1098, 0x0048 }, { 1100, 0xea0c }, { 1107, 0xa80a }, { 1112, 0x1a64 },
01783 { 1118, 0x5800 }, { 1121, 0x80a4 }, { 1125, 0xe090 }, { 1130, 0x1489 },
01784 { 1135, 0x251a }, { 1141, 0xe004 }, { 1145, 0xc098 }, { 1150, 0x0096 },
01785 /* 0x5a00 */
01786 { 1154, 0x7011 }, { 1159, 0x400c }, { 1162, 0x2598 }, { 1168, 0x0001 },
01787 { 1169, 0x11b0 }, { 1174, 0x4021 }, { 1177, 0x20a8 }, { 1181, 0x4c80 },
01788 { 1185, 0x0800 }, { 1186, 0xd249 }, { 1193, 0x1085 }, { 1197, 0x8d2e },
01789 { 1205, 0x8150 }, { 1209, 0x1400 }, { 1211, 0x4421 }, { 1215, 0x2060 },
01790 /* 0x5b00 */
01791 { 1218, 0x0103 }, { 1221, 0x2a80 }, { 1225, 0x2022 }, { 1228, 0x0110 },
01792 { 1230, 0x1802 }, { 1233, 0x4044 }, { 1236, 0xc100 }, { 1239, 0xf000 },
01793 { 1243, 0x4452 }, { 1248, 0x005b }, { 1253, 0xb300 }, { 1258, 0x1486 },
01794 { 1263, 0xa003 }, { 1267, 0x07c0 }, { 1272, 0x8001 }, { 1274, 0x2012 },
01795 /* 0x5c00 */
01796 { 1277, 0x1000 }, { 1278, 0xc080 }, { 1281, 0x5a48 }, { 1287, 0x0065 },
01797 { 1291, 0x0000 }, { 1291, 0x1600 }, { 1294, 0x238c }, { 1300, 0x3c31 },
01798 { 1307, 0x8580 }, { 1311, 0xa004 }, { 1314, 0x044d }, { 1319, 0x0434 },
01799 { 1323, 0x0a00 }, { 1325, 0x2084 }, { 1328, 0x4000 }, { 1329, 0x0016 },
01800 /* 0x5d00 */
01801 { 1332, 0x2042 }, { 1335, 0x0004 }, { 1336, 0x08d8 }, { 1341, 0xa212 },
01802 { 1346, 0x054c }, { 1351, 0x8222 }, { 1355, 0x2417 }, { 1361, 0xc601 },
01803 { 1366, 0x050a }, { 1370, 0x8a3c }, { 1377, 0x0881 }, { 1380, 0x0315 },
01804 { 1385, 0x4888 }, { 1389, 0x0301 }, { 1392, 0x0211 }, { 1395, 0x0300 },
01805 /* 0x5e00 */
01806 { 1397, 0x2081 }, { 1400, 0x8134 }, { 1405, 0x4101 }, { 1408, 0x4024 },
01807 { 1411, 0x0a00 }, { 1413, 0x5943 }, { 1420, 0x7d00 }, { 1426, 0x0001 },
01808 { 1427, 0x4801 }, { 1430, 0x0000 }, { 1430, 0x1534 }, { 1436, 0xe00a },
01809 { 1441, 0x5840 }, { 1445, 0x5036 }, { 1451, 0x0820 }, { 1453, 0x0000 },
01810 /* 0x5f00 */
01811 { 1453, 0x41c4 }, { 1458, 0x3200 }, { 1461, 0x591e }, { 1469, 0xa851 },
01812 { 1475, 0x20b1 }, { 1480, 0x0911 }, { 1484, 0x8099 }, { 1489, 0x6534 },
01813 { 1496, 0xa200 }, { 1499, 0x3040 }, { 1502, 0x9894 }, { 1508, 0x0103 },
01814 { 1511, 0x0b90 }, { 1516, 0x401f }, { 1522, 0xf706 }, { 1531, 0x144c },
01815 /* 0x6000 */
01816 { 1536, 0x2480 }, { 1539, 0x8598 }, { 1545, 0x2010 }, { 1547, 0x0028 },
01817 { 1549, 0x1381 }, { 1554, 0x20d2 }, { 1559, 0x0082 }, { 1561, 0xc002 },
01818 { 1564, 0x4544 }, { 1569, 0x612a }, { 1575, 0x0134 }, { 1579, 0x4883 },
01819 { 1584, 0xcfc14 }, { 1592, 0x6a30 }, { 1598, 0x0024 }, { 1600, 0x3124 },
01820 /* 0x6100 */
01821 { 1605, 0x1484 }, { 1609, 0x52df }, { 1619, 0x0c04 }, { 1622, 0x02e3 },
01822 { 1628, 0x0262 }, { 1632, 0x4000 }, { 1633, 0x1001 }, { 1635, 0x9904 },
01823 { 1640, 0x281b }, { 1646, 0xb18c }, { 1653, 0x2521 }, { 1658, 0x1300 },
01824 { 1661, 0xc007 }, { 1666, 0xf020 }, { 1671, 0xb2a6 }, { 1679, 0x0000 },
01825 /* 0x6200 */
01826 { 1679, 0x009a }, { 1683, 0x1028 }, { 1686, 0x0a8d }, { 1692, 0x2200 },
01827 { 1694, 0x105c }, { 1699, 0x1457 }, { 1706, 0xa010 }, { 1709, 0x2408 },
```

```

01828 { 1712, 0xe000 }, { 1715, 0x0001 }, { 1716, 0x0140 }, { 1718, 0xc4c8 },
01829 { 1724, 0x4010 }, { 1726, 0x0460 }, { 1729, 0x0400 }, { 1730, 0x3014 },
01830 /* 0x6300 */
01831 { 1734, 0x2c18 }, { 1739, 0x0149 }, { 1743, 0x2600 }, { 1746, 0x1260 },
01832 { 1750, 0x4c5e }, { 1758, 0x091c }, { 1763, 0x3060 }, { 1767, 0xb132 },
01833 { 1774, 0x0494 }, { 1778, 0x4631 }, { 1784, 0xe050 }, { 1789, 0x2000 },
01834 { 1790, 0x4122 }, { 1794, 0x103a }, { 1799, 0x1421 }, { 1803, 0x032c },
01835 /* 0x6400 */
01836 { 1808, 0x0600 }, { 1810, 0x4115 }, { 1815, 0x8635 }, { 1822, 0xa021 },
01837 { 1826, 0x8800 }, { 1828, 0xbc1e }, { 1837, 0x200b }, { 1841, 0x2818 },
01838 { 1845, 0x80a0 }, { 1848, 0xab03 }, { 1855, 0x114a }, { 1860, 0xe008 },
01839 { 1864, 0x5e10 }, { 1870, 0x00a3 }, { 1874, 0x2630 }, { 1879, 0x88a1 },
01840 /* 0x6500 */
01841 { 1884, 0x8712 }, { 1890, 0xca58 }, { 1897, 0x4244 }, { 1901, 0x3402 },
01842 { 1905, 0x0288 }, { 1908, 0x8015 }, { 1912, 0x0881 }, { 1915, 0x2400 },
01843 { 1917, 0x0422 }, { 1920, 0x2124 }, { 1924, 0x4049 }, { 1928, 0x801c },
01844 { 1932, 0x4304 }, { 1936, 0x8151 }, { 1941, 0x0000 }, { 1941, 0xc235 },
01845 /* 0x6600 */
01846 { 1948, 0x2311 }, { 1953, 0x6066 }, { 1959, 0x5e5e }, { 1969, 0x028b },
01847 { 1974, 0x5461 }, { 1980, 0x1b82 }, { 1986, 0x1c03 }, { 1991, 0xdba8 },
01848 { 2000, 0x3801 }, { 2004, 0x9e05 }, { 2011, 0x2011 }, { 2014, 0x8826 },
01849 { 2019, 0xd10d }, { 2026, 0x8810 }, { 2029, 0x5900 }, { 2033, 0x0c00 },
01850 /* 0x6700 */
01851 { 2035, 0x40a0 }, { 2038, 0x1208 }, { 2041, 0x0005 }, { 2043, 0x4008 },
01852 { 2045, 0x11a0 }, { 2049, 0x2030 }, { 2052, 0x5040 }, { 2055, 0x0850 },
01853 { 2058, 0xc012 }, { 2062, 0x0b4a }, { 2068, 0x0000 }, { 2068, 0x3827 },
01854 { 2075, 0x032d }, { 2081, 0x1284 }, { 2085, 0x0042 }, { 2087, 0x02c5 },
01855 /* 0x6800 */
01856 { 2092, 0x0000 }, { 2092, 0xa210 }, { 2096, 0xb180 }, { 2101, 0x880b },
01857 { 2106, 0x1430 }, { 2110, 0x09a4 }, { 2115, 0xc800 }, { 2118, 0x1e27 },
01858 { 2126, 0x0154 }, { 2130, 0x1540 }, { 2134, 0x462a }, { 2140, 0x0804 },
01859 { 2142, 0x9120 }, { 2146, 0x324b }, { 2153, 0x3d20 }, { 2159, 0x3863 },
01860 /* 0x6900 */
01861 { 2166, 0x0640 }, { 2169, 0x00cb }, { 2174, 0x0000 }, { 2174, 0x092a },
01862 { 2179, 0x4224 }, { 2183, 0x0880 }, { 2185, 0x1378 }, { 2192, 0x8c07 },
01863 { 2198, 0x2001 }, { 2200, 0x0144 }, { 2203, 0xa962 }, { 2210, 0x1580 },
01864 { 2214, 0x0120 }, { 2216, 0x00c2 }, { 2219, 0xc024 }, { 2223, 0x402a },
01865 /* 0x6a00 */
01866 { 2227, 0x800b }, { 2231, 0x2422 }, { 2235, 0x0111 }, { 2238, 0xc895 },
01867 { 2245, 0x4660 }, { 2250, 0x0867 }, { 2256, 0x0490 }, { 2259, 0x400a },
01868 { 2262, 0x0aca }, { 2268, 0xe802 }, { 2273, 0x8820 }, { 2276, 0xe013 },
01869 { 2282, 0x1340 }, { 2286, 0x3071 }, { 2292, 0x1090 }, { 2295, 0x3007 },
01870 /* 0x6b00 */
01871 { 2300, 0x82cc }, { 2306, 0x4883 }, { 2311, 0x9910 }, { 2316, 0x8860 },
01872 { 2320, 0x2440 }, { 2323, 0x2144 }, { 2327, 0x4881 }, { 2331, 0x6021 },
01873 { 2335, 0x0024 }, { 2337, 0x8880 }, { 2340, 0x730d }, { 2348, 0x6301 },
01874 { 2353, 0x1218 }, { 2357, 0x0440 }, { 2359, 0x40ca }, { 2364, 0x8282 },
01875 /* 0x6c00 */
01876 { 2368, 0x6234 }, { 2374, 0x8205 }, { 2378, 0x51c0 }, { 2383, 0x8c68 },
01877 { 2389, 0xac00 }, { 2393, 0x1a14 }, { 2398, 0xa880 }, { 2402, 0x0b50 },
01878 { 2407, 0x02e0 }, { 2411, 0x91b0 }, { 2417, 0x0000 }, { 2417, 0x0015 },
01879 { 2420, 0xa044 }, { 2424, 0x1457 }, { 2431, 0x5a81 }, { 2437, 0x0014 },
01880 /* 0x6d00 */
01881 { 2439, 0xc490 }, { 2444, 0x040a }, { 2447, 0xc1c0 }, { 2452, 0x9202 },
01882 { 2456, 0x0000 }, { 2456, 0xc080 }, { 2459, 0x80a2 }, { 2463, 0x1001 },
01883 { 2465, 0x0084 }, { 2467, 0x01d6 }, { 2473, 0x1400 }, { 2475, 0xa290 },
01884 { 2480, 0xc510 }, { 2485, 0xa840 }, { 2489, 0x8225 }, { 2494, 0x1051 },
01885 /* 0x6e00 */
01886 { 2498, 0x0011 }, { 2500, 0x4000 }, { 2501, 0x0084 }, { 2503, 0x1a44 },
01887 { 2508, 0x8b30 }, { 2514, 0x709e }, { 2522, 0x010c }, { 2525, 0x2808 },
01888 { 2528, 0x2000 }, { 2529, 0x0208 }, { 2531, 0x6081 }, { 2535, 0x880a },
01889 { 2539, 0xe58b }, { 2548, 0x0000 }, { 2548, 0x6800 }, { 2551, 0x2a00 },
01890 /* 0x6f00 */
01891 { 2554, 0x3510 }, { 2559, 0x0d40 }, { 2563, 0xa640 }, { 2568, 0x1849 },
01892 { 2573, 0x8000 }, { 2574, 0x668e }, { 2582, 0x1106 }, { 2586, 0x6000 },
01893 { 2588, 0x3988 }, { 2594, 0x845d }, { 2601, 0xc1e1 }, { 2608, 0x1061 },
01894 { 2612, 0x05a0 }, { 2616, 0x4400 }, { 2618, 0x0300 }, { 2620, 0x3221 },
01895 /* 0x7000 */
01896 { 2625, 0x20e1 }, { 2630, 0x0080 }, { 2631, 0x8009 }, { 2634, 0x1290 },
01897 { 2638, 0x4f18 }, { 2645, 0x6030 }, { 2649, 0x5030 }, { 2653, 0x4060 },
01898 { 2656, 0x0062 }, { 2659, 0x09f0 }, { 2665, 0x0810 }, { 2667, 0x0093 },
01899 { 2671, 0x0400 }, { 2672, 0x117a }, { 2679, 0x0010 }, { 2680, 0x0400 },
01900 /* 0x7100 */
01901 { 2681, 0x98f8 }, { 2689, 0x4000 }, { 2690, 0xa801 }, { 2694, 0x0103 },
01902 { 2697, 0x0ce2 }, { 2703, 0x5485 }, { 2709, 0x0101 }, { 2711, 0x0200 },
01903 { 2712, 0x10a1 }, { 2716, 0x0c04 }, { 2719, 0x8005 }, { 2722, 0x840d },
01904 { 2727, 0x1813 }, { 2732, 0x1648 }, { 2737, 0x0000 }, { 2737, 0x4100 },
01905 /* 0x7200 */
01906 { 2739, 0x0381 }, { 2743, 0xa488 }, { 2748, 0x8810 }, { 2751, 0x0310 },
01907 { 2754, 0xc02e }, { 2760, 0x5469 }, { 2767, 0xc909 }, { 2773, 0x9982 },
01908 { 2779, 0x6210 }, { 2783, 0x0808 }, { 2785, 0x6100 }, { 2788, 0x4012 },
01909 { 2791, 0x1282 }, { 2795, 0x8160 }, { 2799, 0x0020 }, { 2800, 0x4c18 },
01910 /* 0x7300 */
01911 { 2805, 0x28b4 }, { 2811, 0x430c }, { 2816, 0x1194 }, { 2821, 0x2c26 },
01912 { 2827, 0x2008 }, { 2829, 0xe145 }, { 2836, 0xdac1 }, { 2844, 0x1282 },
01913 { 2848, 0x406b }, { 2854, 0xd1a9 }, { 2862, 0x2c65 }, { 2869, 0xb2a0 },
01914 { 2875, 0x9a60 }, { 2881, 0x224c }, { 2886, 0x02ca }, { 2891, 0xae0b },

```

```
01915 /* 0x7400 */
01916 { 2899, 0x0493 }, { 2904, 0x0c02 }, { 2907, 0xff50 }, { 2917, 0x0203 },
01917 { 2920, 0x28d9 }, { 2927, 0x2086 }, { 2931, 0x69c4 }, { 2938, 0x0006 },
01918 { 2940, 0x82e3 }, { 2947, 0x9707 }, { 2955, 0xcf4b }, { 2965, 0x8a26 },
01919 { 2971, 0x1300 }, { 2974, 0xcd09 }, { 2981, 0x8d10 }, { 2986, 0x9c10 },
01920 /* 0x7500 */
01921 { 2991, 0x0040 }, { 2992, 0x00c4 }, { 2995, 0x8693 }, { 3002, 0xe240 },
01922 { 3007, 0x4189 }, { 3012, 0xc085 }, { 3017, 0x8002 }, { 3019, 0x7e02 },
01923 { 3026, 0x0022 }, { 3028, 0x122d }, { 3034, 0x0014 }, { 3036, 0x8410 },
01924 { 3039, 0xd053 }, { 3046, 0x9080 }, { 3049, 0xd093 }, { 3056, 0x0202 },
01925 /* 0x7600 */
01926 { 3058, 0x959d }, { 3067, 0x7a6c }, { 3076, 0x2268 }, { 3081, 0x172c },
01927 { 3088, 0x0e3b }, { 3096, 0x8220 }, { 3099, 0xe030 }, { 3104, 0x0012 },
01928 { 3106, 0x3022 }, { 3110, 0xb820 }, { 3115, 0x25fd }, { 3125, 0x2000 },
01929 { 3126, 0x5a22 }, { 3132, 0x0210 }, { 3134, 0x1141 }, { 3138, 0x1243 },
01930 /* 0x7700 */
01931 { 3143, 0x4441 }, { 3147, 0x16b4 }, { 3154, 0xe104 }, { 3159, 0x6270 },
01932 { 3165, 0xe464 }, { 3172, 0xd0c4 }, { 3178, 0x1495 }, { 3184, 0x241d },
01933 { 3190, 0x3011 }, { 3194, 0x8470 }, { 3199, 0xc484 }, { 3204, 0x4022 },
01934 { 3207, 0x0208 }, { 3209, 0xc226 }, { 3215, 0x1451 }, { 3220, 0x0913 },
01935 /* 0x7800 */
01936 { 3225, 0x6260 }, { 3230, 0x2002 }, { 3232, 0x600e }, { 3237, 0x00a1 },
01937 { 3240, 0x5198 }, { 3246, 0x5004 }, { 3249, 0x451b }, { 3256, 0x4400 },
01938 { 3258, 0x8400 }, { 3260, 0xe110 }, { 3265, 0x3112 }, { 3270, 0xa80f },
01939 { 3277, 0x5380 }, { 3282, 0x886c }, { 3288, 0x0453 }, { 3293, 0x8ccc },
01940 /* 0x7900 */
01941 { 3300, 0x1041 }, { 3303, 0xd401 }, { 3308, 0x22a1 }, { 3313, 0xa832 },
01942 { 3319, 0x8c70 }, { 3325, 0x1912 }, { 3330, 0x0a80 }, { 3333, 0x5a04 },
01943 { 3338, 0x1800 }, { 3340, 0x197a }, { 3348, 0x8b02 }, { 3353, 0x0912 },
01944 { 3357, 0x8594 }, { 3363, 0x6450 }, { 3368, 0x2c25 }, { 3374, 0x1102 },
01945 /* 0x7a00 */
01946 { 3377, 0x168c }, { 3383, 0x4822 }, { 3387, 0xa882 }, { 3392, 0x0731 },
01947 { 3398, 0x11b0 }, { 3403, 0xb260 }, { 3409, 0x24a1 }, { 3414, 0x4120 },
01948 { 3417, 0x0c65 }, { 3423, 0x4013 }, { 3427, 0x1009 }, { 3430, 0x1a28 },
01949 { 3435, 0x5240 }, { 3439, 0x0802 }, { 3441, 0x1b00 }, { 3445, 0x6812 },
01950 /* 0x7b00 */
01951 { 3450, 0x0080 }, { 3451, 0x8010 }, { 3453, 0xee88 }, { 3461, 0xa013 },
01952 { 3466, 0x4083 }, { 3470, 0x0020 }, { 3471, 0xa651 }, { 3478, 0x008c },
01953 { 3481, 0x4210 }, { 3484, 0x4843 }, { 3489, 0x9021 }, { 3493, 0x3c65 },
01954 { 3501, 0x0524 }, { 3505, 0x0ed0 }, { 3511, 0x0500 }, { 3513, 0x5734 },
01955 /* 0x7c00 */
01956 { 3521, 0xda5e }, { 3531, 0x0a00 }, { 3533, 0x1161 }, { 3538, 0x065a },
01957 { 3544, 0x0440 }, { 3546, 0x7e2e }, { 3556, 0x628a }, { 3562, 0x3205 },
01958 { 3567, 0x80c0 }, { 3570, 0x4010 }, { 3572, 0x0041 }, { 3574, 0x9cc1 },
01959 { 3581, 0xa390 }, { 3587, 0x26b8 }, { 3594, 0x0a40 }, { 3597, 0x0020 },
01960 /* 0x7d00 */
01961 { 3598, 0x8388 }, { 3603, 0x604e }, { 3609, 0x2448 }, { 3613, 0x7002 },
01962 { 3617, 0x2183 }, { 3622, 0x368a }, { 3629, 0x04a0 }, { 3632, 0x8d01 },
01963 { 3637, 0x396e }, { 3646, 0x60c2 }, { 3651, 0x04c0 }, { 3654, 0x02c8 },
01964 { 3658, 0x707c }, { 3666, 0x0280 }, { 3668, 0x2c64 }, { 3674, 0x0662 },
01965 /* 0x7e00 */
01966 { 3679, 0x0101 }, { 3681, 0x30a3 }, { 3687, 0xb181 }, { 3693, 0x8048 },
01967 { 3696, 0x40b0 }, { 3700, 0x8105 }, { 3704, 0xc826 }, { 3710, 0x4108 },
01968 { 3713, 0x24c2 }, { 3718, 0x6522 }, { 3724, 0x0000 }, { 3724, 0x0000 },
01969 { 3724, 0x0000 }, { 3724, 0x0000 }, { 3724, 0x0000 }, { 3724, 0x0000 },
01970 /* 0x7f00 */
01971 { 3724, 0x0000 }, { 3724, 0x0000 }, { 3724, 0x0000 }, { 3724, 0xf800 },
01972 { 3729, 0x8098 }, { 3733, 0x380c }, { 3738, 0x207a }, { 3744, 0xe002 },
01973 { 3748, 0xa801 }, { 3752, 0x10c3 }, { 3757, 0x2446 }, { 3762, 0x9010 },
01974 { 3765, 0xc109 }, { 3770, 0x8800 }, { 3772, 0xd128 }, { 3778, 0xe404 },
01975 /* 0x8000 */
01976 { 3783, 0xe580 }, { 3789, 0xe05a }, { 3796, 0x5051 }, { 3801, 0x56b1 },
01977 { 3809, 0x0011 }, { 3811, 0x0000 }, { 3811, 0x2051 }, { 3815, 0x0022 },
01978 { 3817, 0x4102 }, { 3820, 0x5000 }, { 3822, 0x08c0 }, { 3825, 0x0300 },
01979 { 3827, 0xa100 }, { 3830, 0x01b4 }, { 3835, 0x6001 }, { 3838, 0x464d },
01980 /* 0x8100 */
01981 { 3845, 0x0808 }, { 3847, 0x51c0 }, { 3852, 0x1091 }, { 3856, 0x1421 },
01982 { 3860, 0x14a0 }, { 3864, 0x0084 }, { 3866, 0xa383 }, { 3873, 0x0080 },
01983 { 3874, 0x4872 }, { 3880, 0x4941 }, { 3885, 0x4004 }, { 3887, 0x0814 },
01984 { 3890, 0xcc28 }, { 3896, 0x68a0 }, { 3901, 0x1812 }, { 3905, 0xa367 },
01985 /* 0x8200 */
01986 { 3914, 0x8009 }, { 3917, 0x2618 }, { 3922, 0x0106 }, { 3925, 0x0414 },
01987 { 3928, 0xc878 }, { 3935, 0x1042 }, { 3938, 0x2089 }, { 3942, 0xa810 },
01988 { 3946, 0x469b }, { 3954, 0x0d52 }, { 3960, 0x479b }, { 3969, 0xd495 },
01989 { 3977, 0x0040 }, { 3978, 0x0421 }, { 3981, 0xa515 }, { 3988, 0x60c0 },
01990 /* 0x8300 */
01991 { 3992, 0x0d83 }, { 3998, 0xe800 }, { 4002, 0x7006 }, { 4007, 0x3489 },
01992 { 4013, 0x609c }, { 4019, 0x00fa }, { 4025, 0x0000 }, { 4025, 0xa101 },
01993 { 4029, 0x2055 }, { 4034, 0x3b34 }, { 4042, 0x32c0 }, { 4047, 0xc000 },
01994 { 4049, 0x8281 }, { 4053, 0x2013 }, { 4057, 0x0500 }, { 4059, 0x1340 },
01995 /* 0x8400 */
01996 { 4063, 0x8442 }, { 4067, 0x0222 }, { 4070, 0x8000 }, { 4071, 0x0200 },
01997 { 4072, 0xa5a0 }, { 4078, 0x1746 }, { 4085, 0x04b1 }, { 4090, 0x3159 },
01998 { 4097, 0x0022 }, { 4099, 0x402c }, { 4103, 0x8740 }, { 4108, 0x6412 },
01999 { 4113, 0x9185 }, { 4119, 0x1008 }, { 4121, 0x8480 }, { 4124, 0x2c87 },
02000 /* 0x8500 */
02001 { 4131, 0x508c }, { 4136, 0x5001 }, { 4139, 0x8cbc }, { 4147, 0x805c },
```

```

02002 { 4152, 0x8040 }, { 4154, 0xf24f }, { 4164, 0x8817 }, { 4170, 0xae00 },
02003 { 4175, 0x9a62 }, { 4182, 0xa108 }, { 4186, 0x20a5 }, { 4191, 0xf1d0 },
02004 { 4199, 0x4c84 }, { 4204, 0x8500 }, { 4207, 0x2141 }, { 4211, 0x9048 },
02005 /* 0x8600 */
02006 { 4215, 0x6031 }, { 4220, 0x4b07 }, { 4227, 0x0282 }, { 4230, 0x3540 },
02007 { 4235, 0x0047 }, { 4239, 0x23cc }, { 4246, 0x921f }, { 4254, 0x04e0 },
02008 { 4258, 0x2100 }, { 4260, 0x1542 }, { 4265, 0x21c2 }, { 4270, 0x83ba },
02009 { 4278, 0x002b }, { 4282, 0x14a6 }, { 4288, 0x00a9 }, { 4292, 0x3400 },
02010 /* 0x8700 */
02011 { 4295, 0xc8b0 }, { 4301, 0xc219 }, { 4307, 0xc10a }, { 4312, 0x7606 },
02012 { 4319, 0x2029 }, { 4323, 0x2100 }, { 4325, 0x8032 }, { 4329, 0x0806 },
02013 { 4332, 0x1bf8 }, { 4341, 0x43a9 }, { 4348, 0x7089 }, { 4354, 0xc022 },
02014 { 4358, 0x4702 }, { 4363, 0x9660 }, { 4369, 0x2c1c }, { 4375, 0x850a },
02015 /* 0x8800 */
02016 { 4380, 0x0e4a }, { 4386, 0xdf1d }, { 4397, 0x6100 }, { 4400, 0x1425 },
02017 { 4405, 0x4f2a }, { 4413, 0x9562 }, { 4420, 0x0211 }, { 4423, 0x0a02 },
02018 { 4426, 0x0001 }, { 4427, 0x9d00 }, { 4432, 0x0501 }, { 4435, 0x6400 },
02019 { 4438, 0x7c01 }, { 4444, 0x480e }, { 4449, 0x8080 }, { 4451, 0x00a3 },
02020 /* 0x8900 */
02021 { 4455, 0xe042 }, { 4460, 0x1760 }, { 4466, 0x01c1 }, { 4470, 0x4627 },
02022 { 4477, 0x8265 }, { 4483, 0x1c84 }, { 4488, 0x480e }, { 4493, 0x3c29 },
02023 { 4500, 0x2200 }, { 4502, 0x9831 }, { 4508, 0x0021 }, { 4510, 0x10f1 },
02024 { 4516, 0x0000 }, { 4516, 0x01f0 }, { 4521, 0x2a20 }, { 4525, 0xa24a },
02025 /* 0x8a00 */
02026 { 4531, 0x80b0 }, { 4535, 0x4036 }, { 4540, 0x9855 }, { 4547, 0x60a0 },
02027 { 4551, 0x62a9 }, { 4558, 0x31c8 }, { 4564, 0x00a2 }, { 4567, 0xcee0 },
02028 { 4575, 0x8849 }, { 4580, 0x82c5 }, { 4586, 0xc280 }, { 4590, 0x48c8 },
02029 { 4595, 0x0748 }, { 4600, 0xa0ba }, { 4607, 0x1000 }, { 4608, 0x9071 },
02030 /* 0x8b00 */
02031 { 4614, 0x0c60 }, { 4618, 0xd002 }, { 4622, 0x2000 }, { 4623, 0x1081 },
02032 { 4626, 0x217c }, { 4633, 0x421c }, { 4638, 0x2008 }, { 4640, 0x5340 },
02033 { 4645, 0xa832 }, { 4651, 0xd030 }, { 4656, 0x0000 }, { 4656, 0x0000 },
02034 { 4656, 0x0000 }, { 4656, 0x0000 }, { 4656, 0x0000 }, { 4656, 0x0000 },
02035 /* 0x8c00 */
02036 { 4656, 0x0000 }, { 4656, 0x0000 }, { 4656, 0x0000 }, { 4656, 0x6300 },
02037 { 4660, 0x8aa0 }, { 4665, 0x2b9a }, { 4673, 0x2358 }, { 4679, 0x4868 },
02038 { 4684, 0x08c0 }, { 4687, 0x1a0d }, { 4693, 0x0010 }, { 4694, 0x0600 },
02039 { 4696, 0x8a60 }, { 4701, 0x2260 }, { 4705, 0x9102 }, { 4709, 0xc1a5 },
02040 /* 0x8d00 */
02041 { 4716, 0x020a }, { 4719, 0x0884 }, { 4722, 0x0000 }, { 4722, 0x0000 },
02042 { 4722, 0x0000 }, { 4722, 0x0000 }, { 4722, 0x5220 }, { 4726, 0x8000 },
02043 { 4727, 0x2114 }, { 4731, 0xc023 }, { 4736, 0x9841 }, { 4741, 0x1aa4 },
02044 { 4747, 0x45e1 }, { 4754, 0x02b2 }, { 4759, 0x10b0 }, { 4763, 0x2017 },
02045 /* 0x8e00 */
02046 { 4768, 0x0872 }, { 4773, 0x0052 }, { 4776, 0x00cf }, { 4782, 0x23ca },
02047 { 4789, 0xe803 }, { 4795, 0x7810 }, { 4800, 0xb206 }, { 4806, 0x0e03 },
02048 { 4811, 0x020c }, { 4814, 0x6c25 }, { 4821, 0x6284 }, { 4826, 0x0c28 },
02049 { 4830, 0x809b }, { 4836, 0x1012 }, { 4839, 0x6100 }, { 4842, 0x0683 },
02050 /* 0x8f00 */
02051 { 4847, 0x8185 }, { 4852, 0x41c1 }, { 4857, 0x71ab }, { 4866, 0x04f0 },
02052 { 4871, 0x808b }, { 4876, 0x613e }, { 4884, 0x0020 }, { 4885, 0x0000 },
02053 { 4885, 0x0000 }, { 4885, 0x2000 }, { 4886, 0x0073 }, { 4891, 0x4160 },
02054 { 4895, 0x2c43 }, { 4901, 0x002d }, { 4905, 0x4119 }, { 4910, 0x4862 },
02055 /* 0x9000 */
02056 { 4915, 0x1114 }, { 4919, 0x0900 }, { 4921, 0xb700 }, { 4927, 0x8098 },
02057 { 4931, 0x1018 }, { 4934, 0x2800 }, { 4936, 0x10c4 }, { 4940, 0x0211 },
02058 { 4943, 0x5920 }, { 4948, 0x0ba1 }, { 4954, 0x0027 }, { 4958, 0x605d },
02059 { 4965, 0x11b8 }, { 4971, 0xb3a4 }, { 4979, 0x8820 }, { 4982, 0xc051 },
02060 /* 0x9100 */
02061 { 4987, 0x2171 }, { 4993, 0x55d1 }, { 5001, 0xc2ad }, { 5009, 0x36d2 },
02062 { 5017, 0x8188 }, { 5021, 0x0e88 }, { 5026, 0x2092 }, { 5030, 0x0e10 },
02063 { 5034, 0x446a }, { 5040, 0x413a }, { 5046, 0x7142 }, { 5052, 0xb84f },
02064 { 5061, 0x002c }, { 5064, 0x4698 }, { 5070, 0xf630 }, { 5078, 0x2a83 },
02065 /* 0x9200 */
02066 { 5084, 0x16f3 }, { 5093, 0x314d }, { 5100, 0xc178 }, { 5107, 0x5769 },
02067 { 5116, 0xe4cd }, { 5125, 0x3302 }, { 5130, 0xc3a3 }, { 5138, 0xbbe1 },
02068 { 5148, 0x6700 }, { 5153, 0x8284 }, { 5157, 0x89b1 }, { 5164, 0xbd44 },
02069 { 5172, 0x79ef }, { 5184, 0xb3a9 }, { 5193, 0x51ab }, { 5201, 0x8a01 },
02070 /* 0x9300 */
02071 { 5205, 0x2105 }, { 5209, 0xf032 }, { 5216, 0x06b2 }, { 5222, 0x00d8 },
02072 { 5226, 0x0380 }, { 5229, 0x45a7 }, { 5237, 0xa6b0 }, { 5244, 0xa45b },
02073 { 5252, 0xad07 }, { 5260, 0x4924 }, { 5265, 0x0b5a }, { 5272, 0x0470 },
02074 { 5276, 0x3ef2 }, { 5286, 0xd208 }, { 5291, 0x00c4 }, { 5294, 0x2f80 },
02075 /* 0x9400 */
02076 { 5300, 0xe316 }, { 5308, 0x80e0 }, { 5312, 0xc000 }, { 5314, 0xa81e },
02077 { 5321, 0x1528 }, { 5326, 0x9220 }, { 5330, 0xe90a }, { 5337, 0x0006 },
02078 { 5339, 0x0018 }, { 5341, 0x0000 }, { 5341, 0x0000 }, { 5341, 0x0000 },
02079 { 5341, 0x0000 }, { 5341, 0x0000 }, { 5341, 0x0000 }, { 5341, 0x0000 },
02080 /* 0x9500 */
02081 { 5341, 0x0000 }, { 5341, 0x0000 }, { 5341, 0x0000 }, { 5341, 0x0000 },
02082 { 5341, 0x0000 }, { 5341, 0x0000 }, { 5341, 0x0000 }, { 5341, 0x4300 },
02083 { 5344, 0x7110 }, { 5349, 0xe000 }, { 5352, 0x1a42 }, { 5357, 0xa450 },
02084 { 5362, 0x0b40 }, { 5366, 0xe60f }, { 5375, 0x0051 }, { 5378, 0x0000 },
02085 /* 0x9600 */
02086 { 5378, 0x0000 }, { 5378, 0x6000 }, { 5380, 0x1074 }, { 5385, 0x378a },
02087 { 5393, 0x0002 }, { 5394, 0x01d4 }, { 5399, 0x4002 }, { 5401, 0xd810 },
02088 { 5406, 0x021e }, { 5411, 0xa442 }, { 5416, 0xc270 }, { 5422, 0x0408 },

```

```

02089 { 5424, 0x0400 }, { 5425, 0xe504 }, { 5431, 0x8200 }, { 5433, 0x0402 },
02090 /* 0x9700 */
02091 { 5435, 0x022c }, { 5439, 0x2c00 }, { 5442, 0x010e }, { 5446, 0x000a },
02092 { 5448, 0xc40a }, { 5453, 0x0da0 }, { 5458, 0x4488 }, { 5462, 0xa9c8 },
02093 { 5469, 0x0201 }, { 5471, 0xc6e0 }, { 5478, 0x5004 }, { 5481, 0xd766 },
02094 { 5491, 0x76b2 }, { 5500, 0x6b93 }, { 5509, 0x8013 }, { 5513, 0x0592 },
02095 /* 0x9800 */
02096 { 5518, 0x6480 }, { 5522, 0x5250 }, { 5527, 0xc869 }, { 5534, 0x402d },
02097 { 5539, 0x0490 }, { 5542, 0x06ce }, { 5549, 0x146c }, { 5555, 0x0000 },
02098 { 5555, 0x0000 }, { 5555, 0x0000 }, { 5555, 0x6800 }, { 5558, 0x8d91 },
02099 { 5565, 0x1124 }, { 5569, 0x0000 }, { 5569, 0x04ea }, { 5575, 0x0048 },
02100 /* 0x9900 */
02101 { 5577, 0x0184 }, { 5580, 0x9ce2 }, { 5588, 0x08c4 }, { 5592, 0x1e3e },
02102 { 5601, 0x61c3 }, { 5608, 0xdb10 }, { 5615, 0x0001 }, { 5616, 0x0000 },
02103 { 5616, 0x0000 }, { 5616, 0xa800 }, { 5619, 0x0040 }, { 5620, 0xa627 },
02104 { 5628, 0x0208 }, { 5630, 0x5618 }, { 5636, 0x1c80 }, { 5640, 0x6231 },
02105 /* 0x9a00 */
02106 { 5646, 0x181c }, { 5651, 0x4043 }, { 5655, 0x609d }, { 5662, 0x0168 },
02107 { 5666, 0x5c92 }, { 5673, 0x2052 }, { 5677, 0x0000 }, { 5677, 0x0000 },
02108 { 5677, 0x0000 }, { 5677, 0x0000 }, { 5677, 0xd400 }, { 5681, 0xca74 },
02109 { 5689, 0x414a }, { 5694, 0x18e5 }, { 5701, 0x12b1 }, { 5707, 0xa62c },
02110 /* 0x9b00 */
02111 { 5714, 0x7b3f }, { 5726, 0x1a45 }, { 5732, 0x2841 }, { 5736, 0x26b8 },
02112 { 5743, 0x1900 }, { 5746, 0x48e0 }, { 5751, 0x7d6a }, { 5761, 0x83a8 },
02113 { 5767, 0xae1f }, { 5777, 0x6411 }, { 5782, 0x12c0 }, { 5786, 0xd987 },
02114 { 5795, 0x4182 }, { 5799, 0xa181 }, { 5804, 0x8ca0 }, { 5809, 0xa788 },
02115 /* 0x9c00 */
02116 { 5816, 0x8805 }, { 5820, 0x5742 }, { 5827, 0x07cc }, { 5834, 0x20e2 },
02117 { 5839, 0xc63a }, { 5847, 0xf959 }, { 5857, 0x4f08 }, { 5863, 0x08a5 },
02118 { 5868, 0x0000 }, { 5868, 0x0000 }, { 5868, 0x0000 }, { 5868, 0x0000 },
02119 { 5868, 0x0000 }, { 5868, 0x0000 }, { 5868, 0x0040 }, { 5869, 0x0284 },
02120 /* 0x9d00 */
02121 { 5872, 0x0804 }, { 5874, 0x7182 }, { 5880, 0x8000 }, { 5881, 0x341d },
02122 { 5888, 0x04ac }, { 5893, 0x8018 }, { 5896, 0x0e2c }, { 5902, 0x58c1 },
02123 { 5908, 0x6458 }, { 5914, 0x01ec }, { 5920, 0x5402 }, { 5924, 0x9222 },
02124 { 5929, 0x0688 }, { 5933, 0xc4f0 }, { 5940, 0x4aa1 }, { 5946, 0x4019 },
02125 /* 0x9e00 */
02126 { 5950, 0x4484 }, { 5954, 0x3267 }, { 5962, 0x0000 }, { 5962, 0x0000 },
02127 { 5962, 0x0000 }, { 5962, 0x0000 }, { 5962, 0x1c00 }, { 5962, 0x1c00 },
02128 { 5965, 0xc0bd }, { 5973, 0x4940 }, { 5977, 0xd110 }, { 5982, 0x0039 },
02129 { 5986, 0x0940 }, { 5989, 0x8020 }, { 5991, 0x7090 }, { 5996, 0x8127 },
02130 /* 0x9f00 */
02131 { 6002, 0x820c }, { 6006, 0x8ed7 }, { 6016, 0x8c44 }, { 6021, 0xb696 },
02132 { 6030, 0x00fa }, { 6036, 0x65e8 }, { 6044, 0xe300 }, { 6049, 0x242b },
02133 { 6055, 0x8000 }, { 6056, 0x40d7 }, { 6063, 0x002e },
02134 };
02135
02136 static int
02137 jisx0212_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
02138 {
02139     (void)conv;
02140     if (n >= 2) {
02141         const Summary16 *summary = NULL;
02142         if (wc < 0x0460)
02143             summary = &jisx0212_uni2indx_page00[(wc>4)];
02144         else if (wc >= 0x2100 && wc < 0x2130)
02145             summary = &jisx0212_uni2indx_page21[(wc>4)-0x210];
02146         else if (wc >= 0x4e00 && wc < 0x9fb0)
02147             summary = &jisx0212_uni2indx_page4e[(wc>4)-0x4e0];
02148         if (summary) {
02149             unsigned short used = summary->used;
02150             unsigned int i = wc & 0x0f;
02151             if (used & ((unsigned short) 1 << i)) {
02152                 unsigned short c;
02153                 /* Keep in 'used' only the bits 0..i-1. */
02154                 used &= ((unsigned short) 1 << i) - 1;
02155                 /* Add 'summary->indx' and the number of bits set in 'used'. */
02156                 used = (used & 0x5555) + ((used & 0xaaaa) >> 1);
02157                 used = (used & 0x3333) + ((used & 0xcccc) >> 2);
02158                 used = (used & 0x0f0f) + ((used & 0xf0f0) >> 4);
02159                 used = (used & 0x00ff) + (used >> 8);
02160                 c = jisx0212_2charset[summary->indx + used];
02161                 r[0] = (c >> 8); r[1] = (c & 0xff);
02162                 return 2;
02163             }
02164         }
02165         return RET_ILSEQ;
02166     }
02167     return RET_TOOSMALL;
02168 }
02169 #endif /* NEED_TOMB */

```

12.288 koi8_c.h

```
00001 /* $XFree86: xc/lib/X11/lcUniConv/koi8_c.h,v 1.2 2000/11/28 16:10:29 dawes Exp $ */
```

```

00002
00003 /*
00004  * KOI8-C
00005  */
00006
00007 static const unsigned short koi8_c_2uni[128] = {
00008     /* 0x80 */
00009     0x0493, 0x0497, 0x049b, 0x049d, 0x04a3, 0x04af, 0x04b1, 0x04b3,
00010     0x04b7, 0x04b9, 0x04bb, 0x2580, 0x04d9, 0x04e3, 0x04e9, 0x04ef,
00011     /* 0x90 */
00012     0x0492, 0x0496, 0x049a, 0x049c, 0x04a2, 0x04ae, 0x04b0, 0x04b2,
00013     0x04b6, 0x04b8, 0x04ba, 0x2321, 0x04d8, 0x04e2, 0x04e8, 0x04ee,
00014     /* 0xa0 */
00015     0x00a0, 0x0452, 0x0453, 0x0451, 0x0454, 0x0455, 0x0456, 0x0457,
00016     0x0458, 0x0459, 0x045a, 0x045b, 0x045c, 0x0491, 0x045e, 0x045f,
00017     /* 0xb0 */
00018     0x2116, 0x0402, 0x0403, 0x0401, 0x0404, 0x0405, 0x0406, 0x0407,
00019     0x0486, 0x0409, 0x040a, 0x040b, 0x040c, 0x0490, 0x040e, 0x040f,
00020     /* 0xc0 */
00021     0x044e, 0x0430, 0x0431, 0x0446, 0x0434, 0x0435, 0x0444, 0x0433,
00022     0x0445, 0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e,
00023     /* 0xd0 */
00024     0x043f, 0x044f, 0x0440, 0x0441, 0x0442, 0x0443, 0x0436, 0x0432,
00025     0x044c, 0x044b, 0x0443, 0x0437, 0x0448, 0x044d, 0x0449, 0x0447, 0x044a,
00026     /* 0xe0 */
00027     0x042e, 0x0410, 0x0411, 0x0426, 0x0414, 0x0415, 0x0424, 0x0413,
00028     0x0425, 0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e,
00029     /* 0xf0 */
00030     0x041f, 0x042f, 0x0420, 0x0421, 0x0422, 0x0423, 0x0416, 0x0412,
00031     0x042c, 0x042b, 0x0417, 0x0428, 0x042d, 0x0429, 0x0427, 0x042a,
00032 };
00033
00034 static int
00035 koi8_c_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00036 {
00037     unsigned char c = *s;
00038     if (c < 0x80)
00039         *pwc = (ucs4_t) c;
00040     else
00041         *pwc = (ucs4_t) koi8_c_2uni[c-0x80];
00042     return 1;
00043 }
00044
00045 static const unsigned char koi8_c_page00[1] = {
00046     0xa0, /* 0xa0-0xa7 */
00047 };
00048 static const unsigned char koi8_c_page04[240] = {
00049     0x00, 0xb3, 0xb1, 0xb2, 0xb4, 0xb5, 0xb6, 0xb7, /* 0x00-0x07 */
00050     0xb8, 0xb9, 0xba, 0xbb, 0xbc, 0x00, 0xbe, 0xbf, /* 0x08-0x0f */
00051     0xe1, 0xe2, 0xf7, 0xe7, 0xe4, 0xe5, 0xf6, 0xfa, /* 0x10-0x17 */
00052     0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, 0xf0, /* 0x18-0x1f */
00053     0xf2, 0xf3, 0xf4, 0xf5, 0xe6, 0xe8, 0xe3, 0xfe, /* 0x20-0x27 */
00054     0xfb, 0xfd, 0xff, 0xf9, 0xf8, 0xfc, 0xe0, 0xf1, /* 0x28-0x2f */
00055     0xc1, 0xc2, 0xd7, 0xc7, 0xc4, 0xc5, 0xd6, 0xda, /* 0x30-0x37 */
00056     0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, 0xd0, /* 0x38-0x3f */
00057     0xd2, 0xd3, 0xd4, 0xd5, 0xc6, 0xc8, 0xc3, 0xde, /* 0x40-0x47 */
00058     0xdb, 0xdd, 0xdf, 0xd9, 0xd8, 0xdc, 0xc0, 0xd1, /* 0x48-0x4f */
00059     0x00, 0xa3, 0xa1, 0xa2, 0xa4, 0xa5, 0xa6, 0xa7, /* 0x50-0x57 */
00060     0xa8, 0xa9, 0xaa, 0xab, 0xac, 0x00, 0xae, 0xaf, /* 0x58-0x5f */
00061     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00062     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00063     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00064     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00065     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00066     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00067     0xbd, 0xad, 0x90, 0x80, 0x00, 0x00, 0x91, 0x81, /* 0x90-0x97 */
00068     0x00, 0x00, 0x92, 0x82, 0x93, 0x83, 0x00, 0x00, /* 0x98-0x9f */
00069     0x00, 0x00, 0x94, 0x84, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00070     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x95, 0x85, /* 0xa8-0xaf */
00071     0x96, 0x86, 0x97, 0x87, 0x00, 0x00, 0x98, 0x88, /* 0xb0-0xb7 */
00072     0x99, 0x89, 0x9a, 0x8a, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00073     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00074     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00075     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00076     0x9c, 0x8c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00077     0x00, 0x00, 0x9d, 0x8d, 0x00, 0x00, 0x00, 0x00, /* 0xe0-0xe7 */
00078     0x9e, 0x8e, 0x00, 0x00, 0x00, 0x00, 0x9f, 0x8f, /* 0xe8-0xef */
00079 };
00080 static const unsigned char koi8_c_page22[1] = {
00081     0xb0, /* 0x16-0x16 */
00082 };
00083
00084 static int
00085 koi8_c_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00086 {
00087     unsigned char c = 0;
00088     if (wc < 0x0080) {

```

```

00089     *r = wc;
00090     return 1;
00091 }
00092 else if (wc >= 0x00a0 && wc < 0x00a1)
00093     c = koi8_c_page00[wc-0x00a0];
00094 else if (wc >= 0x0400 && wc < 0x04ef)
00095     c = koi8_c_page04[wc-0x0400];
00096 else if (wc >= 0x2216 && wc < 0x2217)
00097     c = koi8_c_page22[wc-0x2216];
00098 if (c != 0) {
00099     *r = c;
00100     return 1;
00101 }
00102 return RET_ILSEQ;
00103 }

```

12.289 koi8_r.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/koi8_r.h,v 1.3 2000/11/29 17:40:34 dawes Exp $ */
00002
00003 /*
00004  * KOI8-R
00005  */
00006
00007 /* Specification: RFC 1489 */
00008
00009 #ifdef NEED_TOWC
00010 static const unsigned short koi8_r_2uni[128] = {
00011     /* 0x80 */
00012     0x2500, 0x2502, 0x250c, 0x2510, 0x2514, 0x2518, 0x251c, 0x2524,
00013     0x252c, 0x2534, 0x253c, 0x2580, 0x2584, 0x2588, 0x258c, 0x2590,
00014     /* 0x90 */
00015     0x2591, 0x2592, 0x2593, 0x2320, 0x25a0, 0x2219, 0x221a, 0x2248,
00016     0x2264, 0x2265, 0x00a0, 0x2321, 0x00b0, 0x00b2, 0x00b7, 0x00f7,
00017     /* 0xa0 */
00018     0x2550, 0x2551, 0x2552, 0x0451, 0x2553, 0x2554, 0x2555, 0x2556,
00019     0x2557, 0x2558, 0x2559, 0x255a, 0x255b, 0x255c, 0x255d, 0x255e,
00020     /* 0xb0 */
00021     0x255f, 0x2560, 0x2561, 0x0401, 0x2562, 0x2563, 0x2564, 0x2565,
00022     0x2566, 0x2567, 0x2568, 0x2569, 0x256a, 0x256b, 0x256c, 0x00a9,
00023     /* 0xc0 */
00024     0x044e, 0x0430, 0x0431, 0x0446, 0x0434, 0x0435, 0x0444, 0x0433,
00025     0x0445, 0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e,
00026     /* 0xd0 */
00027     0x043f, 0x044f, 0x0440, 0x0441, 0x0442, 0x0443, 0x0436, 0x0432,
00028     0x044c, 0x044b, 0x0437, 0x0448, 0x044d, 0x0449, 0x0447, 0x044a,
00029     /* 0xe0 */
00030     0x042e, 0x0410, 0x0411, 0x0426, 0x0414, 0x0415, 0x0424, 0x0413,
00031     0x0425, 0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e,
00032     /* 0xf0 */
00033     0x041f, 0x042f, 0x0420, 0x0421, 0x0422, 0x0423, 0x0416, 0x0412,
00034     0x042c, 0x042b, 0x0417, 0x0428, 0x042d, 0x0429, 0x0427, 0x042a,
00035 };
00036
00037 static int
00038 koi8_r_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00039 {
00040     unsigned char c = *s;
00041     if (c < 0x80)
00042         *pwc = (ucs4_t) c;
00043     else
00044         *pwc = (ucs4_t) koi8_r_2uni[c-0x80];
00045     return 1;
00046 }
00047 #endif /* NEED_TOWC */
00048
00049 #ifdef NEED_TOMB
00050 static const unsigned char koi8_r_page00[88] = {
00051     0x9a, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00052     0x00, 0xbf, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa8-0xaf */
00053     0x9c, 0x00, 0x9d, 0x00, 0x00, 0x00, 0x00, 0x9e, /* 0xb0-0xb7 */
00054     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00055     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00056     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00057     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00058     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00059     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe0-0xe7 */
00060     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe8-0xef */
00061     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x9f, /* 0xf0-0xf7 */
00062 };
00063 static const unsigned char koi8_r_page04[88] = {
00064     0x00, 0xb3, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x00-0x07 */
00065     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00066     0xe1, 0xe2, 0xf7, 0xe7, 0xe4, 0xe5, 0xf6, 0xfa, /* 0x10-0x17 */
00067     0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, 0xf0, /* 0x18-0x1f */

```

```

00068 0xf2, 0xf3, 0xf4, 0xf5, 0xe6, 0xe8, 0xe3, 0xfe, /* 0x20-0x27 */
00069 0xfb, 0xfd, 0xff, 0xf9, 0xf8, 0xfc, 0xe0, 0xf1, /* 0x28-0x2f */
00070 0xc1, 0xc2, 0xd7, 0xc7, 0xc4, 0xc5, 0xd6, 0xda, /* 0x30-0x37 */
00071 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, 0xd0, /* 0x38-0x3f */
00072 0xd2, 0xd3, 0xd4, 0xd5, 0xc6, 0xc8, 0xc3, 0xde, /* 0x40-0x47 */
00073 0xdb, 0xdd, 0xdf, 0xd9, 0xd8, 0xdc, 0xc0, 0xd1, /* 0x48-0x4f */
00074 0x00, 0xa3, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00075 };
00076 static const unsigned char koi8_r_page22[80] = {
00077 0x00, 0x95, 0x96, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00078 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00079 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00080 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00081 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00082 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00083 0x97, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00084 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00085 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00086 0x00, 0x00, 0x00, 0x00, 0x98, 0x99, 0x00, 0x00, /* 0x60-0x67 */
00087 };
00088 static const unsigned char koi8_r_page23[8] = {
00089 0x93, 0x9b, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00090 };
00091 static const unsigned char koi8_r_page25[168] = {
00092 0x80, 0x00, 0x81, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x00-0x07 */
00093 0x00, 0x00, 0x00, 0x00, 0x82, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00094 0x83, 0x00, 0x00, 0x00, 0x84, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00095 0x85, 0x00, 0x00, 0x00, 0x86, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00096 0x00, 0x00, 0x00, 0x00, 0x87, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00097 0x00, 0x00, 0x00, 0x00, 0x88, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00098 0x00, 0x00, 0x00, 0x00, 0x89, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00099 0x00, 0x00, 0x00, 0x00, 0x8a, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00100 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00101 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00102 0xa0, 0xa1, 0xa2, 0xa4, 0xa5, 0xa6, 0xa7, 0xa8, /* 0x50-0x57 */
00103 0xa9, 0xaa, 0xab, 0xac, 0xad, 0xae, 0xaf, 0xb0, /* 0x58-0x5f */
00104 0xb1, 0xb2, 0xb4, 0xb5, 0xb6, 0xb7, 0xb8, 0xb9, /* 0x60-0x67 */
00105 0xba, 0xbb, 0xbc, 0xbd, 0xbe, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00106 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00107 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00108 0x8b, 0x00, 0x00, 0x00, 0x8c, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00109 0x8d, 0x00, 0x00, 0x00, 0x8e, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00110 0x8f, 0x90, 0x91, 0x92, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00111 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x98-0x9f */
00112 0x94, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00113 };
00114
00115 static int
00116 koi8_r_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00117 {
00118     (void)conv; (void)n;
00119     unsigned char c = 0;
00120     if (wc < 0x0080) {
00121         *r = wc;
00122         return 1;
00123     }
00124     else if (wc >= 0x00a0 && wc < 0x00f8)
00125         c = koi8_r_page00[wc-0x00a0];
00126     else if (wc >= 0x0400 && wc < 0x0458)
00127         c = koi8_r_page04[wc-0x0400];
00128     else if (wc >= 0x2218 && wc < 0x2268)
00129         c = koi8_r_page22[wc-0x2218];
00130     else if (wc >= 0x2320 && wc < 0x2328)
00131         c = koi8_r_page23[wc-0x2320];
00132     else if (wc >= 0x2500 && wc < 0x25a8)
00133         c = koi8_r_page25[wc-0x2500];
00134     if (c != 0) {
00135         *r = c;
00136         return 1;
00137     }
00138     return RET_ILSEQ;
00139 }
00140 #endif /* NEED_TOMB */

```

12.290 koi8_u.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/koi8_u.h,v 1.3 2000/11/29 17:40:34 dawes Exp $ */
00002
00003 /*
00004  * KOI8-U
00005  */
00006
00007 /* Specification: RFC 2319 */
00008 #ifndef NEED_TOWC
00009 static const unsigned short koi8_u_2uni[128] = {

```



```
00010  /* 0x80 */
00011  0x2500, 0x2502, 0x250c, 0x2510, 0x2514, 0x2518, 0x251c, 0x2524,
00012  0x252c, 0x2534, 0x253c, 0x2580, 0x2584, 0x2588, 0x258c, 0x2590,
00013  /* 0x90 */
00014  0x2591, 0x2592, 0x2593, 0x2320, 0x25a0, 0x2219, 0x221a, 0x2248,
00015  0x2264, 0x2265, 0x00a0, 0x2321, 0x00b0, 0x00b2, 0x00b7, 0x00f7,
00016  /* 0xa0 */
00017  0x2550, 0x2551, 0x2552, 0x0451, 0x0454, 0x2554, 0x0456, 0x0457,
00018  0x2557, 0x2558, 0x2559, 0x255a, 0x255b, 0x0491, 0x255d, 0x255e,
00019  /* 0xb0 */
00020  0x255f, 0x2560, 0x2561, 0x0401, 0x0404, 0x2563, 0x0406, 0x0407,
00021  0x2566, 0x2567, 0x2568, 0x2569, 0x256a, 0x0490, 0x256c, 0x00a9,
00022  /* 0xc0 */
00023  0x044e, 0x0430, 0x0431, 0x0446, 0x0434, 0x0435, 0x0444, 0x0433,
00024  0x0445, 0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e,
00025  /* 0xd0 */
00026  0x043f, 0x044f, 0x0440, 0x0441, 0x0442, 0x0443, 0x0436, 0x0432,
00027  0x044c, 0x044b, 0x0437, 0x0448, 0x044d, 0x0449, 0x0447, 0x044a,
00028  /* 0xe0 */
00029  0x042e, 0x0410, 0x0411, 0x0426, 0x0414, 0x0415, 0x0424, 0x0413,
00030  0x0425, 0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e,
00031  /* 0xf0 */
00032  0x041f, 0x042f, 0x0420, 0x0421, 0x0422, 0x0423, 0x0416, 0x0412,
00033  0x042c, 0x042b, 0x0417, 0x0428, 0x042d, 0x0429, 0x0427, 0x042a,
00034  };
00035
00036 static int
00037 koi8_u_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00038 {
00039     unsigned char c = *s;
00040     if (c < 0x80)
00041         *pwc = (ucs4_t) c;
00042     else
00043         *pwc = (ucs4_t) koi8_u_2uni[c-0x80];
00044     return 1;
00045 }
00046 #endif /* NEED_TOWC */
00047
00048 #ifdef NEED_TOMB
00049 static const unsigned char koi8_u_page00[88] = {
00050     0x9a, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00051     0x00, 0xbf, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa8-0xaf */
00052     0x9c, 0x00, 0x9d, 0x00, 0x00, 0x00, 0x00, 0x9e, /* 0xb0-0xb7 */
00053     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00054     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00055     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00056     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00057     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00058     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe0-0xe7 */
00059     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe8-0xef */
00060     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x9f, /* 0xf0-0xf7 */
00061 };
00062 static const unsigned char koi8_u_page04[152] = {
00063     0x00, 0xb3, 0x00, 0x00, 0xb4, 0x00, 0xb6, 0xb7, /* 0x00-0x07 */
00064     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00065     0xe1, 0xe2, 0xf7, 0xe7, 0xe4, 0xe5, 0xf6, 0xfa, /* 0x10-0x17 */
00066     0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, 0xf0, /* 0x18-0x1f */
00067     0xf2, 0xf3, 0xf4, 0xf5, 0xe6, 0xe8, 0xe3, 0xfe, /* 0x20-0x27 */
00068     0xfb, 0xfd, 0xff, 0xf9, 0xf8, 0xfc, 0xe0, 0xf1, /* 0x28-0x2f */
00069     0xc1, 0xc2, 0xd7, 0xc7, 0xc4, 0xc5, 0xd6, 0xda, /* 0x30-0x37 */
00070     0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, 0xd0, /* 0x38-0x3f */
00071     0xd2, 0xd3, 0xd4, 0xd5, 0xc6, 0xc8, 0xc3, 0xde, /* 0x40-0x47 */
00072     0xdb, 0xdd, 0xdf, 0xd9, 0xd8, 0xdc, 0xc0, 0xd1, /* 0x48-0x4f */
00073     0x00, 0xa3, 0x00, 0x00, 0xa4, 0x00, 0xa6, 0xa7, /* 0x50-0x57 */
00074     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00075     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00076     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00077     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00078     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00079     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00080     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00081     0xbd, 0xad, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00082 };
00083 static const unsigned char koi8_u_page22[80] = {
00084     0x00, 0x95, 0x96, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00085     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00086     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00087     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00088     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00089     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00090     0x97, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00091     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00092     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00093     0x00, 0x00, 0x00, 0x00, 0x98, 0x99, 0x00, 0x00, /* 0x60-0x67 */
00094 };
00095 static const unsigned char koi8_u_page23[8] = {
00096     0x93, 0x9b, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */

```

```

00097 };
00098 static const unsigned char koi8_u_page25[168] = {
00099     0x80, 0x00, 0x81, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x00-0x07 */
00100     0x00, 0x00, 0x00, 0x00, 0x82, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00101     0x83, 0x00, 0x00, 0x00, 0x84, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00102     0x85, 0x00, 0x00, 0x00, 0x86, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00103     0x00, 0x00, 0x00, 0x00, 0x87, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00104     0x00, 0x00, 0x00, 0x00, 0x88, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00105     0x00, 0x00, 0x00, 0x00, 0x89, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00106     0x00, 0x00, 0x00, 0x00, 0x8a, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00107     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00108     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00109     0xa0, 0xa1, 0xa2, 0x00, 0xa5, 0x00, 0x00, 0xa8, /* 0x50-0x57 */
00110     0xa9, 0xaa, 0xab, 0xac, 0x00, 0xae, 0xaf, 0xb0, /* 0x58-0x5f */
00111     0xb1, 0xb2, 0x00, 0xb5, 0x00, 0x00, 0xb8, 0xb9, /* 0x60-0x67 */
00112     0xba, 0xbb, 0xbc, 0x00, 0xbe, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00113     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00114     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00115     0x8b, 0x00, 0x00, 0x00, 0x8c, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00116     0x8d, 0x00, 0x00, 0x00, 0x8e, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00117     0x8f, 0x90, 0x91, 0x92, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00118     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x98-0x9f */
00119     0x94, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00120 };
00121
00122 static int
00123 koi8_u_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00124 {
00125     (void)conv; (void)n;
00126     unsigned char c = 0;
00127     if (wc < 0x0080) {
00128         *r = wc;
00129         return 1;
00130     }
00131     else if (wc >= 0x00a0 && wc < 0x00f8)
00132         c = koi8_u_page00[wc-0x00a0];
00133     else if (wc >= 0x0400 && wc < 0x0498)
00134         c = koi8_u_page04[wc-0x0400];
00135     else if (wc >= 0x2218 && wc < 0x2268)
00136         c = koi8_u_page22[wc-0x2218];
00137     else if (wc >= 0x2320 && wc < 0x2328)
00138         c = koi8_u_page23[wc-0x2320];
00139     else if (wc >= 0x2500 && wc < 0x25a8)
00140         c = koi8_u_page25[wc-0x2500];
00141     if (c != 0) {
00142         *r = c;
00143         return 1;
00144     }
00145     return RET_ILSEQ;
00146 }
00147 #endif /* NEED_TOMB */

```

12.291 ksc5601.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/ksc5601.h,v 1.5 2003/05/27 22:26:34 tsi Exp $ */
00002
00003 /*
00004  * KSC5601.1987-0
00005  */
00006 #ifdef NEED_TOWC
00007 static const unsigned short ksc5601_2uni_page21[1115] = {
00008     /* 0x21 */
00009     0x3000, 0x3001, 0x3002, 0x00b7, 0x2025, 0x2026, 0x00a8, 0x3003,
00010     0x00ad, 0x2015, 0x2225, 0xff3c, 0x223c, 0x2018, 0x2019, 0x201c,
00011     0x201d, 0x3014, 0x3015, 0x3008, 0x3009, 0x300a, 0x300b, 0x300c,
00012     0x300d, 0x300e, 0x300f, 0x3010, 0x3011, 0x00b1, 0x00d7, 0x00f7,
00013     0x2260, 0x2264, 0x2265, 0x221e, 0x2234, 0x00b0, 0x2032, 0x2033,
00014     0x2103, 0x212b, 0xffe0, 0xffe1, 0xffe5, 0x2642, 0x2640, 0x2220,
00015     0x22a5, 0x2312, 0x2202, 0x2207, 0x2261, 0x2252, 0x00a7, 0x203b,
00016     0x2606, 0x2605, 0x25cb, 0x25cf, 0x25ce, 0x25c7, 0x25c6, 0x25a1,
00017     0x25a0, 0x25b3, 0x25b2, 0x25bd, 0x25bc, 0x2192, 0x2190, 0x2191,
00018     0x2193, 0x2194, 0x3013, 0x226a, 0x226b, 0x221a, 0x223d, 0x221d,
00019     0x2235, 0x222b, 0x222c, 0x2208, 0x220b, 0x2286, 0x2287, 0x2282,
00020     0x2283, 0x222a, 0x2229, 0x2227, 0x2228, 0xffe2,
00021     /* 0x22 */
00022     0x21d2, 0x21d4, 0x2200, 0x2203, 0x00b4, 0xff5e, 0x02c7, 0x02d8,
00023     0x02dd, 0x02da, 0x02d9, 0x00b8, 0x02db, 0x00a1, 0x00bf, 0x02d0,
00024     0x222e, 0x2211, 0x220f, 0x00a4, 0x2109, 0x2030, 0x25c1, 0x25c0,
00025     0x25b7, 0x25b6, 0x2664, 0x2660, 0x2661, 0x2665, 0x2667, 0x2663,
00026     0x2299, 0x25c8, 0x25a3, 0x25d0, 0x25d1, 0x2592, 0x25a4, 0x25a5,
00027     0x25a8, 0x25a7, 0x25a6, 0x25a9, 0x2668, 0x26f0, 0x260e, 0x261c,
00028     0x261e, 0x00b6, 0x2020, 0x2021, 0x2195, 0x2197, 0x2199, 0x2196,
00029     0x2198, 0x266d, 0x2669, 0x266a, 0x266c, 0x327f, 0x321c, 0x2116,
00030     0x33c7, 0x2122, 0x33c2, 0x33d8, 0x2121, 0xffff, 0xffff, 0xffff,
00031     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,

```

```
00032 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00033 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00034 /* 0x23 */
00035 0xff01, 0xff02, 0xff03, 0xff04, 0xff05, 0xff06, 0xff07, 0xff08,
00036 0xff09, 0xff0a, 0xff0b, 0xff0c, 0xff0d, 0xff0e, 0xff0f, 0xff10,
00037 0xff11, 0xff12, 0xff13, 0xff14, 0xff15, 0xff16, 0xff17, 0xff18,
00038 0xff19, 0xff1a, 0xff1b, 0xff1c, 0xff1d, 0xff1e, 0xff1f, 0xff20,
00039 0xff21, 0xff22, 0xff23, 0xff24, 0xff25, 0xff26, 0xff27, 0xff28,
00040 0xff29, 0xff2a, 0xff2b, 0xff2c, 0xff2d, 0xff2e, 0xff2f, 0xff30,
00041 0xff31, 0xff32, 0xff33, 0xff34, 0xff35, 0xff36, 0xff37, 0xff38,
00042 0xff39, 0xff3a, 0xff3b, 0xffe6, 0xff3d, 0xff3e, 0xff3f, 0xff40,
00043 0xff41, 0xff42, 0xff43, 0xff44, 0xff45, 0xff46, 0xff47, 0xff48,
00044 0xff49, 0xff4a, 0xff4b, 0xff4c, 0xff4d, 0xff4e, 0xff4f, 0xff50,
00045 0xff51, 0xff52, 0xff53, 0xff54, 0xff55, 0xff56, 0xff57, 0xff58,
00046 0xff59, 0xff5a, 0xff5b, 0xff5c, 0xff5d, 0xffe3,
00047 /* 0x24 */
00048 0x3131, 0x3132, 0x3133, 0x3134, 0x3135, 0x3136, 0x3137, 0x3138,
00049 0x3139, 0x313a, 0x313b, 0x313c, 0x313d, 0x313e, 0x313f, 0x3140,
00050 0x3141, 0x3142, 0x3143, 0x3144, 0x3145, 0x3146, 0x3147, 0x3148,
00051 0x3149, 0x314a, 0x314b, 0x314c, 0x314d, 0x314e, 0x314f, 0x3150,
00052 0x3151, 0x3152, 0x3153, 0x3154, 0x3155, 0x3156, 0x3157, 0x3158,
00053 0x3159, 0x315a, 0x315b, 0x315c, 0x315d, 0x315e, 0x315f, 0x3160,
00054 0x3161, 0x3162, 0x3163, 0x3164, 0x3165, 0x3166, 0x3167, 0x3168,
00055 0x3169, 0x316a, 0x316b, 0x316c, 0x316d, 0x316e, 0x316f, 0x3170,
00056 0x3171, 0x3172, 0x3173, 0x3174, 0x3175, 0x3176, 0x3177, 0x3178,
00057 0x3179, 0x317a, 0x317b, 0x317c, 0x317d, 0x317e, 0x317f, 0x3180,
00058 0x3181, 0x3182, 0x3183, 0x3184, 0x3185, 0x3186, 0x3187, 0x3188,
00059 0x3189, 0x318a, 0x318b, 0x318c, 0x318d, 0x318e,
00060 /* 0x25 */
00061 0x2170, 0x2171, 0x2172, 0x2173, 0x2174, 0x2175, 0x2176, 0x2177,
00062 0x2178, 0x2179, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0x2160,
00063 0x2161, 0x2162, 0x2163, 0x2164, 0x2165, 0x2166, 0x2167, 0x2168,
00064 0x2169, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00065 0x0391, 0x0392, 0x0393, 0x0394, 0x0395, 0x0396, 0x0397, 0x0398,
00066 0x0399, 0x039a, 0x039b, 0x039c, 0x039d, 0x039e, 0x039f, 0x03a0,
00067 0x03a1, 0x03a2, 0x03a3, 0x03a4, 0x03a5, 0x03a6, 0x03a7, 0x03a8, 0x03a9,
00068 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00069 0x03b1, 0x03b2, 0x03b3, 0x03b4, 0x03b5, 0x03b6, 0x03b7, 0x03b8,
00070 0x03b9, 0x03ba, 0x03bb, 0x03bc, 0x03bd, 0x03be, 0x03bf, 0x03c0,
00071 0x03c1, 0x03c2, 0x03c3, 0x03c4, 0x03c5, 0x03c6, 0x03c7, 0x03c8, 0x03c9,
00072 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00073 /* 0x26 */
00074 0x2500, 0x2502, 0x250c, 0x2510, 0x2518, 0x2514, 0x251c, 0x252c,
00075 0x2524, 0x2534, 0x253c, 0x2501, 0x2503, 0x250f, 0x2513, 0x251b,
00076 0x2517, 0x2523, 0x2533, 0x252b, 0x253b, 0x254b, 0x2520, 0x252f,
00077 0x2528, 0x2537, 0x253f, 0x251d, 0x2530, 0x2525, 0x2538, 0x2542,
00078 0x2512, 0x2511, 0x251a, 0x2519, 0x2516, 0x2515, 0x250e, 0x250d,
00079 0x251e, 0x251f, 0x2521, 0x2522, 0x2526, 0x2527, 0x2529, 0x252a,
00080 0x252d, 0x252e, 0x2531, 0x2532, 0x2535, 0x2536, 0x2539, 0x253a,
00081 0x253d, 0x253e, 0x2540, 0x2541, 0x2543, 0x2544, 0x2545, 0x2546,
00082 0x2547, 0x2548, 0x2549, 0x254a, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00083 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00084 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00085 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00086 /* 0x27 */
00087 0x3395, 0x3396, 0x3397, 0x2113, 0x3398, 0x33c4, 0x33a3, 0x33a4,
00088 0x33a5, 0x33a6, 0x3399, 0x339a, 0x339b, 0x339c, 0x339d, 0x339e,
00089 0x339f, 0x33a0, 0x33a1, 0x33a2, 0x33ca, 0x338d, 0x338e, 0x338f,
00090 0x33cf, 0x3388, 0x3389, 0x33c8, 0x33a7, 0x33a8, 0x33b0, 0x33b1,
00091 0x33b2, 0x33b3, 0x33b4, 0x33b5, 0x33b6, 0x33b7, 0x33b8, 0x33b9,
00092 0x3380, 0x3381, 0x3382, 0x3383, 0x3384, 0x33ba, 0x33bb, 0x33bc,
00093 0x33bd, 0x33be, 0x33bf, 0x3390, 0x3391, 0x3392, 0x3393, 0x3394,
00094 0x2126, 0x33c0, 0x33c1, 0x338a, 0x338b, 0x338c, 0x33d6, 0x33c5,
00095 0x33ad, 0x33ae, 0x33af, 0x33db, 0x33a9, 0x33aa, 0x33ab, 0x33ac,
00096 0x33dd, 0x33d0, 0x33d3, 0x33c9, 0x33dc, 0x33c6, 0xffffd,
00097 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00098 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd, 0xffffd,
00099 /* 0x28 */
00100 0x00c6, 0x00d0, 0x00aa, 0x0126, 0xffffd, 0x0132, 0xffffd, 0x013f,
00101 0x0141, 0x00d8, 0x0152, 0x00ba, 0x00de, 0x0166, 0x014a, 0xffffd,
00102 0x3260, 0x3261, 0x3262, 0x3263, 0x3264, 0x3265, 0x3266, 0x3267,
00103 0x3268, 0x3269, 0x326a, 0x326b, 0x326c, 0x326d, 0x326e, 0x326f,
00104 0x3270, 0x3271, 0x3272, 0x3273, 0x3274, 0x3275, 0x3276, 0x3277,
00105 0x3278, 0x3279, 0x327a, 0x327b, 0x24d0, 0x24d1, 0x24d2, 0x24d3,
00106 0x24d4, 0x24d5, 0x24d6, 0x24d7, 0x24d8, 0x24d9, 0x24da, 0x24db,
00107 0x24dc, 0x24dd, 0x24de, 0x24df, 0x24e0, 0x24e1, 0x24e2, 0x24e3,
00108 0x24e4, 0x24e5, 0x24e6, 0x24e7, 0x24e8, 0x24e9, 0x2460, 0x2461,
00109 0x2462, 0x2463, 0x2464, 0x2465, 0x2466, 0x2467, 0x2468, 0x2469,
00110 0x246a, 0x246b, 0x246c, 0x246d, 0x246e, 0x00bd, 0x2153, 0x2154,
00111 0x00bc, 0x00be, 0x215b, 0x215c, 0x215d, 0x215e,
00112 /* 0x29 */
00113 0x00e6, 0x0111, 0x00f0, 0x0127, 0x0131, 0x0133, 0x0138, 0x0140,
00114 0x0142, 0x00f8, 0x0153, 0x00df, 0x00fe, 0x0167, 0x014b, 0x0149,
00115 0x3200, 0x3201, 0x3202, 0x3203, 0x3204, 0x3205, 0x3206, 0x3207,
00116 0x3208, 0x3209, 0x320a, 0x320b, 0x320c, 0x320d, 0x320e, 0x320f,
00117 0x3210, 0x3211, 0x3212, 0x3213, 0x3214, 0x3215, 0x3216, 0x3217,
00118 0x3218, 0x3219, 0x321a, 0x321b, 0x249c, 0x249d, 0x249e, 0x249f,
```

```

00119 0x24a0, 0x24a1, 0x24a2, 0x24a3, 0x24a4, 0x24a5, 0x24a6, 0x24a7,
00120 0x24a8, 0x24a9, 0x24aa, 0x24ab, 0x24ac, 0x24ad, 0x24ae, 0x24af,
00121 0x24b0, 0x24b1, 0x24b2, 0x24b3, 0x24b4, 0x24b5, 0x2474, 0x2475,
00122 0x2476, 0x2477, 0x2478, 0x2479, 0x247a, 0x247b, 0x247c, 0x247d,
00123 0x247e, 0x247f, 0x2480, 0x2481, 0x2482, 0x00b9, 0x00b2, 0x00b3,
00124 0x2074, 0x207f, 0x2081, 0x2082, 0x2083, 0x2084,
00125 /* 0x2a */
00126 0x3041, 0x3042, 0x3043, 0x3044, 0x3045, 0x3046, 0x3047, 0x3048,
00127 0x3049, 0x304a, 0x304b, 0x304c, 0x304d, 0x304e, 0x304f, 0x3050,
00128 0x3051, 0x3052, 0x3053, 0x3054, 0x3055, 0x3056, 0x3057, 0x3058,
00129 0x3059, 0x305a, 0x305b, 0x305c, 0x305d, 0x305e, 0x305f, 0x3060,
00130 0x3061, 0x3062, 0x3063, 0x3064, 0x3065, 0x3066, 0x3067, 0x3068,
00131 0x3069, 0x306a, 0x306b, 0x306c, 0x306d, 0x306e, 0x306f, 0x3070,
00132 0x3071, 0x3072, 0x3073, 0x3074, 0x3075, 0x3076, 0x3077, 0x3078,
00133 0x3079, 0x307a, 0x307b, 0x307c, 0x307d, 0x307e, 0x307f, 0x3080,
00134 0x3081, 0x3082, 0x3083, 0x3084, 0x3085, 0x3086, 0x3087, 0x3088,
00135 0x3089, 0x308a, 0x308b, 0x308c, 0x308d, 0x308e, 0x308f, 0x3090,
00136 0x3091, 0x3092, 0x3093, 0xffff, 0xffff, 0xffff, 0xffff,
00137 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00138 /* 0x2b */
00139 0x30a1, 0x30a2, 0x30a3, 0x30a4, 0x30a5, 0x30a6, 0x30a7, 0x30a8,
00140 0x30a9, 0x30aa, 0x30ab, 0x30ac, 0x30ad, 0x30ae, 0x30af, 0x30b0,
00141 0x30b1, 0x30b2, 0x30b3, 0x30b4, 0x30b5, 0x30b6, 0x30b7, 0x30b8,
00142 0x30b9, 0x30ba, 0x30bb, 0x30bc, 0x30bd, 0x30be, 0x30bf, 0x30c0,
00143 0x30c1, 0x30c2, 0x30c3, 0x30c4, 0x30c5, 0x30c6, 0x30c7, 0x30c8,
00144 0x30c9, 0x30ca, 0x30cb, 0x30cc, 0x30cd, 0x30ce, 0x30cf, 0x30d0,
00145 0x30d1, 0x30d2, 0x30d3, 0x30d4, 0x30d5, 0x30d6, 0x30d7, 0x30d8,
00146 0x30d9, 0x30da, 0x30db, 0x30dc, 0x30dd, 0x30de, 0x30df, 0x30e0,
00147 0x30e1, 0x30e2, 0x30e3, 0x30e4, 0x30e5, 0x30e6, 0x30e7, 0x30e8,
00148 0x30e9, 0x30ea, 0x30eb, 0x30ec, 0x30ed, 0x30ee, 0x30ef, 0x30f0,
00149 0x30f1, 0x30f2, 0x30f3, 0x30f4, 0x30f5, 0x30f6, 0xffff, 0xffff,
00150 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00151 /* 0x2c */
00152 0x0410, 0x0411, 0x0412, 0x0413, 0x0414, 0x0415, 0x0401, 0x0416,
00153 0x0417, 0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e,
00154 0x041f, 0x0420, 0x0421, 0x0422, 0x0423, 0x0424, 0x0425, 0x0426,
00155 0x0427, 0x0428, 0x0429, 0x042a, 0x042b, 0x042c, 0x042d, 0x042e,
00156 0x042f, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00157 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00158 0x0430, 0x0431, 0x0432, 0x0433, 0x0434, 0x0435, 0x0451, 0x0436,
00159 0x0437, 0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e,
00160 0x043f, 0x0440, 0x0441, 0x0442, 0x0443, 0x0444, 0x0445, 0x0446,
00161 0x0447, 0x0448, 0x0449, 0x044a, 0x044b, 0x044c, 0x044d, 0x044e,
00162 0x044f,
00163 };
00164 static const unsigned short ksc5601_2uni_page30[2350] = {
00165 /* 0x30 */
00166 0xac00, 0xac01, 0xac04, 0xac07, 0xac08, 0xac09, 0xac0a, 0xac10,
00167 0xac11, 0xac12, 0xac13, 0xac14, 0xac15, 0xac16, 0xac17, 0xac19,
00168 0xac1a, 0xac1b, 0xac1c, 0xac1d, 0xac20, 0xac24, 0xac2c, 0xac2d,
00169 0xac2f, 0xac30, 0xac31, 0xac38, 0xac39, 0xac3c, 0xac40, 0xac4b,
00170 0xac4d, 0xac54, 0xac58, 0xac5c, 0xac70, 0xac71, 0xac74, 0xac77,
00171 0xac78, 0xac7a, 0xac80, 0xac81, 0xac83, 0xac84, 0xac85, 0xac86,
00172 0xac89, 0xac8b, 0xac8c, 0xac90, 0xac94, 0xac9c, 0xac9d,
00173 0xac9f, 0xaca0, 0xaca1, 0xaca8, 0xaca9, 0xacaa, 0xacac, 0xacaf,
00174 0xacb0, 0xacb8, 0xacb9, 0xacbb, 0xacbc, 0xacbd, 0xacc1, 0xacc4,
00175 0xacc8, 0xacc9, 0xaccd, 0xacd5, 0xace0, 0xace1, 0xace4, 0xace7,
00176 0xace8, 0xace9, 0xacec, 0xacef, 0xacf0, 0xacf1, 0xacf3, 0xacf5,
00177 0xacf6, 0xacfc, 0xacfd, 0xad00, 0xad04, 0xad06,
00178 /* 0x31 */
00179 0xad0c, 0xad0d, 0xad0f, 0xad11, 0xad18, 0xad1c, 0xad20, 0xad29,
00180 0xad2c, 0xad2d, 0xad34, 0xad35, 0xad38, 0xad3c, 0xad44, 0xad45,
00181 0xad47, 0xad49, 0xad50, 0xad54, 0xad58, 0xad61, 0xad63, 0xad6c,
00182 0xad6d, 0xad70, 0xad73, 0xad74, 0xad75, 0xad76, 0xad7b, 0xad7c,
00183 0xad7d, 0xad7f, 0xad81, 0xad82, 0xad88, 0xad89, 0xad8c, 0xad90,
00184 0xad9c, 0xad9d, 0xada4, 0xadb7, 0xadc0, 0xadc1, 0xadc4, 0xadc8,
00185 0xadd0, 0xadd1, 0xadd3, 0xaddc, 0xade0, 0xade4, 0xadf8, 0xadf9,
00186 0xadfc, 0xadff, 0xae00, 0xae01, 0xae08, 0xae09, 0xae0b, 0xae0d,
00187 0xae14, 0xae30, 0xae31, 0xae34, 0xae37, 0xae38, 0xae3a, 0xae40,
00188 0xae41, 0xae43, 0xae45, 0xae46, 0xae4a, 0xae4c, 0xae4d, 0xae4e,
00189 0xae50, 0xae54, 0xae56, 0xae5c, 0xae5d, 0xae5f, 0xae60, 0xae61,
00190 0xae65, 0xae68, 0xae69, 0xae6c, 0xae70, 0xae78,
00191 /* 0x32 */
00192 0xae79, 0xae7b, 0xae7c, 0xae7d, 0xae84, 0xae85, 0xae8c, 0xaebc,
00193 0xaebd, 0xaebe, 0xaec0, 0xaec4, 0xaec9, 0xaecd, 0xaecf, 0xaed0,
00194 0xaed1, 0xaed8, 0xaed9, 0xaedc, 0xae8e, 0xae8f, 0xaedf, 0xaf04,
00195 0xaf08, 0xaf09, 0xaf0d, 0xaf10, 0xaf12, 0xaf1c, 0xaf1d, 0xaf1e,
00196 0xaf30, 0xaf32, 0xaf34, 0xaf3c, 0xaf3d, 0xaf3f, 0xaf41, 0xaf42,
00197 0xaf43, 0xaf48, 0xaf49, 0xaf50, 0xaf5c, 0xaf5d, 0xaf64, 0xaf65,
00198 0xaf79, 0xaf80, 0xaf84, 0xaf88, 0xaf90, 0xaf91, 0xaf95, 0xaf9c,
00199 0xafb8, 0xafb9, 0xafbc, 0xafc0, 0xafc7, 0xafc8, 0xafc9, 0xafcb,
00200 0xafcd, 0xafce, 0xafd4, 0xafdc, 0xafef, 0xafef, 0xafff, 0xafff,
00201 0xb001, 0xb004, 0xb00c, 0xb00d, 0xb010, 0xb014, 0xb01d,
00202 0xb01e, 0xb028, 0xb044, 0xb045, 0xb048, 0xb04a, 0xb04c,
00203 0xb04e, 0xb053, 0xb054, 0xb055, 0xb057, 0xb059,
00204 /* 0x33 */
00205 0xb05d, 0xb07c, 0xb07d, 0xb080, 0xb084, 0xb08c, 0xb08d, 0xb08f,

```

```
00206 0xb091, 0xb098, 0xb099, 0xb09a, 0xb09c, 0xb09f, 0xb0a0, 0xb0a1,
00207 0xb0a2, 0xb0a8, 0xb0a9, 0xb0ab, 0xb0ac, 0xb0ad, 0xb0ae, 0xb0af,
00208 0xb0b1, 0xb0b3, 0xb0b4, 0xb0b5, 0xb0b8, 0xb0bc, 0xb0c4, 0xb0c5,
00209 0xb0c7, 0xb0c8, 0xb0c9, 0xb0d0, 0xb0d1, 0xb0d4, 0xb0d8, 0xb0e0,
00210 0xb0e5, 0xb108, 0xb109, 0xb10b, 0xb10c, 0xb110, 0xb112, 0xb113,
00211 0xb118, 0xb119, 0xb11b, 0xb11c, 0xb11d, 0xb123, 0xb124, 0xb125,
00212 0xb128, 0xb12c, 0xb134, 0xb135, 0xb137, 0xb138, 0xb139, 0xb140,
00213 0xb141, 0xb144, 0xb148, 0xb150, 0xb151, 0xb154, 0xb155, 0xb158,
00214 0xb15c, 0xb160, 0xb178, 0xb179, 0xb17c, 0xb180, 0xb182, 0xb188,
00215 0xb189, 0xb18b, 0xb18d, 0xb192, 0xb193, 0xb194, 0xb198, 0xb19c,
00216 0xb1a8, 0xb1cc, 0xb1d0, 0xb1d4, 0xb1dc, 0xb1dd,
00217 /* 0x34 */
00218 0xb1df, 0xb1e8, 0xb1e9, 0xb1ec, 0xb1f0, 0xb1f9, 0xb1fb, 0xb1fd,
00219 0xb204, 0xb205, 0xb208, 0xb20b, 0xb20c, 0xb214, 0xb215, 0xb217,
00220 0xb219, 0xb220, 0xb234, 0xb23c, 0xb258, 0xb25c, 0xb260, 0xb268,
00221 0xb269, 0xb274, 0xb275, 0xb27c, 0xb284, 0xb285, 0xb289, 0xb290,
00222 0xb291, 0xb294, 0xb298, 0xb299, 0xb29a, 0xb2a0, 0xb2a1, 0xb2a3,
00223 0xb2a5, 0xb2a6, 0xb2aa, 0xb2ac, 0xb2b0, 0xb2b4, 0xb2c8, 0xb2c9,
00224 0xb2cc, 0xb2d0, 0xb2d2, 0xb2d8, 0xb2d9, 0xb2db, 0xb2dd, 0xb2e2,
00225 0xb2e4, 0xb2e5, 0xb2e6, 0xb2e8, 0xb2eb, 0xb2ec, 0xb2ed, 0xb2ee,
00226 0xb2ef, 0xb2f3, 0xb2f4, 0xb2f5, 0xb2f7, 0xb2f8, 0xb2f9, 0xb2fa,
00227 0xb2fb, 0xb2ff, 0xb300, 0xb301, 0xb304, 0xb308, 0xb310, 0xb311,
00228 0xb313, 0xb314, 0xb315, 0xb31c, 0xb354, 0xb355, 0xb356, 0xb358,
00229 0xb35b, 0xb35c, 0xb35e, 0xb35f, 0xb364, 0xb365,
00230 /* 0x35 */
00231 0xb367, 0xb369, 0xb36b, 0xb36e, 0xb370, 0xb371, 0xb374, 0xb378,
00232 0xb380, 0xb381, 0xb383, 0xb384, 0xb385, 0xb38c, 0xb390, 0xb394,
00233 0xb3a0, 0xb3a1, 0xb3a8, 0xb3ac, 0xb3c4, 0xb3c5, 0xb3c8, 0xb3cb,
00234 0xb3cc, 0xb3ce, 0xb3d0, 0xb3d4, 0xb3d5, 0xb3d7, 0xb3d9, 0xb3db,
00235 0xb3dd, 0xb3e0, 0xb3e4, 0xb3e8, 0xb3fc, 0xb410, 0xb418, 0xb41c,
00236 0xb420, 0xb428, 0xb429, 0xb42b, 0xb434, 0xb450, 0xb451, 0xb454,
00237 0xb458, 0xb460, 0xb461, 0xb463, 0xb465, 0xb46c, 0xb480, 0xb488,
00238 0xb49d, 0xb4a4, 0xb4a8, 0xb4ac, 0xb4b5, 0xb4b7, 0xb4b9, 0xb4c0,
00239 0xb4c4, 0xb4c8, 0xb4d0, 0xb4d5, 0xb4dc, 0xb4dd, 0xb4e0, 0xb4e3,
00240 0xb4e4, 0xb4e6, 0xb4ec, 0xb4ed, 0xb4ef, 0xb4f1, 0xb4f8, 0xb514,
00241 0xb515, 0xb518, 0xb51b, 0xb51c, 0xb524, 0xb525, 0xb527, 0xb528,
00242 0xb529, 0xb52a, 0xb530, 0xb531, 0xb534, 0xb538,
00243 /* 0x36 */
00244 0xb540, 0xb541, 0xb543, 0xb544, 0xb545, 0xb54b, 0xb54c, 0xb54d,
00245 0xb550, 0xb554, 0xb55c, 0xb55d, 0xb55f, 0xb560, 0xb561, 0xb5a0,
00246 0xb5a1, 0xb5a4, 0xb5a8, 0xb5aa, 0xb5ab, 0xb5b0, 0xb5b1, 0xb5b3,
00247 0xb5b4, 0xb5b5, 0xb5bb, 0xb5bc, 0xb5bd, 0xb5c0, 0xb5c4, 0xb5cc,
00248 0xb5cd, 0xb5cf, 0xb5d0, 0xb5d1, 0xb5d8, 0xb5ec, 0xb610, 0xb611,
00249 0xb614, 0xb618, 0xb625, 0xb62c, 0xb634, 0xb648, 0xb664, 0xb668,
00250 0xb69c, 0xb69d, 0xb6a0, 0xb6a4, 0xb6ab, 0xb6ac, 0xb6b1, 0xb6d4,
00251 0xb6f0, 0xb6f4, 0xb6f8, 0xb700, 0xb701, 0xb705, 0xb728, 0xb729,
00252 0xb72c, 0xb72f, 0xb730, 0xb738, 0xb739, 0xb73b, 0xb744, 0xb748,
00253 0xb74c, 0xb754, 0xb755, 0xb760, 0xb764, 0xb768, 0xb770, 0xb771,
00254 0xb773, 0xb775, 0xb77c, 0xb77d, 0xb780, 0xb784, 0xb78c, 0xb78d,
00255 0xb78f, 0xb790, 0xb791, 0xb792, 0xb796, 0xb797,
00256 /* 0x37 */
00257 0xb798, 0xb799, 0xb79c, 0xb7a0, 0xb7a8, 0xb7a9, 0xb7ab, 0xb7ac,
00258 0xb7ad, 0xb7b4, 0xb7b5, 0xb7b8, 0xb7c7, 0xb7c9, 0xb7ec, 0xb7ed,
00259 0xb7f0, 0xb7f4, 0xb7fc, 0xb7fd, 0xb7ff, 0xb800, 0xb801, 0xb807,
00260 0xb808, 0xb809, 0xb80c, 0xb810, 0xb818, 0xb819, 0xb81b, 0xb81d,
00261 0xb824, 0xb825, 0xb828, 0xb82c, 0xb834, 0xb835, 0xb837, 0xb838,
00262 0xb839, 0xb840, 0xb844, 0xb851, 0xb853, 0xb85c, 0xb85d, 0xb860,
00263 0xb864, 0xb86c, 0xb86d, 0xb86f, 0xb871, 0xb878, 0xb87c, 0xb88d,
00264 0xb8a8, 0xb8b0, 0xb8b4, 0xb8b8, 0xb8c0, 0xb8c1, 0xb8c3, 0xb8c5,
00265 0xb8cc, 0xb8d0, 0xb8d4, 0xb8dd, 0xb8df, 0xb8e1, 0xb8e8, 0xb8e9,
00266 0xb8ec, 0xb8f0, 0xb8f8, 0xb8f9, 0xb8fb, 0xb8fd, 0xb904, 0xb918,
00267 0xb920, 0xb93c, 0xb93d, 0xb940, 0xb944, 0xb94c, 0xb94f, 0xb951,
00268 0xb958, 0xb959, 0xb95c, 0xb960, 0xb968, 0xb969,
00269 /* 0x38 */
00270 0xb96b, 0xb96d, 0xb974, 0xb975, 0xb978, 0xb97c, 0xb984, 0xb985,
00271 0xb987, 0xb989, 0xb98a, 0xb98d, 0xb98e, 0xb9ac, 0xb9ad, 0xb9b0,
00272 0xb9b4, 0xb9bc, 0xb9bd, 0xb9bf, 0xb9c1, 0xb9c8, 0xb9c9, 0xb9cc,
00273 0xb9ce, 0xb9cf, 0xb9d0, 0xb9d1, 0xb9d2, 0xb9d8, 0xb9d9, 0xb9db,
00274 0xb9dd, 0xb9de, 0xb9e1, 0xb9e3, 0xb9e4, 0xb9e5, 0xb9e8, 0xb9ec,
00275 0xb9f4, 0xb9f5, 0xb9f7, 0xb9f8, 0xb9f9, 0xb9fa, 0xba00, 0xba01,
00276 0xba08, 0xba15, 0xba38, 0xba39, 0xba3c, 0xba40, 0xba42, 0xba48,
00277 0xba49, 0xba4b, 0xba4d, 0xba4e, 0xba53, 0xba54, 0xba55, 0xba58,
00278 0xba5c, 0xba64, 0xba65, 0xba67, 0xba68, 0xba69, 0xba70, 0xba71,
00279 0xba74, 0xba78, 0xba83, 0xba84, 0xba85, 0xba87, 0xba8c, 0baa8,
00280 0xbaa9, 0baaab, 0baaac, 0bab0, 0bab2, 0bab8, 0bab9, 0babbb,
00281 0babd, 0bac4, 0bac8, 0bad8, 0bad9, 0bafcc,
00282 /* 0x39 */
00283 0xbb00, 0xbb04, 0xbb0d, 0xbb0f, 0xbb11, 0xbb18, 0xbb1c, 0xbb20,
00284 0xbb29, 0xbb2b, 0xbb34, 0xbb35, 0xbb36, 0xbb38, 0xbb3b, 0xbb3c,
00285 0xbb3d, 0xbb3e, 0xbb44, 0xbb45, 0xbb47, 0xbb49, 0xbb4d, 0xbb4f,
00286 0xbb50, 0xbb54, 0xbb58, 0xbb61, 0xbb63, 0xbb6c, 0xbb88, 0xbb8c,
00287 0xbb90, 0xbba4, 0xbba8, 0bbbac, 0bbb4, 0bbb7, 0bbbc0, 0bbbc4,
00288 0bbbc8, 0bbbd0, 0bbbd3, 0bbbf8, 0bbbf9, 0bbbfc, 0bbbf, 0bbc00,
00289 0bbc02, 0bbc08, 0bbc09, 0bbc0b, 0bbc0c, 0bbc0d, 0bbc0f, 0bbc11,
00290 0bbc14, 0bbc15, 0bbc16, 0bbc17, 0bbc18, 0bbc1b, 0bbc1c, 0bbc1d,
00291 0bbc1e, 0bbc1f, 0bbc24, 0bbc25, 0bbc27, 0bbc29, 0bbc2d, 0bbc30,
00292 0bbc31, 0bbc34, 0bbc38, 0bbc40, 0bbc41, 0bbc43, 0bbc44, 0bbc45,
```

Generated by Doxygen

```
00380 0xc77e, 0xc783, 0xc784, 0xc785, 0xc787, 0xc788, 0xc789, 0xc78a,
00381 0xc78e, 0xc790, 0xc791, 0xc794, 0xc796, 0xc797, 0xc798, 0xc79a,
00382 0xc7a0, 0xc7a1, 0xc7a3, 0xc7a4, 0xc7a5, 0xc7a6, 0xc7ac, 0xc7ad,
00383 0xc7b0, 0xc7b4, 0xc7bc, 0xc7bd, 0xc7bf, 0xc7c0, 0xc7c1, 0xc7c8,
00384 0xc7c9, 0xc7cc, 0xc7ce, 0xc7d0, 0xc7d8, 0xc7dd, 0xc7e4, 0xc7e8,
00385 0xc7ec, 0xc800, 0xc801, 0xc804, 0xc808, 0xc80a,
00386 /* 0x41 */
00387 0xc810, 0xc811, 0xc813, 0xc815, 0xc816, 0xc81c, 0xc81d, 0xc820,
00388 0xc824, 0xc82c, 0xc82d, 0xc82f, 0xc831, 0xc838, 0xc83c, 0xc840,
00389 0xc848, 0xc849, 0xc84c, 0xc84d, 0xc854, 0xc870, 0xc871, 0xc874,
00390 0xc878, 0xc87a, 0xc880, 0xc881, 0xc883, 0xc885, 0xc886, 0xc887,
00391 0xc88b, 0xc88c, 0xc88d, 0xc894, 0xc89d, 0xc89f, 0xc8a1, 0xc8a8,
00392 0xc8bc, 0xc8bd, 0xc8c4, 0xc8c8, 0xc8cc, 0xc8d4, 0xc8d5, 0xc8d7,
00393 0xc8d9, 0xc8e0, 0xc8e1, 0xc8e4, 0xc8f5, 0xc8fc, 0xc8fd, 0xc900,
00394 0xc904, 0xc905, 0xc906, 0xc90c, 0xc90d, 0xc90f, 0xc911, 0xc918,
00395 0xc92c, 0xc934, 0xc950, 0xc951, 0xc954, 0xc958, 0xc960, 0xc961,
00396 0xc963, 0xc96c, 0xc970, 0xc974, 0xc97c, 0xc988, 0xc989, 0xc98c,
00397 0xc990, 0xc998, 0xc999, 0xc99b, 0xc99d, 0xc9c0, 0xc9c1, 0xc9c4,
00398 0xc9c7, 0xc9c8, 0xc9ca, 0xc9d0, 0xc9d1, 0xc9d3,
00399 /* 0x42 */
00400 0xc9d5, 0xc9d6, 0xc9d9, 0xc9da, 0xc9dc, 0xc9dd, 0xc9e0, 0xc9e2,
00401 0xc9e4, 0xc9e7, 0xc9ec, 0xc9ed, 0xc9ef, 0xc9f0, 0xc9f1, 0xc9f8,
00402 0xc9f9, 0xc9fc, 0xca00, 0xca08, 0xca09, 0xca0b, 0xca0c, 0xca0d,
00403 0xca14, 0xca18, 0xca29, 0xca4c, 0xca4d, 0xca50, 0xca54, 0xca5c,
00404 0xca5d, 0xca5f, 0xca60, 0xca61, 0xca68, 0xca7d, 0xca84, 0xca98,
00405 0xcabc, 0xcab0, 0xcac0, 0xcac4, 0xcacc, 0xcacd, 0xcacf, 0xcad1,
00406 0xcad3, 0xcad8, 0xcad9, 0xcae0, 0xcaec, 0xcaf4, 0xcb08, 0xcb10,
00407 0xcb14, 0xcb18, 0xcb20, 0xcb21, 0xcb41, 0xcb48, 0xcb49, 0xcb4c,
00408 0xcb50, 0xcb58, 0xcb59, 0xcb5d, 0xcb64, 0xcb78, 0xcb79, 0xcb9c,
00409 0xcbb8, 0xcbbd, 0xcbe4, 0xcbe7, 0xcbe9, 0xcc0c, 0xcc0d, 0xcc10,
00410 0xcc14, 0xcc1c, 0xcc1d, 0xcc21, 0xcc22, 0xcc27, 0xcc28, 0xcc29,
00411 0xcc2c, 0xcc2e, 0xcc30, 0xcc38, 0xcc39, 0xcc3b,
00412 /* 0x43 */
00413 0xcc3c, 0xcc3d, 0xcc3e, 0xcc44, 0xcc45, 0xcc48, 0xcc4c, 0xcc54,
00414 0xcc55, 0xcc57, 0xcc58, 0xcc59, 0xcc60, 0xcc64, 0xcc66, 0xcc68,
00415 0xcc70, 0xcc75, 0xcc98, 0xcc99, 0xcc9c, 0xccea, 0xccea8, 0xccea9,
00416 0xcccab, 0xccac, 0xccad, 0xccb4, 0xccb5, 0xccb8, 0xccbc, 0xccc4,
00417 0xcccc5, 0xcccc7, 0xcccc9, 0xcccd0, 0xcccd4, 0xcce4, 0xcceec, 0xcccf0,
00418 0xcd01, 0xcd08, 0xcd09, 0xcd0c, 0xcd10, 0xcd18, 0xcd19, 0xcd1b,
00419 0xcd1d, 0xcd24, 0xcd28, 0xcd2c, 0xcd39, 0xcd5c, 0xcd60, 0xcd64,
00420 0xcd6c, 0xcd6d, 0xcd6f, 0xcd71, 0xcd78, 0xcd88, 0xcd94, 0xcd95,
00421 0xcd98, 0xcd9c, 0xcda4, 0xcda5, 0xcda7, 0xcda9, 0xcdb0, 0xcdc4,
00422 0xcdcc, 0xcdcd, 0xcde8, 0xcdec, 0xcdf0, 0xcdf8, 0xcdf9, 0xcdfb,
00423 0xcdfd, 0xce04, 0xce08, 0xce0c, 0xce14, 0xce19, 0xce20, 0xce21,
00424 0xce24, 0xce28, 0xce30, 0xce31, 0xce33, 0xce35,
00425 /* 0x44 */
00426 0xce58, 0xce59, 0xce5c, 0xce5f, 0xce60, 0xce61, 0xce68, 0xce69,
00427 0xce6b, 0xce6d, 0xce74, 0xce75, 0xce78, 0xce7c, 0xce84, 0xce85,
00428 0xce87, 0xce89, 0xce90, 0xce91, 0xce94, 0xce98, 0xcea0, 0xcea1,
00429 0xcea3, 0xcea4, 0xcea5, 0xceac, 0xcead, 0xcec1, 0xcee4, 0xcee5,
00430 0xcee8, 0xceeb, 0xceec, 0xcef4, 0xcef5, 0xcef7, 0xcef8, 0xcef9,
00431 0xcf00, 0xcf01, 0xcf04, 0xcf08, 0xcf10, 0xcf11, 0xcf13, 0xcf15,
00432 0xcf1c, 0xcf20, 0xcf24, 0xcf2c, 0xcf2d, 0xcf2f, 0xcf30, 0xcf31,
00433 0xcf38, 0xcf3d, 0xcf54, 0xcf55, 0xcf58, 0xcf5c, 0xcf64, 0xcf65, 0xcf67,
00434 0xcf69, 0xcf70, 0xcf71, 0xcf74, 0xcf78, 0xcf80, 0xcf85, 0xcf8c,
00435 0xcfa1, 0xcfa8, 0xcfb0, 0xcfc4, 0xcfe0, 0xcfe1, 0xcfe4, 0xcfe8,
00436 0xcff0, 0xcff1, 0xcff3, 0xcff5, 0xcffc, 0xd000, 0xd004, 0xd011,
00437 0xd018, 0xd02d, 0xd034, 0xd035, 0xd038, 0xd03c,
00438 /* 0x45 */
00439 0xd044, 0xd045, 0xd047, 0xd049, 0xd050, 0xd054, 0xd058, 0xd060,
00440 0xd06c, 0xd06d, 0xd070, 0xd074, 0xd07c, 0xd07d, 0xd081, 0xd0a4,
00441 0xd0a5, 0xd0a8, 0xd0ac, 0xd0b4, 0xd0b5, 0xd0b7, 0xd0b9, 0xd0c0,
00442 0xd0c1, 0xd0c4, 0xd0c8, 0xd0c9, 0xd0d0, 0xd0d1, 0xd0d3, 0xd0d4,
00443 0xd0d5, 0xd0dc, 0xd0dd, 0xd0e0, 0xd0e4, 0xd0ec, 0xd0ed, 0xd0ef,
00444 0xd0f0, 0xd0f1, 0xd0f8, 0xd10d, 0xd130, 0xd131, 0xd134, 0xd138,
00445 0xd13a, 0xd140, 0xd141, 0xd143, 0xd144, 0xd145, 0xd14c, 0xd14d,
00446 0xd150, 0xd154, 0xd15c, 0xd15d, 0xd15f, 0xd161, 0xd168, 0xd16c,
00447 0xd17c, 0xd184, 0xd188, 0xd1a0, 0xd1a1, 0xd1a4, 0xd1a8, 0xd1b0,
00448 0xd1b1, 0xd1b3, 0xd1b5, 0xd1ba, 0xd1bc, 0xd1c0, 0xd1d8, 0xd1f4,
00449 0xd1f8, 0xd207, 0xd209, 0xd210, 0xd22c, 0xd22d, 0xd230, 0xd234,
00450 0xd23c, 0xd23d, 0xd23f, 0xd241, 0xd248, 0xd25c,
00451 /* 0x46 */
00452 0xd264, 0xd280, 0xd281, 0xd284, 0xd288, 0xd290, 0xd291, 0xd295,
00453 0xd29c, 0xd2a0, 0xd2a4, 0xd2ac, 0xd2b1, 0xd2b8, 0xd2b9, 0xd2bc,
00454 0xd2bf, 0xd2c0, 0xd2c2, 0xd2c8, 0xd2c9, 0xd2cb, 0xd2d4, 0xd2d8,
00455 0xd2dc, 0xd2e4, 0xd2e5, 0xd2f0, 0xd2f1, 0xd2f4, 0xd2f8, 0xd300,
00456 0xd301, 0xd303, 0xd305, 0xd30c, 0xd30d, 0xd30e, 0xd310, 0xd314,
00457 0xd316, 0xd31c, 0xd31d, 0xd31f, 0xd320, 0xd321, 0xd325, 0xd328,
00458 0xd329, 0xd32c, 0xd330, 0xd338, 0xd339, 0xd33b, 0xd33c, 0xd33d,
00459 0xd344, 0xd345, 0xd37c, 0xd37d, 0xd380, 0xd384, 0xd38c, 0xd38d,
00460 0xd38f, 0xd390, 0xd391, 0xd398, 0xd399, 0xd39c, 0xd3a0, 0xd3a8,
00461 0xd3a9, 0xd3ab, 0xd3ad, 0xd3b4, 0xd3b8, 0xd3bc, 0xd3c4, 0xd3c5,
00462 0xd3c8, 0xd3c9, 0xd3d0, 0xd3d8, 0xd3e1, 0xd3e3, 0xd3ec, 0xd3ed,
00463 0xd3f0, 0xd3f4, 0xd3fc, 0xd3fd, 0xd3ff, 0xd401,
00464 /* 0x47 */
00465 0xd408, 0xd41d, 0xd440, 0xd444, 0xd45c, 0xd460, 0xd464, 0xd46d,
00466 0xd46f, 0xd478, 0xd479, 0xd47c, 0xd47f, 0xd480, 0xd482, 0xd488,
```



```
00467 0xd489, 0xd48b, 0xd48d, 0xd494, 0xd4a9, 0xd4cc, 0xd4d0, 0xd4d4,
00468 0xd4dc, 0xd4df, 0xd4e8, 0xd4ec, 0xd4f0, 0xd4f8, 0xd4fb, 0xd4fd,
00469 0xd504, 0xd508, 0xd50c, 0xd514, 0xd515, 0xd517, 0xd53c, 0xd53d,
00470 0xd540, 0xd544, 0xd54c, 0xd54d, 0xd54f, 0xd551, 0xd558, 0xd559,
00471 0xd55c, 0xd560, 0xd565, 0xd568, 0xd569, 0xd56b, 0xd56d, 0xd574,
00472 0xd575, 0xd578, 0xd57c, 0xd584, 0xd585, 0xd587, 0xd588, 0xd589,
00473 0xd590, 0xd5a5, 0xd5c8, 0xd5c9, 0xd5cc, 0xd5d0, 0xd5d2, 0xd5d8,
00474 0xd5d9, 0xd5db, 0xd5dd, 0xd5e4, 0xd5e5, 0xd5e8, 0xd5ec, 0xd5f4,
00475 0xd5f5, 0xd5f7, 0xd5f9, 0xd600, 0xd601, 0xd604, 0xd608, 0xd610,
00476 0xd611, 0xd613, 0xd614, 0xd615, 0xd61c, 0xd620,
00477 /* 0x48 */
00478 0xd624, 0xd62d, 0xd638, 0xd639, 0xd63c, 0xd640, 0xd645, 0xd648,
00479 0xd649, 0xd64b, 0xd64d, 0xd651, 0xd654, 0xd655, 0xd658, 0xd65c,
00480 0xd667, 0xd669, 0xd670, 0xd671, 0xd674, 0xd683, 0xd685, 0xd68c,
00481 0xd68d, 0xd690, 0xd694, 0xd69d, 0xd69f, 0xd6a1, 0xd6a8, 0xd6ac,
00482 0xd6b0, 0xd6b9, 0xd6bb, 0xd6c4, 0xd6c5, 0xd6c8, 0xd6cc, 0xd6d1,
00483 0xd6d4, 0xd6d7, 0xd6d9, 0xd6e0, 0xd6e4, 0xd6e8, 0xd6f0, 0xd6f5,
00484 0xd6fc, 0xd6fd, 0xd700, 0xd704, 0xd711, 0xd718, 0xd719, 0xd71c,
00485 0xd720, 0xd728, 0xd729, 0xd72b, 0xd72d, 0xd734, 0xd735, 0xd738,
00486 0xd73c, 0xd744, 0xd747, 0xd749, 0xd750, 0xd751, 0xd754, 0xd756,
00487 0xd757, 0xd758, 0xd759, 0xd760, 0xd761, 0xd763, 0xd765, 0xd769,
00488 0xd76c, 0xd770, 0xd774, 0xd77c, 0xd77d, 0xd781, 0xd788, 0xd789,
00489 0xd78c, 0xd790, 0xd798, 0xd799, 0xd79b, 0xd79d,
00490 };
00491 static const unsigned short ksc5601_2uni_page4a[4888] = {
00492 /* 0x4a */
00493 0x4f3d, 0x4f73, 0x5047, 0x50f9, 0x52a0, 0x53ef, 0x5475, 0x54e5,
00494 0x5609, 0x5ac1, 0x5bb6, 0x6687, 0x67b6, 0x67b7, 0x67ef, 0x6b4c,
00495 0x73c2, 0x75c2, 0x7a3c, 0x82db, 0x8304, 0x8857, 0x8888, 0x8a36,
00496 0x8cc8, 0x8dcf, 0x8efb, 0x8fe6, 0x99d5, 0x523b, 0x5374, 0x5404,
00497 0x606a, 0x6164, 0x6bbc, 0x73cf, 0x811a, 0x89ba, 0x89d2, 0x95a3,
00498 0x4f83, 0x520a, 0x58be, 0x5978, 0x59e6, 0x5e72, 0x5e79, 0x61c7,
00499 0x63c0, 0x6746, 0x67ec, 0x687f, 0x6f97, 0x764e, 0x770b, 0x78f5,
00500 0x7a08, 0x7aff, 0x7c21, 0x809d, 0x826e, 0x8271, 0x8aeb, 0x9593,
00501 0x4e6b, 0x559d, 0x66f7, 0x6e34, 0x78a3, 0x7aed, 0x845b, 0x8910,
00502 0x874e, 0x97a8, 0x52d8, 0x574e, 0x582a, 0x5d4c, 0x611f, 0x61be,
00503 0x6221, 0x6562, 0x67d1, 0x6a44, 0x6e1b, 0x7518, 0x75b3, 0x76e3,
00504 0x77b0, 0x7d3a, 0x90af, 0x9451, 0x9452, 0x9f95,
00505 /* 0x4b */
00506 0x5323, 0x5cac, 0x7532, 0x80db, 0x9240, 0x9598, 0x525b, 0x5808,
00507 0x59dc, 0x5ca1, 0x5d17, 0x5eb7, 0x5f3a, 0x5f4a, 0x6177, 0x6c5f,
00508 0x757a, 0x7586, 0x7ce0, 0x7d73, 0x7db1, 0x7f8c, 0x8154, 0x8221,
00509 0x8591, 0x8941, 0x8b1b, 0x92fc, 0x964d, 0x9c47, 0x4ecb, 0x4ef7,
00510 0x500b, 0x51f1, 0x584f, 0x6137, 0x613e, 0x6168, 0x6539, 0x69ea,
00511 0x6f11, 0x75a5, 0x7686, 0x76d6, 0x7b87, 0x82a5, 0x84cb, 0xf900,
00512 0x93a7, 0x958b, 0x5580, 0x5ba2, 0x5751, 0xf901, 0x7cb3, 0x7fb9,
00513 0x91b5, 0x5028, 0x53bb, 0x5c45, 0x5de8, 0x62d2, 0x636e, 0x64da,
00514 0x64e7, 0x6e20, 0x70ac, 0x795b, 0x8ddd, 0x8e1e, 0xf902, 0x907d,
00515 0x9245, 0x92f8, 0x4e7e, 0x4ef6, 0x5065, 0x5dfe, 0x5efa, 0x6106,
00516 0x6957, 0x8171, 0x8654, 0x8e47, 0x9375, 0x9a2b, 0x4e5e, 0x5091,
00517 0x6770, 0x6840, 0x5109, 0x528d, 0x5292, 0x6aa2,
00518 /* 0x4c */
00519 0x77bc, 0x9210, 0x9ed4, 0x52ab, 0x602f, 0x8ff2, 0x5048, 0x61a9,
00520 0x63ed, 0x64ca, 0x683c, 0x6a84, 0x6fc0, 0x8188, 0x89a1, 0x9694,
00521 0x5805, 0x727d, 0x72ac, 0x7504, 0x7d79, 0x7e6d, 0x80a9, 0x898b,
00522 0x8b74, 0x9063, 0x9d51, 0x6289, 0x6c7a, 0x6f54, 0x7d50, 0x7f3a,
00523 0x8a23, 0x517c, 0x614a, 0x7b9d, 0x8b19, 0x9257, 0x938c, 0x4eac,
00524 0x4fd3, 0x501e, 0x50be, 0x5106, 0x52c1, 0x52cd, 0x537f, 0x5770,
00525 0x5883, 0x5e9a, 0x5f91, 0x6176, 0x61ac, 0x64ce, 0x656c, 0x666f,
00526 0x66bb, 0x66f4, 0x6897, 0x6d87, 0x7085, 0x70f1, 0x749f, 0x74a5,
00527 0x74ca, 0x75d9, 0x786c, 0x78ec, 0x7adf, 0x7af6, 0x7d45, 0x7d93,
00528 0x8015, 0x803f, 0x811b, 0x8396, 0x8b66, 0x8f15, 0x9015, 0x93e1,
00529 0x9803, 0x9838, 0x9a5a, 0x9be8, 0x4fc2, 0x5553, 0x583a, 0x5951,
00530 0x5b63, 0x5c46, 0x60b8, 0x6212, 0x6842, 0x68b0,
00531 /* 0x4d */
00532 0x68e8, 0x6eaa, 0x754c, 0x7678, 0x78ce, 0x7a3d, 0x7cfb, 0x7e6b,
00533 0x7e7c, 0x8a08, 0x8aa1, 0x8c3f, 0x968e, 0x9dc4, 0x53e4, 0x53e9,
00534 0x544a, 0x5471, 0x56fa, 0x59d1, 0x5b64, 0x5c3b, 0x5eab, 0x62f7,
00535 0x6537, 0x6545, 0x6572, 0x66a0, 0x67af, 0x69c1, 0x6cbd, 0x75fc,
00536 0x7690, 0x777e, 0x7a3f, 0x7f94, 0x8003, 0x80a1, 0x818f, 0x82e6,
00537 0x82fd, 0x83f0, 0x85c1, 0x8831, 0x88b4, 0x8aa5, 0xf903, 0x8f9c,
00538 0x932e, 0x96c7, 0x9867, 0x9ad8, 0x9f13, 0x54ed, 0x659b, 0x66f2,
00539 0x688f, 0x7a40, 0x8c37, 0x9d60, 0x56f0, 0x5764, 0x5d11, 0x6606,
00540 0x68b1, 0x68cd, 0x6efe, 0x7428, 0x889e, 0x9be4, 0x6c68, 0xf904,
00541 0x9aa8, 0x4f9b, 0x516c, 0x5171, 0x529f, 0x5b54, 0x5de5, 0x6050,
00542 0x606d, 0x62f1, 0x63a7, 0x653b, 0x73d9, 0x7a7a, 0x86a3, 0x8ca2,
00543 0x978f, 0x4e32, 0x5be1, 0x6208, 0x679c, 0x74dc,
00544 /* 0x4e */
00545 0x79d1, 0x83d3, 0x8a87, 0x8ab2, 0x8de8, 0x904e, 0x934b, 0x9846,
00546 0x5ed3, 0x69e8, 0x85ff, 0x90ed, 0xf905, 0x51a0, 0x5b98, 0x5bec,
00547 0x6163, 0x68fa, 0x6b3e, 0x704c, 0x742f, 0x74d8, 0x7ba1, 0x7f50,
00548 0x83c5, 0x89c0, 0x8cab, 0x95dc, 0x9928, 0x522e, 0x605d, 0x62ec,
00549 0x9002, 0x4f8a, 0x5149, 0x5321, 0x58d9, 0x5ee3, 0x66e0, 0x6d38,
00550 0x709a, 0x72c2, 0x73d6, 0x7b50, 0x80f1, 0x945b, 0x5366, 0x639b,
00551 0x7f6b, 0x4e56, 0x5080, 0x584a, 0x58de, 0x602a, 0x6127, 0x62d0,
00552 0x69d0, 0x9b41, 0x5b8f, 0x7d18, 0x80b1, 0x8f5f, 0x4ea4, 0x50d1,
00553 0x54ac, 0x55ac, 0x5b0c, 0x5da0, 0x5de7, 0x652a, 0x654e, 0x6821,
```



```
00554 0x6a4b, 0x72e1, 0x768e, 0x77ef, 0x7d5e, 0x7ff9, 0x81a0, 0x854e,
00555 0x86df, 0x8f03, 0x8f4e, 0x90ca, 0x9903, 0x9a55, 0x9bab, 0x4e18,
00556 0x4e45, 0x4e5d, 0x4ec7, 0x4ff1, 0x5177, 0x52fe,
00557 /* 0x4f */
00558 0x5340, 0x53e3, 0x53e5, 0x548e, 0x5614, 0x5775, 0x57a2, 0x5bc7,
00559 0x5d87, 0x5ed0, 0x61fc, 0x62d8, 0x6551, 0x67b8, 0x67e9, 0x69cb,
00560 0x6b50, 0x6bc6, 0x6bec, 0x6c42, 0x6e9d, 0x7078, 0x72d7, 0x7396,
00561 0x7403, 0x77bf, 0x77e9, 0x7a76, 0x7d7f, 0x8009, 0x81fc, 0x8205,
00562 0x820a, 0x82df, 0x8862, 0x8b33, 0x8cfc, 0x8ec0, 0x9011, 0x90b1,
00563 0x9264, 0x92b6, 0x99d2, 0x9a45, 0x9ce9, 0x9dd7, 0x9f9c, 0x570b,
00564 0x5c40, 0x83ca, 0x97a0, 0x97ab, 0x9eb4, 0x541b, 0x7a98, 0x7fa4,
00565 0x88d9, 0x8ecd, 0x90e1, 0x5800, 0x5c48, 0x6398, 0x7a9f, 0x5bae,
00566 0x5f13, 0x7a79, 0x7aae, 0x828e, 0x8eac, 0x5026, 0x5238, 0x52f8,
00567 0x5377, 0x5708, 0x62f3, 0x6372, 0x6b0a, 0x6dc3, 0x7737, 0x53a5,
00568 0x7357, 0x8568, 0x8e76, 0x95d5, 0x673a, 0x6ac3, 0x6f70, 0x8a6d,
00569 0x8ecc, 0x994b, 0xf906, 0x6677, 0x6b78, 0x8cb4,
00570 /* 0x50 */
00571 0x9b3c, 0xf907, 0x53eb, 0x572d, 0x594e, 0x63c6, 0x69fb, 0x73ea,
00572 0x7845, 0x7aba, 0x7ac5, 0x7cfe, 0x8475, 0x898f, 0x8d73, 0x9035,
00573 0x95a8, 0x52fb, 0x5747, 0x7547, 0x7b60, 0x83cc, 0x921e, 0xf908,
00574 0x6a58, 0x514b, 0x524b, 0x5287, 0x621f, 0x68d8, 0x6975, 0x9699,
00575 0x50c5, 0x52a4, 0x52e4, 0x61c3, 0x65a4, 0x6839, 0x69ff, 0x747e,
00576 0x7b4b, 0x82b9, 0x83eb, 0x89b2, 0x8b39, 0x8fd1, 0x9949, 0xf909,
00577 0x4eca, 0x5997, 0x64d2, 0x6611, 0x6a8e, 0x7434, 0x7981, 0x79bd,
00578 0x82a9, 0x887e, 0x887f, 0x895f, 0xf90a, 0x9326, 0x4f0b, 0x53ca,
00579 0x6025, 0x6271, 0x6c72, 0x7d1a, 0x7d66, 0x4e98, 0x5162, 0x77dc,
00580 0x80af, 0x4f01, 0x4f0e, 0x5176, 0x5180, 0x55dc, 0x5668, 0x573b,
00581 0x57fa, 0x57fc, 0x5914, 0x5947, 0x5993, 0x5bc4, 0x5c90, 0x5d0e,
00582 0x5df1, 0x5e7e, 0x5fcc, 0x6280, 0x65d7, 0x65e3,
00583 /* 0x51 */
00584 0x671e, 0x671f, 0x675e, 0x68cb, 0x68c4, 0x6a5f, 0x6b3a, 0x6c23,
00585 0x6c7d, 0x6c82, 0x6dc7, 0x7398, 0x7426, 0x742a, 0x7482, 0x74a3,
00586 0x7578, 0x757f, 0x7881, 0x78ef, 0x7941, 0x7947, 0x7948, 0x797a,
00587 0x7b95, 0x7d00, 0x7dba, 0x7f88, 0x8006, 0x802d, 0x808c, 0x8a18,
00588 0x8b4f, 0x8c48, 0x8d77, 0x9321, 0x9324, 0x98e2, 0x9951, 0x9a0e,
00589 0x9a0f, 0x9a65, 0x9e92, 0x7dca, 0x4f76, 0x5409, 0x62ee, 0x6854,
00590 0x91d1, 0x55ab, 0x513a, 0xf90b, 0xf90c, 0x5a1c, 0x61e6, 0xf90d,
00591 0x62cf, 0x62ff, 0xf90e, 0xf90f, 0xf910, 0xf911, 0xf912, 0xf913,
00592 0x90a3, 0xf914, 0xf915, 0xf916, 0xf917, 0xf918, 0x8afe, 0xf919,
00593 0xf91a, 0xf91b, 0xf91c, 0x6696, 0xf91d, 0x7156, 0xf91e, 0xf91f,
00594 0x96e3, 0xf920, 0x634f, 0x637a, 0x5357, 0xf921, 0x678f, 0x6960,
00595 0x6e73, 0xf922, 0xf923, 0xf924, 0xf925,
00596 /* 0x52 */
00597 0x7d0d, 0xf926, 0xf927, 0x8872, 0x56ca, 0x5a18, 0xf928, 0xf929,
00598 0xf92a, 0xf92b, 0xf92c, 0x4e43, 0xf92d, 0x5167, 0x5948, 0x67f0,
00599 0x8010, 0xf92e, 0x5973, 0x5e74, 0x649a, 0x79ca, 0x5ff5, 0x606c,
00600 0x62c8, 0x637b, 0x5be7, 0x5bd7, 0x52aa, 0xf92f, 0x5974, 0x5f29,
00601 0x6012, 0xf930, 0xf931, 0xf932, 0x7459, 0xf933, 0xf934, 0xf935,
00602 0xf936, 0xf937, 0xf938, 0x99d1, 0xf939, 0xf93a, 0xf93b, 0xf93c,
00603 0xf93d, 0xf93e, 0xf93f, 0xf940, 0xf941, 0xf942, 0xf943, 0x6fc3,
00604 0xf944, 0xf945, 0x81bf, 0x8fb2, 0x60f1, 0xf946, 0xf947, 0x8166,
00605 0xf948, 0xf949, 0x5c3f, 0xf94a, 0xf94b, 0xf94c, 0xf94d, 0xf94e,
00606 0xf94f, 0xf950, 0xf951, 0x5ae9, 0x8a25, 0x677b, 0x7d10, 0xf952,
00607 0xf953, 0xf954, 0xf955, 0xf956, 0xf957, 0x80fd, 0xf958, 0xf959,
00608 0x5c3c, 0x6ce5, 0x533f, 0x6eba, 0x591a, 0x8336,
00609 /* 0x53 */
00610 0x4e39, 0x4eb6, 0x4f46, 0x55ae, 0x5718, 0x58c7, 0x5f56, 0x65b7,
00611 0x65e6, 0x6a80, 0x6bb5, 0x6e4d, 0x77ed, 0x7aef, 0x7c1e, 0x7dde,
00612 0x86cb, 0x8892, 0x9132, 0x935b, 0x64bb, 0x6fbe, 0x737a, 0x75b8,
00613 0x9054, 0x5556, 0x574d, 0x61ba, 0x64d4, 0x66c7, 0x6de1, 0x6e5b,
00614 0x6f6d, 0x6fb9, 0x75f0, 0x8043, 0x81bd, 0x8541, 0x8983, 0x8ac7,
00615 0x8b5a, 0x931f, 0x6c93, 0x7553, 0x7b54, 0x8e0f, 0x905d, 0x5510,
00616 0x5802, 0x5858, 0x5e62, 0x6207, 0x649e, 0x68e0, 0x7576, 0x7cd6,
00617 0x87b3, 0x9ee8, 0x4ee3, 0x5788, 0x576e, 0x5927, 0x5c0d, 0x5cb1,
00618 0x5e36, 0x5f85, 0x6234, 0x64e1, 0x73b3, 0x81fa, 0x888b, 0x8cb8,
00619 0x968a, 0x9edb, 0x5b85, 0x5fb7, 0x60b3, 0x5012, 0x5200, 0x5230,
00620 0x5716, 0x5835, 0x5857, 0x5c0e, 0x5c60, 0x5cf6, 0x5d8b, 0x5ea6,
00621 0x5f92, 0x60bc, 0x6311, 0x6389, 0x6417, 0x6843,
00622 /* 0x54 */
00623 0x68f9, 0x6ac2, 0x6dd8, 0x6e21, 0x6ed4, 0x6fe4, 0x71fe, 0x76dc,
00624 0x7779, 0x79b1, 0x7a3b, 0x8404, 0x89a9, 0x8ced, 0x8df3, 0x8e48,
00625 0x9003, 0x9014, 0x9053, 0x90fd, 0x934d, 0x9676, 0x97dc, 0x6bd2,
00626 0x7006, 0x7258, 0x72a2, 0x7368, 0x7763, 0x79bf, 0x7be4, 0x7e9b,
00627 0x8b80, 0x58a9, 0x60c7, 0x6566, 0x65fd, 0x66be, 0x6c8c, 0x711e,
00628 0x71c9, 0x8c5a, 0x9813, 0x4e6d, 0x7a81, 0x4edd, 0x51ac, 0x51cd,
00629 0x52d5, 0x540c, 0x61a7, 0x6771, 0x6850, 0x68df, 0x6d1e, 0x6ff7,
00630 0x75bc, 0x77b3, 0x7ae5, 0x80f4, 0x8463, 0x9285, 0x515c, 0x6597,
00631 0x675c, 0x6793, 0x75d8, 0x7ac7, 0x8373, 0xf95a, 0x8c46, 0x9017,
00632 0x982d, 0x5c6f, 0x81c0, 0x829a, 0x9041, 0x906f, 0x920d, 0x5f97,
00633 0x5d9d, 0x6a59, 0x71c8, 0x767b, 0x7b49, 0x85e4, 0x8b04, 0x9127,
00634 0x9a30, 0x5587, 0x61f6, 0xf95b, 0x7669, 0x7f85,
00635 /* 0x55 */
00636 0x863f, 0x87ba, 0x88f8, 0x908f, 0xf95c, 0x6d1b, 0x70d9, 0x73de,
00637 0x7d61, 0x843d, 0xf95d, 0x916a, 0x99f1, 0xf95e, 0x4e82, 0x5375,
00638 0x6b04, 0x6b12, 0x703e, 0x721b, 0x862d, 0x9e1e, 0x524c, 0x8fa3,
00639 0x5d50, 0x64e5, 0x652c, 0x6b16, 0x6feb, 0x7c43, 0x7e9c, 0x85cd,
00640 0x8964, 0x89bd, 0x62c9, 0x81d8, 0x881f, 0x5eca, 0x6717, 0x6d6a,
```

```

00641 0x72fc, 0x7405, 0x746f, 0x8782, 0x90de, 0x4f86, 0x5d0d, 0x5fa0,
00642 0x840a, 0x51b7, 0x63a0, 0x7565, 0x4eae, 0x5006, 0x5169, 0x51c9,
00643 0x6881, 0x6a11, 0x7cae, 0x7cb1, 0x7ce7, 0x826f, 0x8ad2, 0x8f1b,
00644 0x91cf, 0x4fb6, 0x5137, 0x52f5, 0x5442, 0x5eec, 0x616e, 0x623e,
00645 0x65c5, 0x6ada, 0x6ffe, 0x792a, 0x85dc, 0x8823, 0x95ad, 0x9a62,
00646 0x9a6a, 0x9e97, 0x9ece, 0x529b, 0x66c6, 0x6b77, 0x701d, 0x792b,
00647 0x8f62, 0x9742, 0x6190, 0x6200, 0x6523, 0x6f23,
00648 /* 0x56 */
00649 0x7149, 0x7489, 0x7df4, 0x806f, 0x84ee, 0x8f26, 0x9023, 0x934a,
00650 0x51bd, 0x5217, 0x52a3, 0x6d0c, 0x70c8, 0x88c2, 0x5ec9, 0x6582,
00651 0x6bae, 0x6fc2, 0x7c3e, 0x7375, 0x4ee4, 0x4f36, 0x56f9, 0xf95f,
00652 0x5cba, 0x5dba, 0x601c, 0x73b2, 0x7b2d, 0x7f9a, 0x7fce, 0x8046,
00653 0x901e, 0x9234, 0x96f6, 0x9748, 0x9818, 0x9f61, 0x4f8b, 0x6fa7,
00654 0x79ae, 0x91b4, 0x96b7, 0x52de, 0xf960, 0x6488, 0x64c4, 0x6ad3,
00655 0x6f5e, 0x7018, 0x7210, 0x76e7, 0x8001, 0x8606, 0x865c, 0x8def,
00656 0x8f05, 0x9732, 0x9b6f, 0x9dfa, 0x9e75, 0x788c, 0x797f, 0x7da0,
00657 0x83c9, 0x9304, 0x9e7f, 0x9e93, 0x8ad6, 0x58df, 0x5f04, 0x6727,
00658 0x7027, 0x74cf, 0x7c60, 0x807e, 0x5121, 0x7028, 0x7262, 0x78ca,
00659 0x8cc2, 0x8cda, 0x8cf4, 0x96f7, 0x4e86, 0x50da, 0x5bee, 0x5ed6,
00660 0x6599, 0x71ce, 0x7642, 0x77ad, 0x804a, 0x84fc,
00661 /* 0x57 */
00662 0x907c, 0x9b27, 0x9f8d, 0x58d8, 0x5a41, 0x5c62, 0x6a13, 0x6dda,
00663 0x6f0f, 0x763b, 0x7d2f, 0x7e37, 0x851e, 0x8938, 0x93e4, 0x964b,
00664 0x5289, 0x65d2, 0x67f3, 0x69b4, 0x6d41, 0x6e9c, 0x7007, 0x7409,
00665 0x7460, 0x7559, 0x7624, 0x786b, 0x8b2c, 0x985e, 0x516d, 0x622e,
00666 0x9678, 0x4f96, 0x502b, 0x5d19, 0x6dea, 0x7db8, 0x8f2a, 0x5f8b,
00667 0x6144, 0x6817, 0xf961, 0x9686, 0x52d2, 0x808b, 0x51dc, 0x51cc,
00668 0x695e, 0x7a1c, 0x7dbe, 0x83f1, 0x9675, 0x4fda, 0x5229, 0x5398,
00669 0x540f, 0x550e, 0x5c65, 0x60a7, 0x674e, 0x68a8, 0x6d6c, 0x7281,
00670 0x72f8, 0x7406, 0x7483, 0xf962, 0x75e2, 0x7c6c, 0x7f79, 0x7fb8,
00671 0x8389, 0x88cf, 0x88e1, 0x91cc, 0x91d0, 0x96e2, 0x9bc9, 0x541d,
00672 0x6f7e, 0x71d0, 0x7498, 0x85fa, 0x8eaa, 0x96a3, 0x9c57, 0x9e9f,
00673 0x6797, 0x6dcb, 0x7433, 0x81e8, 0x9716, 0x782c,
00674 /* 0x58 */
00675 0x7acb, 0x7b20, 0x7c92, 0x6469, 0x746a, 0x75f2, 0x78bc, 0x78e8,
00676 0x99ac, 0x9b54, 0x9ebb, 0x5bde, 0x5e55, 0x6f20, 0x819c, 0x83ab,
00677 0x9088, 0x4e07, 0x534d, 0x5a29, 0x5dd2, 0x5f4e, 0x6162, 0x633d,
00678 0x6669, 0x66fc, 0x6eff, 0x6f2b, 0x7063, 0x779e, 0x842c, 0x8513,
00679 0x883b, 0x8f13, 0x9945, 0x9c3b, 0x551c, 0x62b9, 0x672b, 0x6cab,
00680 0x8309, 0x896a, 0x977a, 0x4ea1, 0x5984, 0x5fd8, 0x5fd9, 0x671b,
00681 0x7db2, 0x7f54, 0x8292, 0x832b, 0x83bd, 0x8f1e, 0x9099, 0x57cb,
00682 0x59b9, 0x5a92, 0x5bd0, 0x6627, 0x679a, 0x6885, 0x6bcf, 0x7164,
00683 0x7f75, 0x8cb7, 0x8ce3, 0x9081, 0x9b45, 0x8108, 0x8c8a, 0x964c,
00684 0x9a40, 0x9ea5, 0x5b5f, 0x6c13, 0x731b, 0x76f2, 0x76df, 0x840c,
00685 0x51aa, 0x8993, 0x514d, 0x5195, 0x52c9, 0x68c9, 0x6c94, 0x7704,
00686 0x7720, 0x7dbf, 0x7dec, 0x9762, 0x9eb5, 0x6ec5,
00687 /* 0x59 */
00688 0x8511, 0x51a5, 0x540d, 0x547d, 0x660e, 0x669d, 0x6927, 0x6e9f,
00689 0x76bf, 0x7791, 0x8317, 0x84c2, 0x879f, 0x9169, 0x9298, 0x9cf4,
00690 0x8882, 0x4fae, 0x5192, 0x52df, 0x59c6, 0x5e3d, 0x6155, 0x6478,
00691 0x6479, 0x66ae, 0x67d0, 0x6a21, 0x6bcd, 0x6bdb, 0x725f, 0x7261,
00692 0x7441, 0x7738, 0x77db, 0x8017, 0x82bc, 0x8305, 0x8b00, 0x8b28,
00693 0x8c8c, 0x6728, 0x6c90, 0x7267, 0x76ee, 0x7766, 0x7a46, 0x9da9,
00694 0x6b7f, 0x6c92, 0x6922, 0x6726, 0x8499, 0x536f, 0x5893, 0x5999,
00695 0x5edf, 0x63cf, 0x6634, 0x6773, 0x6e3a, 0x732b, 0x7ad7, 0x82d7,
00696 0x9328, 0x52d9, 0x5deb, 0x61ae, 0x61cb, 0x620a, 0x62c7, 0x64ab,
00697 0x65e0, 0x6959, 0x6b66, 0x6bcb, 0x7121, 0x73f7, 0x755d, 0x7e46,
00698 0x821e, 0x8302, 0x856a, 0x8aa3, 0x8cbf, 0x9727, 0x9d61, 0x58a8,
00699 0x9ed8, 0x5011, 0x520e, 0x543b, 0x554f, 0x6587,
00700 /* 0x5a */
00701 0x6c76, 0x7d0a, 0x7d0b, 0x805e, 0x868a, 0x9580, 0x96ef, 0x52ff,
00702 0x6c95, 0x7269, 0x5473, 0x5a9a, 0x5c3e, 0x5d4b, 0x5f4c, 0x5fae,
00703 0x672a, 0x68b6, 0x6963, 0x6e3c, 0x6e44, 0x7709, 0x7c73, 0x7f8e,
00704 0x8587, 0x8b0e, 0x8ff7, 0x9761, 0x9ef4, 0x5cb7, 0x60b6, 0x610d,
00705 0x61ab, 0x654f, 0x65fb, 0x65fc, 0x6c11, 0x6cef, 0x739f, 0x73c9,
00706 0x7de1, 0x9594, 0x5bc6, 0x871c, 0x8b10, 0x525d, 0x535a, 0x62cd,
00707 0x640f, 0x64b2, 0x6734, 0x6a38, 0x6cca, 0x73c0, 0x749e, 0x7b94,
00708 0x7c95, 0x7e1b, 0x818a, 0x8236, 0x8584, 0x8feb, 0x96f9, 0x99c1,
00709 0x4f34, 0x534a, 0x53cd, 0x53db, 0x62cc, 0x642c, 0x6500, 0x6591,
00710 0x69c3, 0x6cee, 0x6f58, 0x73ed, 0x7554, 0x7622, 0x76e4, 0x76fc,
00711 0x78d0, 0x78fb, 0x792c, 0x7d46, 0x822c, 0x87e0, 0x8fd4, 0x9812,
00712 0x98ef, 0x52c3, 0x62d4, 0x64a5, 0x6e24, 0x6f51,
00713 /* 0x5b */
00714 0x767c, 0x8dcb, 0x91b1, 0x9262, 0x9aee, 0x9b43, 0x5023, 0x508d,
00715 0x574a, 0x59a8, 0x5c28, 0x5e47, 0x5f77, 0x623f, 0x653e, 0x65b9,
00716 0x65c1, 0x6609, 0x678b, 0x699c, 0x6ec2, 0x78c5, 0x7d21, 0x80aa,
00717 0x8180, 0x822b, 0x82b3, 0x84a1, 0x868c, 0x8a2a, 0x8b17, 0x90a6,
00718 0x9632, 0x9f90, 0x500d, 0x4ff3, 0xf963, 0x57f9, 0x5f98, 0x62dc,
00719 0x6392, 0x676e, 0x6e43, 0x7119, 0x76c3, 0x80cc, 0x80da, 0x88f4,
00720 0x88f5, 0x8919, 0x8ce0, 0x8f29, 0x914d, 0x966a, 0x4f2f, 0x4f70,
00721 0x5e1b, 0x67cf, 0x6822, 0x767d, 0x767e, 0x9b44, 0x5e61, 0x6a0a,
00722 0x7169, 0x71d4, 0x756a, 0xf964, 0x7e41, 0x8543, 0x85e9, 0x98dc,
00723 0x4f10, 0x7b4f, 0x7f70, 0x95a5, 0x51e1, 0x5e06, 0x68b5, 0x6c3e,
00724 0x6c4e, 0x6cdb, 0x72af, 0x7bc4, 0x8303, 0x6cd5, 0x743a, 0x50fb,
00725 0x5288, 0x58c1, 0x64d8, 0x6a97, 0x74a7, 0x7656,
00726 /* 0x5c */
00727 0x78a7, 0x8617, 0x95e2, 0x9739, 0xf965, 0x535e, 0x5f01, 0x8b8a,

```

```
00728 0x8fa8, 0x8faf, 0x908a, 0x5225, 0x77a5, 0x9c49, 0x9f08, 0x4e19,
00729 0x5002, 0x5175, 0x5c5b, 0x5e77, 0x661e, 0x663a, 0x67c4, 0x68c5,
00730 0x70b3, 0x7501, 0x75c5, 0x79c9, 0x7add, 0x8f27, 0x9920, 0x9a08,
00731 0x4fdd, 0x5821, 0x5831, 0x5bf6, 0x666e, 0x6b65, 0x6d11, 0x6e7a,
00732 0x6f7d, 0x73e4, 0x752b, 0x83e9, 0x88dc, 0x8913, 0x8b5c, 0x8f14,
00733 0x4f0f, 0x50d5, 0x5310, 0x535c, 0x5b93, 0x5fa9, 0x670d, 0x798f,
00734 0x8179, 0x832f, 0x8514, 0x8907, 0x8986, 0x8f39, 0x8f3b, 0x99a5,
00735 0x9c12, 0x672c, 0x4e76, 0x4ff8, 0x5949, 0x5c01, 0x5cef, 0x5cf0,
00736 0x6367, 0x68d2, 0x70fd, 0x71a2, 0x742b, 0x7e2b, 0x84ec, 0x8702,
00737 0x9022, 0x92d2, 0x9cf3, 0x4e0d, 0x4ed8, 0x4fef, 0x5085, 0x5256,
00738 0x526f, 0x5426, 0x5490, 0x57e0, 0x592b, 0x5a66,
00739 /* 0x5d */
00740 0x5b5a, 0x5b75, 0x5bcc, 0x5e9c, 0xf966, 0x6276, 0x6577, 0x65a7,
00741 0x6d6e, 0x6ea5, 0x7236, 0x7b26, 0x7c3f, 0x7f36, 0x8150, 0x8151,
00742 0x819a, 0x8240, 0x8299, 0x83a9, 0x8a03, 0x8ca0, 0x8ce6, 0x8cfb,
00743 0x8d74, 0x8dba, 0x90e8, 0x91dc, 0x961c, 0x9644, 0x99d9, 0x9ce7,
00744 0x5317, 0x5206, 0x5429, 0x5674, 0x58b3, 0x5954, 0x596e, 0x5fff,
00745 0x61a4, 0x626e, 0x6610, 0x6c7e, 0x711a, 0x76c6, 0x7c89, 0x7cde,
00746 0x7d1b, 0x82ac, 0x8cc1, 0x96f0, 0xf967, 0x4f5b, 0x5f17, 0x5f7f,
00747 0x62c2, 0x5d29, 0x670b, 0x68da, 0x787c, 0x7e43, 0x9d6c, 0x4e15,
00748 0x5099, 0x5315, 0x532a, 0x5351, 0x5983, 0x5a62, 0x5e87, 0x60b2,
00749 0x618a, 0x6249, 0x6279, 0x6590, 0x6787, 0x69a7, 0x6bd4, 0x6bd6,
00750 0x6bd7, 0x6bd8, 0x6cb8, 0xf968, 0x7435, 0x75fa, 0x7812, 0x7891,
00751 0x79d5, 0x79d8, 0x7c83, 0x7dcf, 0x7fe1, 0x80a5,
00752 /* 0x5e */
00753 0x813e, 0x81c2, 0x83f2, 0x871a, 0x88e8, 0x8ab9, 0x8b6c, 0x8cbb,
00754 0x9119, 0x975e, 0x98db, 0x9f3b, 0x56ac, 0x5b2a, 0x5f6c, 0x658c,
00755 0x6ab3, 0x6baf, 0x6d5c, 0x6ff1, 0x7015, 0x725d, 0x73ad, 0x8ca7,
00756 0x8cd3, 0x983b, 0x6191, 0x6c37, 0x8058, 0x9a01, 0x4e4d, 0x4e8b,
00757 0x4e9b, 0x4ed5, 0x4f3a, 0x4f3c, 0x4f7f, 0x4fdf, 0x50ff, 0x53f2,
00758 0x53f8, 0x5506, 0x55e3, 0x56db, 0x58eb, 0x5962, 0x5a11, 0x5beb,
00759 0x5bfa, 0x5c04, 0x5df3, 0x5e2b, 0x5f99, 0x601d, 0x6368, 0x659c,
00760 0x65af, 0x67f6, 0x67fb, 0x68ad, 0x6b7b, 0x6c99, 0x6cd7, 0x6e23,
00761 0x7009, 0x7345, 0x7802, 0x793e, 0x7940, 0x7960, 0x79c1, 0x7be9,
00762 0x7d17, 0x7d72, 0x8086, 0x820d, 0x838e, 0x84d1, 0x86c7, 0x88df,
00763 0x8a50, 0x8a5e, 0x8b1d, 0x8cdc, 0x8d66, 0x8fad, 0x90aa, 0x98fc,
00764 0x99df, 0x9e9d, 0x524a, 0xf969, 0x6714, 0xf96a,
00765 /* 0x5f */
00766 0x5098, 0x522a, 0x5c71, 0x6563, 0x6c55, 0x73ca, 0x7523, 0x759d,
00767 0x7b97, 0x849c, 0x9178, 0x9730, 0x4e77, 0x6492, 0x6bba, 0x715e,
00768 0x85a9, 0x4e09, 0xf96b, 0x6749, 0x68ee, 0x6e17, 0x829f, 0x8518,
00769 0x886b, 0x63f7, 0x6f81, 0x9212, 0x98af, 0x4e0a, 0x50b7, 0x50cf,
00770 0x511f, 0x5546, 0x55aa, 0x5617, 0x5b40, 0x5c19, 0x5ce0, 0x5e38,
00771 0x5e8a, 0x5ea0, 0x5ec2, 0x60f3, 0x6851, 0x6a61, 0x6e58, 0x723d,
00772 0x7240, 0x72c0, 0x76f8, 0x7965, 0x7bb1, 0x7fd4, 0x88f3, 0x89f4,
00773 0x8a73, 0x8c61, 0x8cde, 0x971c, 0x585e, 0x74bd, 0x8cfd, 0x55c7,
00774 0xf96c, 0x7a61, 0x7d22, 0x8272, 0x7272, 0x751f, 0x7525, 0xf96d,
00775 0x7b19, 0x5885, 0x58fb, 0x5dbc, 0x5e8f, 0x5eb6, 0x5f90, 0x6055,
00776 0x6292, 0x637f, 0x654d, 0x6691, 0x66d9, 0x66f8, 0x6816, 0x68f2,
00777 0x7280, 0x745e, 0x7b6e, 0x7d6e, 0x7dd6, 0x7f72,
00778 /* 0x60 */
00779 0x80e5, 0x8212, 0x85af, 0x897f, 0x8a93, 0x901d, 0x92e4, 0x9ecd,
00780 0x9f20, 0x5915, 0x596d, 0x5e2d, 0x60dc, 0x6614, 0x6673, 0x6790,
00781 0x6c50, 0x6dc5, 0x6f5f, 0x77f3, 0x78a9, 0x84c6, 0x91cb, 0x932b,
00782 0x4ed9, 0x50ca, 0x5148, 0x5584, 0x5b0b, 0x5ba3, 0x6247, 0x657e,
00783 0x65cb, 0x6e32, 0x717d, 0x7401, 0x7444, 0x7487, 0x74bf, 0x766c,
00784 0x79aa, 0x7dda, 0x7e55, 0x7fa8, 0x817a, 0x81b3, 0x8239, 0x861a,
00785 0x87ec, 0x8a75, 0x8de3, 0x9078, 0x9291, 0x9425, 0x994d, 0x9bae,
00786 0x5368, 0x5c51, 0x6954, 0x6cc4, 0x6d29, 0x6e2b, 0x820c, 0x859b,
00787 0x893b, 0x8a2d, 0x8aaa, 0x96ea, 0x9f67, 0x5261, 0x66b9, 0x6bb2,
00788 0x7e96, 0x87fe, 0x8d0d, 0x9583, 0x965d, 0x651d, 0x6d89, 0x71ee,
00789 0xf96e, 0x57ce, 0x59d3, 0x5bac, 0x6027, 0x60fa, 0x6210, 0x661f,
00790 0x665f, 0x7329, 0x73f9, 0x76db, 0x7701, 0x7b6c,
00791 /* 0x61 */
00792 0x8056, 0x8072, 0x8165, 0x8aa0, 0x9192, 0x4e16, 0x52e2, 0x6b72,
00793 0x6d17, 0x7a05, 0x7b39, 0x7d30, 0xf96f, 0x8cb0, 0x53ec, 0x562f,
00794 0x5851, 0x5bb5, 0x5c0f, 0x5c11, 0x5de2, 0x6240, 0x6383, 0x6414,
00795 0x66d2, 0x68b3, 0x6cbc, 0x6d88, 0x6eaf, 0x701f, 0x70a4, 0x71d2,
00796 0x7526, 0x758f, 0x758e, 0x7619, 0x7b11, 0x7be0, 0x7c2b, 0x7d20,
00797 0x7d39, 0x852c, 0x856d, 0x8607, 0x8a34, 0x900d, 0x9061, 0x90b5,
00798 0x92b7, 0x97f6, 0x9a37, 0x4fd7, 0x5c6c, 0x675f, 0x6d91, 0x7c9f,
00799 0x7e8c, 0x8b16, 0x8d16, 0x901f, 0x5b6b, 0x5dfd, 0x640d, 0x84c0,
00800 0x905c, 0x98e1, 0x7387, 0x5b8b, 0x609a, 0x677e, 0x6dde, 0x8a1f,
00801 0x8aa6, 0x9001, 0x980c, 0x5237, 0xf970, 0x7051, 0x788e, 0x9396,
00802 0x8870, 0x91d7, 0x4fee, 0x53d7, 0x55fd, 0x56da, 0x5782, 0x58fd,
00803 0x5ac2, 0x5b88, 0x5cab, 0x5cc0, 0x5e25, 0x6101,
00804 /* 0x62 */
00805 0x620d, 0x624b, 0x6388, 0x641c, 0x6536, 0x6578, 0x6a39, 0x6b8a,
00806 0x6c34, 0x6d19, 0x6f31, 0x71e7, 0x72e9, 0x7378, 0x7407, 0x74b2,
00807 0x7626, 0x7761, 0x79c0, 0x7a57, 0x7aea, 0x7cb9, 0x7df8, 0x7dac,
00808 0x7e61, 0x7f9e, 0x8129, 0x8331, 0x8490, 0x84da, 0x85ea, 0x8896,
00809 0x8ab0, 0x8b90, 0x8f38, 0x9042, 0x9083, 0x916c, 0x929e, 0x92b9,
00810 0x968b, 0x96a7, 0x96a8, 0x96d6, 0x9700, 0x9808, 0x9996, 0x9ad3,
00811 0x9b1a, 0x53d4, 0x587e, 0x5919, 0x5b70, 0x5bbf, 0x6dd1, 0x6f5a,
00812 0x719f, 0x7421, 0x74b9, 0x8085, 0x83fd, 0x5de1, 0x5f87, 0x5faa,
00813 0x6042, 0x65ec, 0x6812, 0x696f, 0x6a53, 0x6b89, 0x6d35, 0x6df3,
00814 0x73e3, 0x76fe, 0x77ac, 0x7b4d, 0x7d14, 0x8123, 0x821c, 0x8340,
```

```

00815 0x84f4, 0x8563, 0x8a62, 0x8ac4, 0x9187, 0x931e, 0x9806, 0x99b4,
00816 0x620c, 0x8853, 0x8ff0, 0x9265, 0x5d07, 0x5d27,
00817 /* 0x63 */
00818 0x5d69, 0x745f, 0x819d, 0x8768, 0x6fd5, 0x62fe, 0x7fd2, 0x8936,
00819 0x8972, 0x4e1e, 0x4e58, 0x50e7, 0x52dd, 0x5347, 0x627f, 0x6607,
00820 0x7e69, 0x8805, 0x965e, 0x4f8d, 0x5319, 0x5636, 0x59cb, 0x5aa4,
00821 0x5c38, 0x5c4e, 0x5c4d, 0x5e02, 0x5f11, 0x6043, 0x65bd, 0x662f,
00822 0x6642, 0x67be, 0x67f4, 0x731c, 0x77e2, 0x793a, 0x7fc5, 0x8494,
00823 0x84cd, 0x8996, 0x8a66, 0x8a69, 0x8ae1, 0x8c55, 0x8c7a, 0x57f4,
00824 0x5bd4, 0x5f0f, 0x606f, 0x62ed, 0x690d, 0x6b96, 0x6e5c, 0x7184,
00825 0x7bd2, 0x8755, 0x8b58, 0x8efe, 0x98df, 0x98fe, 0x4f3b, 0x4f81,
00826 0x4fe1, 0x547b, 0x5a20, 0x5bb8, 0x613c, 0x65b0, 0x6668, 0x71fc,
00827 0x7533, 0x795e, 0x7d33, 0x814e, 0x81e3, 0x8398, 0x85aa, 0x85ce,
00828 0x8703, 0x8a0a, 0x8eab, 0x8f9b, 0xf971, 0x8fc5, 0x5931, 0x5ba4,
00829 0x5be6, 0x6089, 0x5be9, 0x5c0b, 0x5fc3, 0x6c81,
00830 /* 0x64 */
00831 0xf972, 0x6dff1, 0x700b, 0x751a, 0x82af, 0x8af6, 0x4ec0, 0x5341,
00832 0xf973, 0x96d9, 0x6c0f, 0x4e9e, 0x4fc4, 0x5152, 0x555e, 0x5a25,
00833 0x5ce8, 0x6211, 0x7259, 0x82bd, 0x83aa, 0x86fe, 0x8859, 0x8a1d,
00834 0x963f, 0x96c5, 0x9913, 0x9d09, 0x9d5d, 0x580a, 0x5cb3, 0x5dbd,
00835 0x5e44, 0x60e1, 0x6115, 0x63e1, 0x6a02, 0x6e25, 0x9102, 0x9354,
00836 0x984e, 0x9c10, 0x9f77, 0x5b89, 0x5cb8, 0x6309, 0x664f, 0x6848,
00837 0x773c, 0x96c1, 0x978d, 0x9854, 0x9b9f, 0x65a1, 0x8b01, 0x8ecb,
00838 0x95bc, 0x5535, 0x5ca9, 0x5dd6, 0x5eb5, 0x6697, 0x764c, 0x83f4,
00839 0x95c7, 0x58d3, 0x62bc, 0x72ce, 0x9d28, 0x4ef0, 0x592e, 0x600f,
00840 0x663b, 0x6b83, 0x79e7, 0x9d26, 0x5393, 0x54c0, 0x57c3, 0x5d16,
00841 0x611b, 0x66d6, 0x6daf, 0x788d, 0x827e, 0x9698, 0x9744, 0x5384,
00842 0x627c, 0x6396, 0x6db2, 0x7e0a, 0x814b, 0x984d,
00843 /* 0x65 */
00844 0x6afb, 0x7f4c, 0x9daf, 0x9e1a, 0x4e5f, 0x503b, 0x51b6, 0x591c,
00845 0x60f9, 0x63f6, 0x6930, 0x723a, 0x8036, 0xf974, 0x91ce, 0x5f31,
00846 0xf975, 0xf976, 0x7d04, 0x82e5, 0x846f, 0x84bb, 0x85e5, 0x8e8d,
00847 0xf977, 0x4f6f, 0xf978, 0xf979, 0x58e4, 0x5b43, 0x6059, 0x63da,
00848 0x6518, 0x656d, 0x6698, 0xf97a, 0x694a, 0x6a23, 0x6d0b, 0x7001,
00849 0x716c, 0x75d2, 0x760d, 0x79b3, 0x7a70, 0xf97b, 0x7f8a, 0xf97c,
00850 0x8944, 0xf97d, 0x8b93, 0x91c0, 0x967d, 0xf97e, 0x990a, 0x5704,
00851 0x5fa1, 0x65bc, 0x6f01, 0x7600, 0x79a6, 0x8a9e, 0x99ad, 0x9b5a,
00852 0x9f6c, 0x5104, 0x61b6, 0x6291, 0x6a8d, 0x81c6, 0x5043, 0x5830,
00853 0x5f66, 0x7109, 0x8a00, 0x8afa, 0x5b7c, 0x8616, 0x4ffa, 0x513c,
00854 0x5b44, 0x5944, 0x63a9, 0x6df9, 0x5daa, 0x696d, 0x5186, 0x4e88,
00855 0x4f59, 0xf97f, 0xf980, 0xf981, 0x5982, 0xf982,
00856 /* 0x66 */
00857 0xf983, 0x6b5f, 0x6c5d, 0xf984, 0x74b5, 0x7916, 0xf985, 0x8207,
00858 0x8245, 0x8339, 0x8f3f, 0x8f5d, 0xf986, 0x9918, 0xf987, 0xf988,
00859 0xf989, 0x4ea6, 0xf98a, 0x57df, 0x5f79, 0x6613, 0xf98b, 0xf98c,
00860 0x75ab, 0x7e79, 0x8b6f, 0xf98d, 0x9006, 0x9a5b, 0x56a5, 0x5827,
00861 0x59f8, 0x5a1f, 0x5bb4, 0xf98e, 0x5ef6, 0xf98f, 0xf990, 0x6350,
00862 0x633b, 0xf991, 0x693d, 0x6c87, 0x6cbf, 0x6d8e, 0x6d93, 0x6df5,
00863 0x6f14, 0xf992, 0x70df, 0x7136, 0x7159, 0xf993, 0x71c3, 0x71d5,
00864 0xf994, 0x784f, 0x786f, 0xf995, 0x7b75, 0x7de3, 0xf996, 0x7e2f,
00865 0xf997, 0x884d, 0x8edf, 0xf998, 0xf999, 0xf99a, 0x925b, 0xf99b,
00866 0x9cf6, 0xf99c, 0xf99d, 0xf99e, 0x6085, 0x6d85, 0xf99f, 0x71b1,
00867 0xf9a0, 0xf9a1, 0x95b1, 0x53ad, 0xf9a2, 0xf9a3, 0xf9a4, 0x67d3,
00868 0xf9a5, 0x708e, 0x7130, 0x7430, 0x8276, 0x82d2,
00869 /* 0x67 */
00870 0xf9a6, 0x95bb, 0x9ae5, 0x9e7d, 0x66c4, 0xf9a7, 0x71c1, 0x8449,
00871 0xf9a8, 0xf9a9, 0x584b, 0xf9aa, 0xf9ab, 0x5db8, 0x5f71, 0xf9ac,
00872 0x6620, 0x668e, 0x6979, 0x69ae, 0x6c38, 0x6cf3, 0x6e36, 0x6f41,
00873 0x6fda, 0x701b, 0x702f, 0x7150, 0x71df, 0x7370, 0xf9ad, 0x745b,
00874 0xf9ae, 0x74d4, 0x76c8, 0x7a4e, 0x7e93, 0xf9af, 0xf9b0, 0x82f1,
00875 0x8a60, 0x8fce, 0xf9b1, 0x9348, 0xf9b2, 0x9719, 0xf9b3, 0xf9b4,
00876 0x4e42, 0x502a, 0xf9b5, 0x5208, 0x53e1, 0x66f3, 0x6c6d, 0x6fca,
00877 0x730a, 0x777f, 0x7a62, 0x82ae, 0x85dd, 0x8602, 0xf9b6, 0x88d4,
00878 0x8a63, 0x8b7d, 0x8c6b, 0xf9b7, 0x92b3, 0xf9b8, 0x9713, 0x9810,
00879 0x4e94, 0x4f0d, 0x4fc9, 0x50b2, 0x5348, 0x543e, 0x5433, 0x55da,
00880 0x5862, 0x58ba, 0x5967, 0x5a1b, 0x5be4, 0x609f, 0xf9b9, 0x61ca,
00881 0x6556, 0x65ff, 0x6664, 0x68a7, 0x6c5a, 0x6fb3,
00882 /* 0x68 */
00883 0x70cf, 0x71ac, 0x7352, 0x7b7d, 0x8708, 0x8aa4, 0x9c32, 0x9f07,
00884 0x5c4b, 0x6c83, 0x7344, 0x7389, 0x923a, 0x6eab, 0x7465, 0x761f,
00885 0x7a69, 0x7e15, 0x860a, 0x5140, 0x58c5, 0x64c1, 0x74ee, 0x7515,
00886 0x7670, 0x7fc1, 0x9095, 0x96cd, 0x9954, 0x6e26, 0x74e6, 0x7aa9,
00887 0x7aaa, 0x81e5, 0x86d9, 0x8778, 0x8a1b, 0x5a49, 0x5b8c, 0x5b9b,
00888 0x68a1, 0x6900, 0x6d63, 0x73a9, 0x7413, 0x742c, 0x7897, 0x7de9,
00889 0x7feb, 0x8118, 0x8155, 0x839e, 0x8c4c, 0x962e, 0x9811, 0x66f0,
00890 0x5f80, 0x65fa, 0x6789, 0x6c6a, 0x738b, 0x502d, 0x5a03, 0x6b6a,
00891 0x77ee, 0x5916, 0x5d6c, 0x5dcd, 0x7325, 0x754f, 0xf9ba, 0xf9bb,
00892 0x50e5, 0x51f9, 0x582f, 0x592d, 0x5996, 0x59da, 0x5be5, 0xf9bc,
00893 0xf9bd, 0x5da2, 0x62d7, 0x6416, 0x6493, 0x64fe, 0xf9be, 0x66dc,
00894 0xf9bf, 0x6a48, 0xf9c0, 0x71ff, 0x7464, 0xf9c1,
00895 /* 0x69 */
00896 0x7a88, 0x7aaf, 0x7e47, 0x7e5e, 0x8000, 0x8170, 0xf9c2, 0x87ef,
00897 0x8981, 0x8b20, 0x9059, 0xf9c3, 0x9080, 0x9952, 0x617e, 0x6b32,
00898 0x6d74, 0x7e1f, 0x8925, 0x8fb1, 0x4fd1, 0x50ad, 0x5197, 0x52c7,
00899 0x57c7, 0x5889, 0x5bb9, 0x5eb8, 0x6142, 0x6995, 0x6d8c, 0x6e67,
00900 0x6eb6, 0x7194, 0x7462, 0x7528, 0x752c, 0x8073, 0x8338, 0x84c9,
00901 0x8e0a, 0x9394, 0x93de, 0xf9c4, 0x4e8e, 0x4f51, 0x5076, 0x512a,

```

```
00902 0x53c8, 0x53cb, 0x53f3, 0x5b87, 0x5bd3, 0x5c24, 0x611a, 0x6182,
00903 0x65f4, 0x725b, 0x7397, 0x7440, 0x76c2, 0x7950, 0x7991, 0x79b9,
00904 0x7d06, 0x7fbd, 0x828b, 0x85d5, 0x865e, 0x8fc2, 0x9047, 0x90f5,
00905 0x91ea, 0x9685, 0x96e8, 0x96e9, 0x52d6, 0x5f67, 0x65ed, 0x6631,
00906 0x682f, 0x715c, 0x7a36, 0x90c1, 0x980a, 0x4e91, 0xf9c5, 0x6a52,
00907 0x6b9e, 0x6f90, 0x7189, 0x8018, 0x82b8, 0x8553,
00908 /* 0x6a */
00909 0x904b, 0x9695, 0x96f2, 0x97fb, 0x851a, 0x9b31, 0x4e90, 0x718a,
00910 0x96c4, 0x5143, 0x539f, 0x54e1, 0x5713, 0x5712, 0x57a3, 0x5a9b,
00911 0x5ac4, 0x5bc3, 0x6028, 0x613f, 0x63f4, 0x6c85, 0x6d39, 0x6e72,
00912 0x6e90, 0x7230, 0x733f, 0x7457, 0x82d1, 0x8881, 0x8f45, 0x9060,
00913 0xf9c6, 0x9662, 0x9662, 0x9858, 0x9d1b, 0x6708, 0x8d8a, 0x925e, 0x4f4d,
00914 0x5049, 0x50de, 0x5371, 0x570d, 0x59d4, 0x5a01, 0x5c09, 0x6170,
00915 0x6690, 0x6e20, 0x7232, 0x744b, 0x7def, 0x80c3, 0x840e, 0x8466,
00916 0x853f, 0x875f, 0x885b, 0x8918, 0x8b02, 0x9055, 0x97cb, 0x9b4f,
00917 0x4e73, 0x4f91, 0x5112, 0x516a, 0xf9c7, 0x552f, 0x55a9, 0x5b7a,
00918 0x5ba5, 0x5e7c, 0x5e7d, 0x5ebe, 0x60a0, 0x60df, 0x6108, 0x6109,
00919 0x63c4, 0x6538, 0x6709, 0xf9c8, 0x67d4, 0x67da, 0xf9c9, 0x6961,
00920 0x6962, 0x6cb9, 0x6d27, 0xf9ca, 0x6e38, 0xf9cb,
00921 /* 0x6b */
00922 0x6fe1, 0x7336, 0x7337, 0xf9cc, 0x745c, 0x7531, 0xf9cd, 0x7652,
00923 0xf9ce, 0xf9cf, 0x7dad, 0x81fe, 0x8438, 0x88d5, 0x8a98, 0x8adb,
00924 0x8aed, 0x8e30, 0x8e42, 0x904a, 0x903e, 0x907a, 0x9149, 0x91c9,
00925 0x936e, 0xf9d0, 0xf9d1, 0x5809, 0xf9d2, 0x6bd3, 0x8089, 0x80b2,
00926 0xf9d3, 0xf9d4, 0x5141, 0x596b, 0x5c39, 0xf9d5, 0xf9d6, 0x6f64,
00927 0x73a7, 0x80e4, 0x8d07, 0xf9d7, 0x9217, 0x958f, 0xf9d8, 0xf9d9,
00928 0xf9da, 0xf9db, 0x807f, 0x620e, 0x701c, 0x7d68, 0x878d, 0xf9dc,
00929 0x57a0, 0x6069, 0x6147, 0x6bb7, 0x8abe, 0x9280, 0x96b1, 0x4e59,
00930 0x541f, 0x6deb, 0x852d, 0x9670, 0x97f3, 0x98ee, 0x63d6, 0x6ce3,
00931 0x9091, 0x51dd, 0x61c9, 0x81ba, 0x9df9, 0x4f9d, 0x501a, 0x5100,
00932 0x5b9c, 0x610f, 0x61ff, 0x64ec, 0x6905, 0x6bc5, 0x7591, 0x77e3,
00933 0x7fa9, 0x8264, 0x858f, 0x87fb, 0x8863, 0x8abc,
00934 /* 0x6c */
00935 0x8b70, 0x91ab, 0x4e8c, 0x4ee5, 0x4f0a, 0xf9dd, 0xf9de, 0x5937,
00936 0x59e8, 0xf9df, 0x5df2, 0x5f1b, 0x5f5b, 0x6021, 0xf9e0, 0xf9e1,
00937 0xf9e2, 0xf9e3, 0x723e, 0x723e, 0xf9e4, 0x7570, 0x75cd, 0xf9e5,
00938 0x79fb, 0xf9e6, 0x800c, 0x8033, 0x8084, 0x82e1, 0x8351, 0xf9e7,
00939 0xf9e8, 0x8cb0, 0x8cb3, 0x9087, 0xf9e9, 0xf9ea, 0x98f4, 0x990c,
00940 0xf9eb, 0xf9ec, 0x7037, 0x76ca, 0x7fca, 0x7fcc, 0x7ffc, 0x8b1a,
00941 0x4eba, 0x4ec1, 0x5203, 0x5370, 0xf9ed, 0x54bd, 0x56e0, 0x59fb,
00942 0x5bc5, 0x5f15, 0x5fcd, 0x6e6e, 0xf9ee, 0xf9ef, 0x7d6a, 0x8335,
00943 0xf9f0, 0x8693, 0x8a8d, 0xf9f1, 0x976d, 0x9777, 0xf9f2, 0xf9f3,
00944 0x4e00, 0x4f5a, 0x4f7e, 0x58f9, 0x65e5, 0x6ea2, 0x9038, 0x93b0,
00945 0x99b9, 0x4efb, 0x58ec, 0x598a, 0x59d9, 0x6041, 0xf9f4, 0xf9f5,
00946 0x7a14, 0xf9f6, 0x834f, 0x8cc3, 0x5165, 0x5344,
00947 /* 0x6d */
00948 0xf9f7, 0xf9f8, 0xf9f9, 0x4ecd, 0x5269, 0x5b55, 0x82bf, 0x4ed4,
00949 0x523a, 0x54a8, 0x59c9, 0x59ff, 0x5b57, 0x5b5c, 0x6063,
00950 0x6148, 0x6ecb, 0x7099, 0x716e, 0x7386, 0x74f7, 0x75b5, 0x78c1,
00951 0x7d2b, 0x8005, 0x81ea, 0x8328, 0x8517, 0x85c9, 0x8aee, 0x8cc7,
00952 0x96cc, 0x4f5c, 0x52fa, 0x56bc, 0x65ab, 0x6628, 0x707c, 0x70b8,
00953 0x7235, 0x7dbd, 0x828d, 0x914c, 0x96c0, 0x9d72, 0x5b71, 0x68e7,
00954 0x6b98, 0x6f7a, 0x76de, 0x5c91, 0x66ab, 0x6f5b, 0x7bb4, 0x7c2a,
00955 0x8836, 0x96dc, 0x4e08, 0x4ed7, 0x5320, 0x5834, 0x58bb, 0x58ef,
00956 0x596c, 0x5c07, 0x5e33, 0x5e84, 0x5f35, 0x638c, 0x66b2, 0x6756,
00957 0x6a1f, 0x6aa3, 0x6b0c, 0x6f3f, 0x7246, 0xf9fa, 0x7350, 0x748b,
00958 0x7ae0, 0x7ca7, 0x8178, 0x81df, 0x81e7, 0x838a, 0x846c, 0x8523,
00959 0x8594, 0x85cf, 0x88dd, 0x8d13, 0x91ac, 0x9577,
00960 /* 0x6e */
00961 0x969c, 0x518d, 0x54c9, 0x5728, 0x5bb0, 0x624d, 0x6750, 0x683d,
00962 0x6893, 0x6e3d, 0x6ed3, 0x707d, 0x7e21, 0x88c1, 0x8ca1, 0x8f09,
00963 0x9f4b, 0x9f4e, 0x722d, 0x7b8f, 0x8acd, 0x931a, 0x4f47, 0x4f4e,
00964 0x5132, 0x5480, 0x59d0, 0x5e95, 0x62b5, 0x6775, 0x696e, 0x6a17,
00965 0x6cae, 0x6e1a, 0x72d9, 0x732a, 0x75bd, 0x7bb8, 0x7d35, 0x82e7,
00966 0x83f9, 0x8457, 0x85f7, 0x8a5b, 0x8caf, 0x8e87, 0x9019, 0x90b8,
00967 0x96ce, 0x9f5f, 0x52e3, 0x540a, 0x5ae1, 0x5bc2, 0x6458, 0x6575,
00968 0x6ef4, 0x72c4, 0xf9fb, 0x7684, 0x7a4d, 0x7b1b, 0x7c4d, 0x7e3e,
00969 0x7fdf, 0x837b, 0x8b2b, 0x8cca, 0x8d64, 0x8de1, 0x8e5f, 0x8fea,
00970 0x8ff9, 0x9069, 0x93d1, 0x4f43, 0x4f7a, 0x50b3, 0x5168, 0x5178,
00971 0x524d, 0x526a, 0x5861, 0x587c, 0x5960, 0x5c08, 0x5c55, 0x5edb,
00972 0x609b, 0x6230, 0x6813, 0x6bbf, 0x6c08, 0x6fb1,
00973 /* 0x6f */
00974 0x714e, 0x7420, 0x7530, 0x7538, 0x7551, 0x7672, 0x7b4c, 0x7b8b,
00975 0x7bad, 0x7bc6, 0x7e8f, 0x8a6e, 0x8f3e, 0x8f49, 0x923f, 0x9293,
00976 0x9322, 0x942b, 0x96fb, 0x985a, 0x986b, 0x991e, 0x5207, 0x622a,
00977 0x6298, 0x6d59, 0x7664, 0x7aca, 0x7bc0, 0x7d76, 0x5360, 0x5cbe,
00978 0x5e97, 0x6f38, 0x70b9, 0x7c98, 0x9711, 0x9b8e, 0x9ede, 0x63a5,
00979 0x647a, 0x8776, 0x4e01, 0x4e95, 0x4ead, 0x505c, 0x5075, 0x5448,
00980 0x59c3, 0x5b9a, 0x5e40, 0x5ead, 0x5ef7, 0x5f81, 0x60c3, 0x633a,
00981 0x653f, 0x6574, 0x65cc, 0x6676, 0x6678, 0x67fe, 0x6968, 0x6a89,
00982 0x6b63, 0x6c40, 0x6dc0, 0x6de8, 0x6elf, 0x6e5e, 0x701e, 0x70a1,
00983 0x738e, 0x73fd, 0x753a, 0x775b, 0x7887, 0x798e, 0x7a0b, 0x7a7d,
00984 0x7cbe, 0x7d8e, 0x8247, 0x8a02, 0x8aea, 0x8c9e, 0x912d, 0x914a,
00985 0x91d8, 0x9266, 0x92cc, 0x9320, 0x9706, 0x9756,
00986 /* 0x70 */
00987 0x975c, 0x9802, 0x9f0e, 0x5236, 0x5291, 0x557c, 0x5824, 0x5e1d,
00988 0x5f1f, 0x608c, 0x63d0, 0x68af, 0x6fdf, 0x796d, 0x7b2c, 0x81cd,
```

```
00989 0x85ba, 0x88fd, 0x8af8, 0x8e44, 0x918d, 0x9664, 0x969b, 0x973d,
00990 0x984c, 0x9f4a, 0x4fce, 0x5146, 0x51cb, 0x52a9, 0x5632, 0x5f14,
00991 0x5f6b, 0x63aa, 0x64cd, 0x65e9, 0x6641, 0x66fa, 0x66f9, 0x671d,
00992 0x689d, 0x68d7, 0x69fd, 0x6f15, 0x6f6e, 0x7167, 0x71e5, 0x722a,
00993 0x74aa, 0x773a, 0x7956, 0x795a, 0x79df, 0x7a20, 0x7a95, 0x7c97,
00994 0x7cdf, 0x7d44, 0x7e70, 0x8087, 0x85fb, 0x86a4, 0x8a54, 0x8abf,
00995 0x8d99, 0x8e81, 0x9020, 0x906d, 0x91e3, 0x963b, 0x96d5, 0x9ce5,
00996 0x65cf, 0x7c07, 0x8db3, 0x93c3, 0x5b58, 0x5c0a, 0x5352, 0x62d9,
00997 0x731d, 0x5027, 0x5b97, 0x5f9e, 0x60b0, 0x616b, 0x68d5, 0x6dd9,
00998 0x742e, 0x7a2e, 0x7d42, 0x7d9c, 0x7e31, 0x816b,
00999 /* 0x71 */
01000 0x8e2a, 0x8e35, 0x937e, 0x9418, 0x4f50, 0x5750, 0x5de6, 0x5ea7,
01001 0x632b, 0x7f6a, 0x4e3b, 0x4f4f, 0x4f8f, 0x505a, 0x59dd, 0x80c4,
01002 0x546a, 0x5468, 0x55fe, 0x594f, 0x5b99, 0x5dde, 0x5eda, 0x665d,
01003 0x6731, 0x67f1, 0x682a, 0x6ce8, 0x6d32, 0x6e4a, 0x6f8d, 0x70b7,
01004 0x73e0, 0x7587, 0x7c4c, 0x7d02, 0x7d2c, 0x7da2, 0x821f, 0x86db,
01005 0x8a3b, 0x8a85, 0x8d70, 0x8e8a, 0x8f33, 0x9031, 0x914e, 0x9152,
01006 0x9444, 0x99d0, 0x7af9, 0x7ca5, 0x4fca, 0x5101, 0x51c6, 0x57c8,
01007 0x5bef, 0x5cfb, 0x6659, 0x6a3d, 0x6d5a, 0x6e96, 0x6fec, 0x710c,
01008 0x756f, 0x7ae3, 0x8822, 0x9021, 0x9075, 0x96cb, 0x99ff, 0x8301,
01009 0x4e2d, 0x4ef2, 0x8846, 0x91cd, 0x537d, 0x6adb, 0x696b, 0x6c41,
01010 0x847a, 0x589e, 0x618e, 0x66fe, 0x62ef, 0x70dd, 0x7511, 0x75c7,
01011 0x7e52, 0x84b8, 0x8b49, 0x8d08, 0x4e4b, 0x53ea,
01012 /* 0x72 */
01013 0x54ab, 0x5730, 0x5740, 0x5fd7, 0x6301, 0x6307, 0x646f, 0x652f,
01014 0x65e8, 0x667a, 0x679d, 0x67b3, 0x6b62, 0x6c60, 0x6c9a, 0x6f2c,
01015 0x77e5, 0x7825, 0x7949, 0x7957, 0x7d19, 0x80a2, 0x8102, 0x81f3,
01016 0x829d, 0x82b7, 0x8718, 0x8a8c, 0xf9fc, 0x8d04, 0x8dbe, 0x9072,
01017 0x76f4, 0x7a19, 0x7a37, 0x7e54, 0x8077, 0x5507, 0x55d4, 0x5875,
01018 0x632f, 0x6422, 0x6649, 0x664b, 0x686d, 0x699b, 0x6b84, 0x6d25,
01019 0x6eb1, 0x73cd, 0x7468, 0x74a1, 0x755b, 0x75b9, 0x76e1, 0x771e,
01020 0x778b, 0x79e6, 0x7e09, 0x7e1d, 0x81fb, 0x852f, 0x8897, 0x8a3a,
01021 0x8cd1, 0x8eeb, 0x8fb0, 0x9032, 0x93ad, 0x9663, 0x9673, 0x9707,
01022 0x4f84, 0x53f1, 0x59ea, 0x5ac9, 0x5e19, 0x684e, 0x74c6, 0x75be,
01023 0x79e9, 0x7a92, 0x81a3, 0x86ed, 0x8cea, 0x8dcc, 0x8fed, 0x659f,
01024 0x6715, 0xf9fd, 0x57f7, 0x6f57, 0x7ddd, 0x8f2f,
01025 /* 0x73 */
01026 0x93f6, 0x96c6, 0x5fb5, 0x61f2, 0x6f84, 0x4e14, 0x4f98, 0x501f,
01027 0x53c9, 0x55df, 0x5d6f, 0x5dee, 0x6b21, 0x6b64, 0x78cb, 0x7b9a,
01028 0xf9fe, 0x8e49, 0x8eca, 0x906e, 0x6349, 0x643e, 0x7740, 0x7a84,
01029 0x932f, 0x947f, 0x9f6a, 0x64b0, 0x6faf, 0x71e6, 0x74a8, 0x74da,
01030 0x7ac4, 0x7c12, 0x7e82, 0x7cb2, 0x7e98, 0x8b9a, 0x8d0a, 0x947d,
01031 0x9910, 0x994c, 0x5239, 0x5bdf, 0x64e6, 0x672d, 0x7d2e, 0x50ed,
01032 0x53c3, 0x5879, 0x6158, 0x6159, 0x61fa, 0x65ac, 0x7ad9, 0x8b92,
01033 0x8b96, 0x5009, 0x5021, 0x5275, 0x5531, 0x5a3c, 0x5ee0, 0x5f70,
01034 0x6134, 0x655e, 0x660c, 0x6636, 0x66a2, 0x69cd, 0x6ec4, 0x6f32,
01035 0x7316, 0x7621, 0x7a93, 0x8139, 0x8259, 0x83d6, 0x84bc, 0x50b5,
01036 0x57f0, 0x5bc0, 0x5be8, 0x5f69, 0x63a1, 0x7826, 0x7db5, 0x83dc,
01037 0x8521, 0x91c7, 0x91f5, 0x518a, 0x67f5, 0x7b56,
01038 /* 0x74 */
01039 0x8cac, 0x51c4, 0x59bb, 0x60bd, 0x8655, 0x501c, 0xf9ff, 0x5254,
01040 0x5c3a, 0x617d, 0x621a, 0x62d3, 0x64f2, 0x65a5, 0x6ecc, 0x7620,
01041 0x810a, 0x8e60, 0x965f, 0x96bb, 0x4edf, 0x5343, 0x5598, 0x5929,
01042 0x5ddd, 0x64c5, 0x6cc9, 0x6dfa, 0x7394, 0x7a7f, 0x821b, 0x85a6,
01043 0x8ce4, 0x8e10, 0x9077, 0x91e7, 0x95e1, 0x9621, 0x97c6, 0x51f8,
01044 0x54f2, 0x5586, 0x5fb9, 0x64a4, 0x6f88, 0x7db4, 0x8f1f, 0x8f4d,
01045 0x9435, 0x50c9, 0x5c16, 0x6cbe, 0x6dfb, 0x751b, 0x77bb, 0x7c3d,
01046 0x7c64, 0x8a79, 0x8ac2, 0x581e, 0x59be, 0x5e16, 0x6377, 0x7252,
01047 0x758a, 0x776b, 0x8adc, 0x8cbc, 0x8f12, 0x5ef3, 0x6674, 0x6df8,
01048 0x807d, 0x83c1, 0x8acb, 0x9751, 0x9bd6, 0xfa00, 0x5243, 0x66ff,
01049 0x6d95, 0x6eef, 0x7de0, 0x8ae6, 0x902e, 0x905e, 0x9ad4, 0x521d,
01050 0x527f, 0x54e8, 0x6194, 0x6284, 0x62db, 0x68a2,
01051 /* 0x75 */
01052 0x6912, 0x695a, 0x6a35, 0x7092, 0x7126, 0x785d, 0x7901, 0x790e,
01053 0x79d2, 0x7a0d, 0x8096, 0x8278, 0x82d5, 0x8349, 0x8549, 0x8c82,
01054 0x8d85, 0x9162, 0x918b, 0x91ae, 0x4fc3, 0x56d1, 0x71ed, 0x77d7,
01055 0x8700, 0x89f8, 0x5bf8, 0x5fd6, 0x6751, 0x90a8, 0x53e2, 0x585a,
01056 0x5bf5, 0x60a4, 0x6181, 0x6460, 0x7e3d, 0x8070, 0x8525, 0x9283,
01057 0x64ae, 0x50ac, 0x5d14, 0x6700, 0x589c, 0x62bd, 0x63a8, 0x690e,
01058 0x6978, 0x6a1e, 0x6e6b, 0x76ba, 0x79cb, 0x82bb, 0x8429, 0x8acf,
01059 0x8da8, 0x8ffd, 0x9112, 0x914b, 0x919c, 0x9310, 0x9318, 0x939a,
01060 0x96db, 0x9a36, 0x9c0d, 0x4e11, 0x755c, 0x795d, 0x7afa, 0x7b51,
01061 0x7bc9, 0x7e2e, 0x84c4, 0x8e59, 0x8e74, 0x8ef8, 0x9010, 0x6625,
01062 0x693f, 0x7443, 0x51fa, 0x672e, 0x9edc, 0x5145, 0x5fe0, 0x6c96,
01063 0x87f2, 0x885d, 0x8877, 0x60b4, 0x81b5, 0x8403,
01064 /* 0x76 */
01065 0x8d05, 0x53d6, 0x5439, 0x5634, 0x5a36, 0x5c31, 0x708a, 0x7fe0,
01066 0x805a, 0x8106, 0x81ed, 0x8da3, 0x9189, 0x9a5f, 0x9df2, 0x5074,
01067 0x4ec4, 0x53a0, 0x60fb, 0x6e2c, 0x5c64, 0x4f88, 0x5024, 0x55e4,
01068 0x5cd9, 0x5e5f, 0x6065, 0x6894, 0x6cbb, 0x6dc4, 0x71be, 0x75d4,
01069 0x75f4, 0x7661, 0x7a1a, 0x7a49, 0x7dc7, 0x7dfb, 0x7f6e, 0x81f4,
01070 0x86a9, 0x8f1c, 0x96c9, 0x99b3, 0x9f52, 0x5247, 0x52c5, 0x98ed,
01071 0x89aa, 0x4e03, 0x67d2, 0x6f06, 0x4fb5, 0x5be2, 0x6795, 0x6c88,
01072 0x6d78, 0x741b, 0x7827, 0x91dd, 0x937c, 0x87c4, 0x79e4, 0x7a31,
01073 0x5feb, 0x4ed6, 0x54a4, 0x553e, 0x58ae, 0x59a5, 0x60f0, 0x6253,
01074 0x62d6, 0x6736, 0x6955, 0x8235, 0x9640, 0x99b1, 0x99dd, 0x502c,
01075 0x5353, 0x5544, 0x577c, 0xfa01, 0x6258, 0xfa02, 0x64e2, 0x666b,
```



```
01076 0x67dd, 0x6fc1, 0x6fef, 0x7422, 0x7438, 0x8a17,
01077 /* 0x77 */
01078 0x9438, 0x5451, 0x5606, 0x5766, 0x5f48, 0x619a, 0x6b4e, 0x7058,
01079 0x70ad, 0x7dbb, 0x8a95, 0x596a, 0x812b, 0x63a2, 0x7708, 0x803d,
01080 0x8caa, 0x5854, 0x642d, 0x69bb, 0x5b95, 0x5e11, 0x6e6f, 0xfa03,
01081 0x8569, 0x514c, 0x53f0, 0x592a, 0x6020, 0x614b, 0x6b86, 0x6c70,
01082 0x6cf0, 0x7b1e, 0x80ce, 0x82d4, 0x8dc6, 0x90b0, 0x98b1, 0xfa04,
01083 0x64c7, 0x6fa4, 0x6491, 0x6504, 0x514e, 0x5410, 0x571f, 0x8a0e,
01084 0x615f, 0x6876, 0xfa05, 0x75db, 0x7b52, 0x7d71, 0x901a, 0x5806,
01085 0x69cc, 0x817f, 0x892a, 0x9000, 0x9839, 0x5078, 0x5957, 0x59ac,
01086 0x6295, 0x900f, 0x9b2a, 0x615d, 0x7279, 0x95d6, 0x5761, 0x5a46,
01087 0x5df4, 0x628a, 0x64ad, 0x64fa, 0x6777, 0x6ce2, 0x6d3e, 0x722c,
01088 0x7436, 0x7834, 0x7f77, 0x82ad, 0x8ddb, 0x9817, 0x5224, 0x5742,
01089 0x677f, 0x7248, 0x74e3, 0x8ca9, 0x8fa6, 0x9211,
01090 /* 0x78 */
01091 0x962a, 0x516b, 0x53ed, 0x634c, 0x4f69, 0x5504, 0x6096, 0x6557,
01092 0x6c9b, 0x6d7f, 0x724c, 0x72fd, 0x7a17, 0x8987, 0x8c9d, 0x5f6d,
01093 0x6f8e, 0x70f9, 0x81a8, 0x610e, 0x4fbf, 0x504f, 0x6241, 0x7247,
01094 0x7bc7, 0x7de8, 0x7fe9, 0x904d, 0x97ad, 0x9a19, 0x8cb6, 0x576a,
01095 0x5e73, 0x67b0, 0x840d, 0x8a55, 0x5420, 0x5b16, 0x5e63, 0x5ee2,
01096 0x5f0a, 0x6583, 0x80ba, 0x853d, 0x9589, 0x965b, 0x4f48, 0x5305,
01097 0x530d, 0x530f, 0x5486, 0x54fa, 0x5703, 0x5e03, 0x6016, 0x629b,
01098 0x62b1, 0x6355, 0xfa06, 0x6ce1, 0x6d66, 0x75b1, 0x7832, 0x80de,
01099 0x812f, 0x82de, 0x8461, 0x84b2, 0x888d, 0x8912, 0x900b, 0x92ea,
01100 0x98fd, 0x9b91, 0x5e45, 0x66b4, 0x66dd, 0x7011, 0x7206, 0xfa07,
01101 0x4ff5, 0x527d, 0x5f6a, 0x6153, 0x6753, 0x6a19, 0x6f02, 0x74e2,
01102 0x7968, 0x8868, 0x8c79, 0x98c7, 0x98c4, 0x9a43,
01103 /* 0x79 */
01104 0x54c1, 0x7a1f, 0x6953, 0x8af7, 0x8c4a, 0x98a8, 0x99ae, 0x5f7c,
01105 0x62ab, 0x75b2, 0x76ae, 0x88ab, 0x907f, 0x9642, 0x5339, 0x5f3c,
01106 0x5fc5, 0x6ccc, 0x73cc, 0x7562, 0x758b, 0x7b46, 0x82fe, 0x999d,
01107 0x4e4f, 0x903c, 0x4e0b, 0x4f55, 0x53a6, 0x59f0, 0x5ec8, 0x6630,
01108 0x6cb3, 0x7455, 0x8377, 0x8766, 0x8cc0, 0x9050, 0x971e, 0x9c15,
01109 0x58d1, 0x5b78, 0x8650, 0x8b14, 0x9db4, 0x5bd2, 0x6068, 0x608d,
01110 0x65f1, 0x6c57, 0x6f22, 0x6fa3, 0x701a, 0x7f55, 0x7ff0, 0x9591,
01111 0x9592, 0x9650, 0x97d3, 0x5272, 0x8f44, 0x51fd, 0x542b, 0x54b8,
01112 0x5563, 0x558a, 0x6abb, 0x6db5, 0x7dd8, 0x8266, 0x929c, 0x9677,
01113 0x9e79, 0x5408, 0x54c8, 0x76d2, 0x86e4, 0x95a4, 0x95d4, 0x965c,
01114 0x4ea2, 0x4f09, 0x59ee, 0x5ae6, 0x5df7, 0x6052, 0x6297, 0x676d,
01115 0x6841, 0x6c86, 0x6e2f, 0x7f38, 0x809b, 0x822a,
01116 /* 0x7a */
01117 0xfa08, 0xfa09, 0x9805, 0x4ea5, 0x5055, 0x54b3, 0x5793, 0x595a,
01118 0x5b69, 0x5bb3, 0x61c8, 0x6977, 0x6d77, 0x7023, 0x87f9, 0x89e3,
01119 0x8a72, 0x8ae7, 0x9082, 0x99ed, 0x9ab8, 0x52be, 0x6838, 0x5016,
01120 0x5e78, 0x674f, 0x8347, 0x884c, 0x4eab, 0x5411, 0x56ae, 0x736e,
01121 0x9115, 0x97ff, 0x9909, 0x9957, 0x9999, 0x5653, 0x589f, 0x865b,
01122 0x8a31, 0x61b2, 0x6af6, 0x737b, 0x8ed2, 0x6b47, 0x96aa, 0x9a57,
01123 0x5955, 0x7200, 0x8d6b, 0x9769, 0x4fd4, 0x5cf4, 0x5f26, 0x61f8,
01124 0x665b, 0x6ceb, 0x70ab, 0x7384, 0x73b9, 0x73fe, 0x7729, 0x774d,
01125 0x7d43, 0x7d62, 0x7e23, 0x8237, 0x8852, 0xfa0a, 0x8ce2, 0x9249,
01126 0x986f, 0x5b51, 0x7a74, 0x8840, 0x9801, 0x5acc, 0x4fe0, 0x5354,
01127 0x593e, 0x5cfd, 0x633e, 0x6d79, 0x72f9, 0x8105, 0x8107, 0x83a2,
01128 0x92cf, 0x9830, 0x4ea8, 0x5144, 0x5211, 0x578b,
01129 /* 0x7b */
01130 0x5f62, 0x6cc2, 0x6ece, 0x7005, 0x7050, 0x70af, 0x7192, 0x73e9,
01131 0x7469, 0x834a, 0x87a2, 0x8861, 0x9008, 0x90a2, 0x93a3, 0x99a8,
01132 0x516e, 0x5f57, 0x60e0, 0x6167, 0x66b3, 0x8559, 0x8e4a, 0x91af,
01133 0x978b, 0x4e4e, 0x4e92, 0x547c, 0x58d5, 0x58fa, 0x597d, 0x5cb5,
01134 0x5f27, 0x6236, 0x6248, 0x660a, 0x6667, 0x6beb, 0x6d69, 0x6dcf,
01135 0x6e56, 0x6ef8, 0x6f94, 0x6fe0, 0x705d, 0x72d0, 0x7425,
01136 0x745a, 0x74e0, 0x7693, 0x795c, 0x7cca, 0x7e1e, 0x80e1, 0x82a6,
01137 0x846b, 0x84bf, 0x864e, 0x865f, 0x8774, 0x8b77, 0x8c6a, 0x93ac,
01138 0x9800, 0x9865, 0x60d1, 0x6216, 0x9177, 0x5a5a, 0x660f, 0x6df7,
01139 0x6e3e, 0x743f, 0x9b42, 0x5fffd, 0x60da, 0x7b0f, 0x54c4, 0x5f18,
01140 0x6c5e, 0x6cd3, 0x6d2a, 0x70d8, 0x7d05, 0x8679, 0x8a0c, 0x9d3b,
01141 0x5316, 0x548c, 0x5b05, 0x6a3a, 0x706b, 0x7575,
01142 /* 0x7c */
01143 0x798d, 0x79be, 0x82b1, 0x83ef, 0x8a71, 0x8b41, 0x8ca8, 0x9774,
01144 0xfa0b, 0x64f4, 0x652b, 0x78ba, 0x78bb, 0x7a6b, 0x4e38, 0x559a,
01145 0x5950, 0x5ba6, 0x5e7b, 0x60a3, 0x63db, 0x6b61, 0x6665, 0x6853,
01146 0x6e19, 0x7165, 0x74b0, 0x7d08, 0x9084, 0x9a69, 0x9c25, 0x6d3b,
01147 0x6ed1, 0x733e, 0x8c41, 0x95ca, 0x51f0, 0x5e4c, 0x5fa8, 0x604d,
01148 0x60f6, 0x6130, 0x614c, 0x6643, 0x6644, 0x69a5, 0x6cc1, 0x6e5f,
01149 0x6ec9, 0x6f62, 0x714c, 0x749c, 0x7687, 0x7bc1, 0x7c27, 0x8352,
01150 0x8757, 0x9051, 0x968d, 0x9ec3, 0x532f, 0x56de, 0x5efb, 0x5f8a,
01151 0x6062, 0x6094, 0x61f7, 0x6666, 0x6703, 0x6a9c, 0x6dee, 0x6fae,
01152 0x7070, 0x736a, 0x7e6a, 0x81be, 0x8334, 0x86d4, 0x8aa8, 0x8cc4,
01153 0x5283, 0x7372, 0x5b96, 0x6a6b, 0x9404, 0x54ee, 0x5686, 0x5b5d,
01154 0x6548, 0x6585, 0x66c9, 0x689f, 0x6d8d, 0x6dc6,
01155 /* 0x7d */
01156 0x723b, 0x80b4, 0x9175, 0x9a4d, 0x4faf, 0x5019, 0x539a, 0x540e,
01157 0x543c, 0x5589, 0x55c5, 0x5e3f, 0x5f8c, 0x673d, 0x7166, 0x73dd,
01158 0x9005, 0x52db, 0x52f3, 0x5864, 0x58ce, 0x7104, 0x718f, 0x71fb,
01159 0x85b0, 0x8a13, 0x6688, 0x85a8, 0x55a7, 0x6684, 0x714a, 0x8431,
01160 0x5349, 0x5599, 0x6bc1, 0x5f59, 0x5fbd, 0x63ee, 0x6689, 0x7147,
01161 0x8af1, 0x8f1d, 0x9ebe, 0x4f11, 0x643a, 0x70cb, 0x7566, 0x8667,
01162 0x6064, 0x8b4e, 0x9df8, 0x5147, 0x51f6, 0x5308, 0x6d36, 0x80f8,
```

```
01163     0x9ed1, 0x6615, 0x6b23, 0x7098, 0x75d5, 0x5403, 0x5c79, 0x7d07,
01164     0x8a16, 0x6b20, 0x6b3d, 0x6b46, 0x5438, 0x6070, 0x6d3d, 0x7fd5,
01165     0x8208, 0x50d6, 0x51de, 0x559c, 0x566b, 0x56cd, 0x59ec, 0x5b09,
01166     0x5e0c, 0x6199, 0x6198, 0x6231, 0x665e, 0x66e6, 0x7199, 0x71b9,
01167     0x71ba, 0x72a7, 0x79a7, 0x7a00, 0x7fb2, 0x8a70,
01168 };
01169
01170 static int
01171 ksc5601_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
01172 {
01173     unsigned char c1 = (s[0] & 0x7F);
01174     if ((c1 >= 0x21 && c1 <= 0x2c) || (c1 >= 0x30 && c1 <= 0x48) || (c1 >= 0x4a && c1 <= 0x7d)) {
01175         if (n >= 2) {
01176             unsigned char c2 = (s[1] & 0x7F);
01177             if (c2 >= 0x21 && c2 < 0x7f) {
01178                 unsigned int i = 94 * (c1 - 0x21) + (c2 - 0x21);
01179                 unsigned short wc = 0xffff;
01180                 if (i < 1410) {
01181                     if (i < 1115)
01182                         wc = ksc5601_2uni_page21[i];
01183                     } else if (i < 3854) {
01184                         if (i < 3760)
01185                             wc = ksc5601_2uni_page30[i-1410];
01186                     } else {
01187                         if (i < 8742)
01188                             wc = ksc5601_2uni_page4a[i-3854];
01189                     }
01190                     if (wc != 0xffff) {
01191                         *pwc = (ucs4_t) wc;
01192                         return 2;
01193                     }
01194                 }
01195                 return RET_ILSEQ;
01196             }
01197             return RET_TOOFEW(0);
01198         }
01199         return RET_ILSEQ;
01200     }
01201 #endif /* NEED_TOWC */
01202
01203 #ifndef NEED_TOMB
01204 static const unsigned short ksc5601_2charset[8224] = {
01205     0x222e, 0x2234, 0x2157, 0x2127, 0x2823, 0x2129, 0x2146, 0x213e,
01206     0x2977, 0x2978, 0x2225, 0x2252, 0x2124, 0x222c, 0x2976, 0x282c,
01207     0x2879, 0x2876, 0x287a, 0x222f, 0x2821, 0x2822, 0x213f, 0x282a,
01208     0x282d, 0x292c, 0x2921, 0x2923, 0x2140, 0x292a, 0x292d, 0x2922,
01209     0x2824, 0x2924, 0x2925, 0x2826, 0x2926, 0x2927, 0x2828, 0x2928,
01210     0x2829, 0x2929, 0x2930, 0x282f, 0x292f, 0x282b, 0x292b, 0x282e,
01211     0x292e, 0x2227, 0x2230, 0x2228, 0x222b, 0x222a, 0x222d, 0x2229,
01212     0x2541, 0x2542, 0x2543, 0x2544, 0x2545, 0x2546, 0x2547, 0x2548,
01213     0x2549, 0x254a, 0x254b, 0x254c, 0x254d, 0x254e, 0x254f, 0x2550,
01214     0x2551, 0x2552, 0x2553, 0x2554, 0x2555, 0x2556, 0x2557, 0x2558,
01215     0x2561, 0x2562, 0x2563, 0x2564, 0x2565, 0x2566, 0x2567, 0x2568,
01216     0x2569, 0x256a, 0x256b, 0x256c, 0x256d, 0x256e, 0x256f, 0x2570,
01217     0x2571, 0x2572, 0x2573, 0x2574, 0x2575, 0x2576, 0x2577, 0x2578,
01218     0x2c27, 0x2c21, 0x2c22, 0x2c23, 0x2c24, 0x2c25, 0x2c26, 0x2c28,
01219     0x2c29, 0x2c2a, 0x2c2b, 0x2c2c, 0x2c2d, 0x2c2e, 0x2c2f, 0x2c30,
01220     0x2c31, 0x2c32, 0x2c33, 0x2c34, 0x2c35, 0x2c36, 0x2c37, 0x2c38,
01221     0x2c39, 0x2c3a, 0x2c3b, 0x2c3c, 0x2c3d, 0x2c3e, 0x2c3f, 0x2c40,
01222     0x2c41, 0x2c51, 0x2c52, 0x2c53, 0x2c54, 0x2c55, 0x2c56, 0x2c58,
01223     0x2c59, 0x2c5a, 0x2c5b, 0x2c5c, 0x2c5d, 0x2c5e, 0x2c5f, 0x2c60,
01224     0x2c61, 0x2c62, 0x2c63, 0x2c64, 0x2c65, 0x2c66, 0x2c67, 0x2c68,
01225     0x2c69, 0x2c6a, 0x2c6b, 0x2c6c, 0x2c6d, 0x2c6e, 0x2c6f, 0x2c70,
01226     0x2c71, 0x2c57, 0x212a, 0x212e, 0x212f, 0x2130, 0x2131, 0x2253,
01227     0x2254, 0x2125, 0x2126, 0x2236, 0x2147, 0x2148, 0x2158, 0x2979,
01228     0x297a, 0x297b, 0x297c, 0x297d, 0x297e, 0x2149, 0x2235, 0x2724,
01229     0x2260, 0x2265, 0x2262, 0x2759, 0x214a, 0x2877, 0x2878, 0x287b,
01230     0x287c, 0x287d, 0x287e, 0x2530, 0x2531, 0x2532, 0x2533, 0x2534,
01231     0x2535, 0x2536, 0x2537, 0x2538, 0x2539, 0x2521, 0x2522, 0x2523,
01232     0x2524, 0x2525, 0x2526, 0x2527, 0x2528, 0x2529, 0x252a, 0x2167,
01233     0x2168, 0x2166, 0x2169, 0x216a, 0x2255, 0x2256, 0x2257, 0x2259,
01234     0x2257, 0x2221, 0x2222, 0x2223, 0x2153, 0x2224, 0x2154, 0x2174,
01235     0x2175, 0x2233, 0x2232, 0x216e, 0x2170, 0x2144, 0x2150, 0x212b,
01236     0x217c, 0x217d, 0x217e, 0x217a, 0x217f, 0x2173, 0x2231, 0x2145,
01237     0x2171, 0x212d, 0x216f, 0x2156, 0x2141, 0x2155, 0x2142, 0x2143,
01238     0x216c, 0x216d, 0x2178, 0x2179, 0x2176, 0x2177, 0x2241, 0x2151,
01239     0x2152, 0x2867, 0x2868, 0x2869, 0x286a, 0x286b, 0x286c, 0x286d,
01240     0x286e, 0x286f, 0x2870, 0x2871, 0x2872, 0x2873, 0x2874, 0x2875,
01241     0x2967, 0x2968, 0x2969, 0x296a, 0x296b, 0x296c, 0x296d, 0x296e,
01242     0x296f, 0x2970, 0x2971, 0x2972, 0x2973, 0x2974, 0x2975, 0x297d,
01243     0x294e, 0x294f, 0x2950, 0x2951, 0x2952, 0x2953, 0x2954, 0x2955,
01244     0x2956, 0x2957, 0x2958, 0x2959, 0x295a, 0x295b, 0x295c, 0x295d,
01245     0x295e, 0x295f, 0x2960, 0x2961, 0x2962, 0x2963, 0x2964, 0x2965,
01246     0x2966, 0x284d, 0x284e, 0x284f, 0x2850, 0x2851, 0x2852, 0x2853,
01247     0x2854, 0x2855, 0x2856, 0x2857, 0x2858, 0x2859, 0x285a, 0x285b,
01248     0x285c, 0x285d, 0x285e, 0x285f, 0x2860, 0x2861, 0x2862, 0x2863,
01249     0x2864, 0x2865, 0x2866, 0x2621, 0x262c, 0x2622, 0x262d, 0x2623,
```



```
01250 0x2648, 0x2647, 0x262e, 0x2624, 0x2642, 0x2641, 0x262f, 0x2626,
01251 0x2646, 0x2645, 0x2631, 0x2625, 0x2644, 0x2643, 0x2630, 0x2627,
01252 0x263c, 0x2649, 0x264a, 0x2637, 0x264b, 0x264c, 0x2632, 0x2629,
01253 0x263e, 0x264d, 0x264e, 0x2639, 0x264f, 0x2650, 0x2634, 0x2628,
01254 0x2651, 0x2652, 0x2638, 0x263d, 0x2653, 0x2654, 0x2633, 0x262a,
01255 0x2655, 0x2656, 0x263a, 0x263f, 0x2657, 0x2658, 0x2635, 0x262b,
01256 0x2659, 0x265a, 0x263b, 0x265b, 0x265c, 0x2640, 0x265d, 0x265e,
01257 0x265f, 0x2660, 0x2661, 0x2662, 0x2663, 0x2664, 0x2636, 0x2246,
01258 0x2161, 0x2160, 0x2243, 0x2247, 0x2248, 0x224b, 0x224a, 0x2249,
01259 0x224c, 0x2163, 0x2162, 0x223a, 0x2239, 0x2165, 0x2164, 0x2238,
01260 0x2237, 0x215f, 0x215e, 0x2242, 0x215b, 0x215d, 0x215c, 0x2244,
01261 0x2245, 0x215a, 0x2159, 0x224f, 0x224e, 0x2250, 0x2251, 0x214f,
01262 0x214e, 0x223c, 0x223d, 0x2240, 0x223b, 0x223e, 0x223f, 0x224d,
01263 0x225b, 0x225c, 0x225d, 0x225a, 0x2121, 0x2122, 0x2123, 0x2128,
01264 0x2134, 0x2135, 0x2136, 0x2137, 0x2138, 0x2139, 0x213a, 0x213b,
01265 0x213c, 0x213d, 0x216b, 0x2132, 0x2133, 0x2a21, 0x2a22, 0x2a23,
01266 0x2a24, 0x2a25, 0x2a26, 0x2a27, 0x2a28, 0x2a29, 0x2a2a, 0x2a2b,
01267 0x2a2c, 0x2a2d, 0x2a2e, 0x2a2f, 0x2a30, 0x2a31, 0x2a32, 0x2a33,
01268 0x2a34, 0x2a35, 0x2a36, 0x2a37, 0x2a38, 0x2a39, 0x2a3a, 0x2a3b,
01269 0x2a3c, 0x2a3d, 0x2a3e, 0x2a3f, 0x2a40, 0x2a41, 0x2a42, 0x2a43,
01270 0x2a44, 0x2a45, 0x2a46, 0x2a47, 0x2a48, 0x2a49, 0x2a4a, 0x2a4b,
01271 0x2a4c, 0x2a4d, 0x2a4e, 0x2a4f, 0x2a50, 0x2a51, 0x2a52, 0x2a53,
01272 0x2a54, 0x2a55, 0x2a56, 0x2a57, 0x2a58, 0x2a59, 0x2a5a, 0x2a5b,
01273 0x2a5c, 0x2a5d, 0x2a5e, 0x2a5f, 0x2a60, 0x2a61, 0x2a62, 0x2a63,
01274 0x2a64, 0x2a65, 0x2a66, 0x2a67, 0x2a68, 0x2a69, 0x2a6a, 0x2a6b,
01275 0x2a6c, 0x2a6d, 0x2a6e, 0x2a6f, 0x2a70, 0x2a71, 0x2a72, 0x2a73,
01276 0x2b21, 0x2b22, 0x2b23, 0x2b24, 0x2b25, 0x2b26, 0x2b27, 0x2b28,
01277 0x2b29, 0x2b2a, 0x2b2b, 0x2b2c, 0x2b2d, 0x2b2e, 0x2b2f, 0x2b30,
01278 0x2b31, 0x2b32, 0x2b33, 0x2b34, 0x2b35, 0x2b36, 0x2b37, 0x2b38,
01279 0x2b39, 0x2b3a, 0x2b3b, 0x2b3c, 0x2b3d, 0x2b3e, 0x2b3f, 0x2b40,
01280 0x2b41, 0x2b42, 0x2b43, 0x2b44, 0x2b45, 0x2b46, 0x2b47, 0x2b48,
01281 0x2b49, 0x2b4a, 0x2b4b, 0x2b4c, 0x2b4d, 0x2b4e, 0x2b4f, 0x2b50,
01282 0x2b51, 0x2b52, 0x2b53, 0x2b54, 0x2b55, 0x2b56, 0x2b57, 0x2b58,
01283 0x2b59, 0x2b5a, 0x2b5b, 0x2b5c, 0x2b5d, 0x2b5e, 0x2b5f, 0x2b60,
01284 0x2b61, 0x2b62, 0x2b63, 0x2b64, 0x2b65, 0x2b66, 0x2b67, 0x2b68,
01285 0x2b69, 0x2b6a, 0x2b6b, 0x2b6c, 0x2b6d, 0x2b6e, 0x2b6f, 0x2b70,
01286 0x2b71, 0x2b72, 0x2b73, 0x2b74, 0x2b75, 0x2b76, 0x2421, 0x2422,
01287 0x2423, 0x2424, 0x2425, 0x2426, 0x2427, 0x2428, 0x2429, 0x242a,
01288 0x242b, 0x242c, 0x242d, 0x242e, 0x242f, 0x2430, 0x2431, 0x2432,
01289 0x2433, 0x2434, 0x2435, 0x2436, 0x2437, 0x2438, 0x2439, 0x243a,
01290 0x243b, 0x243c, 0x243d, 0x243e, 0x243f, 0x2440, 0x2441, 0x2442,
01291 0x2443, 0x2444, 0x2445, 0x2446, 0x2447, 0x2448, 0x2449, 0x244a,
01292 0x244b, 0x244c, 0x244d, 0x244e, 0x244f, 0x2450, 0x2451, 0x2452,
01293 0x2453, 0x2454, 0x2455, 0x2456, 0x2457, 0x2458, 0x2459, 0x245a,
01294 0x245b, 0x245c, 0x245d, 0x245e, 0x245f, 0x2460, 0x2461, 0x2462,
01295 0x2463, 0x2464, 0x2465, 0x2466, 0x2467, 0x2468, 0x2469, 0x246a,
01296 0x246b, 0x246c, 0x246d, 0x246e, 0x246f, 0x2470, 0x2471, 0x2472,
01297 0x2473, 0x2474, 0x2475, 0x2476, 0x2477, 0x2478, 0x2479, 0x247a,
01298 0x247b, 0x247c, 0x247d, 0x247e, 0x2931, 0x2932, 0x2933, 0x2934,
01299 0x2935, 0x2936, 0x2937, 0x2938, 0x2939, 0x293a, 0x293b, 0x293c,
01300 0x293d, 0x293e, 0x293f, 0x2940, 0x2941, 0x2942, 0x2943, 0x2944,
01301 0x2945, 0x2946, 0x2947, 0x2948, 0x2949, 0x294a, 0x294b, 0x294c,
01302 0x225f, 0x2831, 0x2832, 0x2833, 0x2834, 0x2835, 0x2836, 0x2837,
01303 0x2838, 0x2839, 0x283a, 0x283b, 0x283c, 0x283d, 0x283e, 0x283f,
01304 0x2840, 0x2841, 0x2842, 0x2843, 0x2844, 0x2845, 0x2846, 0x2847,
01305 0x2848, 0x2849, 0x284a, 0x284b, 0x284c, 0x225e, 0x2749, 0x274a,
01306 0x274b, 0x274c, 0x274d, 0x273a, 0x273b, 0x275c, 0x275d, 0x275e,
01307 0x2736, 0x2737, 0x2738, 0x2754, 0x2755, 0x2756, 0x2757, 0x2758,
01308 0x2721, 0x2722, 0x2723, 0x2725, 0x272b, 0x272c, 0x272d, 0x272e,
01309 0x272f, 0x2730, 0x2731, 0x2732, 0x2733, 0x2734, 0x2727, 0x2728,
01310 0x2729, 0x272a, 0x273d, 0x273e, 0x2765, 0x2766, 0x2767, 0x2768,
01311 0x2761, 0x2762, 0x2763, 0x273f, 0x2740, 0x2741, 0x2742, 0x2743,
01312 0x2744, 0x2745, 0x2746, 0x2747, 0x2748, 0x274e, 0x274f, 0x2750,
01313 0x2751, 0x2752, 0x2753, 0x275a, 0x275b, 0x2263, 0x276c, 0x2726,
01314 0x2760, 0x276f, 0x2261, 0x273c, 0x276d, 0x2735, 0x2739, 0x276a,
01315 0x276b, 0x275f, 0x2264, 0x2764, 0x276e, 0x2769, 0x6c69, 0x6f4b,
01316 0x7652, 0x5832, 0x6d5b, 0x5f32, 0x5f3e, 0x793b, 0x5c74, 0x7564,
01317 0x7326, 0x5d60, 0x6126, 0x4e78, 0x5c30, 0x632a, 0x7169, 0x4d7a,
01318 0x7c2f, 0x5321, 0x712b, 0x6751, 0x522c, 0x4e79, 0x717d, 0x5e3f,
01319 0x7b3a, 0x7939, 0x4e52, 0x632b, 0x6b60, 0x4e7a, 0x4b77, 0x6525,
01320 0x4a61, 0x544c, 0x6a61, 0x5c63, 0x5f2d, 0x4b6b, 0x552f, 0x5675,
01321 0x6578, 0x5e40, 0x6c23, 0x694d, 0x6a27, 0x6976, 0x7b3b, 0x6769,
01322 0x6f4c, 0x5066, 0x5e41, 0x642c, 0x584c, 0x7971, 0x4e5f, 0x7a24,
01323 0x6632, 0x7a7b, 0x7a3d, 0x4c48, 0x6f4d, 0x5555, 0x5322, 0x6c51,
01324 0x6427, 0x6c52, 0x7631, 0x4e7b, 0x5051, 0x4b3f, 0x6d24, 0x6d28,
01325 0x5e42, 0x7662, 0x6d5c, 0x5c75, 0x6039, 0x544e, 0x7435, 0x535b,
01326 0x5635, 0x6c24, 0x6466, 0x716a, 0x4b6c, 0x4b40, 0x6c72, 0x506a,
01327 0x7972, 0x6c25, 0x505f, 0x676a, 0x506b, 0x5c51, 0x5b69, 0x7d4c,
01328 0x5b57, 0x5a61, 0x5636, 0x635f, 0x5e43, 0x5e44, 0x4a21, 0x6e6c,
01329 0x5323, 0x6e37, 0x784f, 0x6a48, 0x6e38, 0x712c, 0x7125, 0x694e,
01330 0x793c, 0x6579, 0x6c6a, 0x5d56, 0x6d42, 0x7825, 0x653a, 0x5b58,
01331 0x4a22, 0x514d, 0x6e6d, 0x6c6b, 0x5e45, 0x6360, 0x4a49, 0x7269,
01332 0x554e, 0x7636, 0x4e42, 0x5647, 0x6334, 0x712d, 0x6a62, 0x5742,
01333 0x7327, 0x4d6a, 0x6b6e, 0x5932, 0x7d25, 0x7655, 0x5562, 0x7835,
01334 0x4c75, 0x7535, 0x642d, 0x676b, 0x7155, 0x703b, 0x6935, 0x4c49,
01335 0x7a55, 0x6154, 0x5756, 0x5c41, 0x5e46, 0x7a6f, 0x6361, 0x6173,
01336 0x5c76, 0x4e7c, 0x5b44, 0x7871, 0x5c64, 0x656f, 0x5c31, 0x5556,
```

```

01337 0x735a, 0x4b41, 0x5b43, 0x597a, 0x536e, 0x7a38, 0x7d26, 0x6b6f,
01338 0x7426, 0x4c4a, 0x7328, 0x735b, 0x5b27, 0x7637, 0x4f66, 0x7072,
01339 0x4b5a, 0x6752, 0x5743, 0x7670, 0x685e, 0x6526, 0x6567, 0x4a23,
01340 0x4c27, 0x6a49, 0x7836, 0x7a25, 0x712e, 0x6f4e, 0x4b6d, 0x7630,
01341 0x6f4f, 0x694f, 0x775e, 0x4e53, 0x5c77, 0x5b28, 0x4b78, 0x5f21,
01342 0x5d61, 0x754a, 0x6936, 0x676c, 0x6e6e, 0x7370, 0x5f3f, 0x4c4b,
01343 0x5041, 0x7452, 0x603a, 0x5f40, 0x4e60, 0x5c52, 0x7d6a, 0x5676,
01344 0x6a4a, 0x6869, 0x632c, 0x7350, 0x4a24, 0x5b78, 0x5e47, 0x6b70,
01345 0x7156, 0x6562, 0x4c4c, 0x4b7b, 0x6a63, 0x5f41, 0x566d, 0x6950,
01346 0x6e39, 0x5563, 0x5153, 0x6570, 0x6834, 0x6b43, 0x6a2a, 0x7a7c,
01347 0x7576, 0x703c, 0x7d54, 0x603b, 0x4e43, 0x503a, 0x773a, 0x5873,
01348 0x774d, 0x642e, 0x545f, 0x5067, 0x6c7d, 0x522e, 0x6e6f, 0x5557,
01349 0x6a64, 0x7822, 0x4d6b, 0x573f, 0x7b31, 0x4d6c, 0x5c32, 0x506c,
01350 0x4e7d, 0x6e70, 0x4c42, 0x506d, 0x6577, 0x737c, 0x6e22, 0x5933,
01351 0x5874, 0x6937, 0x4e2e, 0x5922, 0x5871, 0x544f, 0x6527, 0x5552,
01352 0x5629, 0x7422, 0x7157, 0x5558, 0x703d, 0x5750, 0x5450, 0x574f,
01353 0x6b6a, 0x7d6b, 0x5b6d, 0x7c45, 0x4b42, 0x7d55, 0x7448, 0x686a,
01354 0x7573, 0x795e, 0x536f, 0x6c53, 0x5d42, 0x6f37, 0x6754, 0x4a4a,
01355 0x597b, 0x7a7d, 0x562a, 0x7478, 0x7777, 0x5c2c, 0x5757, 0x5f22,
01356 0x4e3e, 0x5370, 0x7024, 0x616c, 0x4f67, 0x734b, 0x6d29, 0x4a3e,
01357 0x746f, 0x764e, 0x5e7b, 0x503b, 0x5537, 0x6e71, 0x7428, 0x5c78,
01358 0x4b27, 0x5a4e, 0x6066, 0x6d25, 0x6e72, 0x5c79, 0x795c, 0x735c,
01359 0x7872, 0x7479, 0x7c71, 0x503c, 0x5b79, 0x5731, 0x4b7c, 0x7025,
01360 0x4b7d, 0x5574, 0x4d6d, 0x4a25, 0x562b, 0x5042, 0x703e, 0x523d,
01361 0x4c24, 0x7a36, 0x4c4d, 0x5a7a, 0x764f, 0x6938, 0x5875, 0x4c4e,
01362 0x574d, 0x5451, 0x696d, 0x4a6b, 0x5962, 0x7d32, 0x632d, 0x564c,
01363 0x5934, 0x6127, 0x6e53, 0x5043, 0x7d33, 0x5564, 0x4f68, 0x6d43,
01364 0x5032, 0x4e7e, 0x5a28, 0x7850, 0x7d56, 0x7851, 0x7852, 0x5c53,
01365 0x5d62, 0x7b79, 0x5d41, 0x6335, 0x6d5d, 0x4e44, 0x4b21, 0x5d63,
01366 0x7c5d, 0x792f, 0x527b, 0x4f21, 0x6428, 0x7436, 0x6c7e, 0x632e,
01367 0x676d, 0x7d41, 0x5a62, 0x5833, 0x5d64, 0x706f, 0x7671, 0x7a70,
01368 0x5175, 0x5a4f, 0x5c54, 0x5c26, 0x6f3f, 0x4e4f, 0x6059, 0x5956,
01369 0x6c54, 0x6a4b, 0x4a3f, 0x5530, 0x4f69, 0x716d, 0x4c4f, 0x6478,
01370 0x646d, 0x5758, 0x7d27, 0x6a2b, 0x7632, 0x4f70, 0x793d, 0x6674,
01371 0x4b5b, 0x7351, 0x6951, 0x7329, 0x5060, 0x6952, 0x5a63, 0x6252,
01372 0x7622, 0x6174, 0x5a64, 0x6755, 0x753f, 0x4f22, 0x4d2f, 0x4f23,
01373 0x4d30, 0x717e, 0x5023, 0x612f, 0x7823, 0x4a26, 0x773b, 0x726a,
01374 0x5e48, 0x6953, 0x5e49, 0x7d5e, 0x4a40, 0x796a, 0x514e, 0x6e54,
01375 0x5452, 0x5923, 0x7d28, 0x5a7e, 0x774e, 0x7a3e, 0x4f56, 0x5770,
01376 0x6b61, 0x7845, 0x5c7a, 0x5d43, 0x795f, 0x676f, 0x7d65, 0x7623,
01377 0x597c, 0x7d29, 0x676e, 0x5565, 0x6f50, 0x4d31, 0x7722, 0x7132,
01378 0x7131, 0x4d32, 0x5a2b, 0x4a27, 0x6362, 0x7b3c, 0x5924, 0x6e3a,
01379 0x7853, 0x7b7a, 0x4f24, 0x5c7b, 0x7663, 0x6d2a, 0x7221, 0x4e61,
01380 0x7a26, 0x7960, 0x6c56, 0x646e, 0x7921, 0x7b6f, 0x796b, 0x6e23,
01381 0x6a2c, 0x4a28, 0x747a, 0x4d56, 0x7c76, 0x7449, 0x7854, 0x7826,
01382 0x5e4a, 0x7246, 0x575a, 0x5350, 0x5845, 0x6a66, 0x735d, 0x645a,
01383 0x7664, 0x7672, 0x5f42, 0x597d, 0x4c76, 0x533a, 0x642f, 0x7961,
01384 0x7026, 0x4b53, 0x603c, 0x744a, 0x547a, 0x7d2a, 0x7962, 0x7437,
01385 0x7d42, 0x7c30, 0x7d6c, 0x4a62, 0x7d3d, 0x6a67, 0x5f43, 0x5152,
01386 0x4e62, 0x5324, 0x7d2b, 0x5f60, 0x7247, 0x6770, 0x506e, 0x732a,
01387 0x5e4b, 0x7638, 0x6175, 0x7133, 0x7723, 0x4a29, 0x4f25, 0x5f44,
01388 0x6130, 0x703f, 0x7624, 0x6336, 0x7a46, 0x506f, 0x7d6d, 0x5d44,
01389 0x7c77, 0x663f, 0x5e2d, 0x7a3f, 0x6571, 0x6d44, 0x5225, 0x7d6e,
01390 0x7536, 0x6176, 0x5e4c, 0x7c5e, 0x6c57, 0x4d5d, 0x5637, 0x4d33,
01391 0x7855, 0x6558, 0x4f6a, 0x4f50, 0x6a4c, 0x6a2e, 0x6a2d, 0x5371,
01392 0x5325, 0x774f, 0x6e24, 0x5024, 0x7222, 0x5070, 0x7223, 0x7778,
01393 0x5033, 0x5b29, 0x533b, 0x4a6c, 0x7126, 0x4b55, 0x7767, 0x4d5e,
01394 0x7724, 0x7840, 0x535d, 0x4c50, 0x4f26, 0x7673, 0x6177, 0x535c,
01395 0x7a7e, 0x7a27, 0x6b59, 0x4f27, 0x6a2f, 0x646f, 0x6939, 0x7158,
01396 0x5858, 0x6072, 0x6634, 0x5c7c, 0x7371, 0x6350, 0x727b, 0x5b46,
01397 0x5071, 0x5072, 0x4f5c, 0x5351, 0x4c31, 0x7758, 0x4b28, 0x6b3c,
01398 0x643e, 0x745c, 0x5c42, 0x7027, 0x6640, 0x4a6d, 0x686b, 0x6568,
01399 0x5c43, 0x6d5e, 0x5372, 0x4c77, 0x4e54, 0x672b, 0x4b43, 0x6131,
01400 0x7732, 0x5373, 0x5352, 0x7540, 0x5f5d, 0x6e73, 0x6771, 0x7d34,
01401 0x7248, 0x7352, 0x6e74, 0x6253, 0x4c51, 0x5f6a, 0x693a, 0x5957,
01402 0x754d, 0x7172, 0x7a47, 0x5978, 0x5442, 0x7665, 0x5d45, 0x6772,
01403 0x6d5f, 0x4a4b, 0x5b7a, 0x6835, 0x5326, 0x7d35, 0x7949, 0x6462,
01404 0x7b3d, 0x5724, 0x4e45, 0x4e55, 0x5666, 0x653d, 0x5e4d, 0x6c73,
01405 0x6d60, 0x6c6c, 0x7b3e, 0x5f6b, 0x6178, 0x793e, 0x5073, 0x602a,
01406 0x6862, 0x6254, 0x527d, 0x6528, 0x5953, 0x535e, 0x7438, 0x773c,
01407 0x5c7d, 0x686c, 0x6467, 0x6377, 0x6c28, 0x7a71, 0x6572, 0x5074,
01408 0x522f, 0x5c65, 0x5025, 0x7134, 0x7c31, 0x4c78, 0x5d46, 0x7a51,
01409 0x775f, 0x7a28, 0x6e75, 0x5e4e, 0x6773, 0x772c, 0x6b44, 0x6d61,
01410 0x602b, 0x5d47, 0x5233, 0x523f, 0x4a4c, 0x7b3f, 0x657d, 0x5d65,
01411 0x584d, 0x6c74, 0x5075, 0x686d, 0x5052, 0x5958, 0x7666, 0x5b2a,
01412 0x7760, 0x5859, 0x7423, 0x745d, 0x6f51, 0x5935, 0x6d2b, 0x6337,
01413 0x6e3b, 0x4d34, 0x6073, 0x6a4d, 0x6c75, 0x686e, 0x4b29, 0x712f,
01414 0x4a4d, 0x6c29, 0x726b, 0x7d6f, 0x7973, 0x6641, 0x6c58, 0x6d2c,
01415 0x6a4e, 0x685f, 0x5e4f, 0x5226, 0x6774, 0x5156, 0x6642, 0x6363,
01416 0x6430, 0x5834, 0x7625, 0x735e, 0x7768, 0x6846, 0x7b66,
01417 0x5d66, 0x5c7e, 0x585a, 0x5a2c, 0x6a30, 0x6338, 0x4a2a, 0x6179,
01418 0x6a31, 0x726c, 0x7a6e, 0x6e55, 0x7974, 0x526c, 0x7b7b, 0x7d70,
01419 0x603d, 0x4e63, 0x7846, 0x5e2e, 0x5f45, 0x653e, 0x6d2d, 0x7a6a,
01420 0x4d6e, 0x6d26, 0x6d2e, 0x706d, 0x5d21, 0x6d2f, 0x7c78, 0x586b,
01421 0x4c79, 0x4d35, 0x7a29, 0x615d, 0x6255, 0x6d4f, 0x5d22, 0x794a,
01422 0x6a68, 0x656d, 0x536b, 0x6954, 0x617a, 0x644c, 0x6164, 0x6847,
01423 0x4e5b, 0x5c55, 0x7735, 0x7c73, 0x7073, 0x4e2f, 0x7135, 0x6f52,

```

```
01424 0x6848, 0x6b71, 0x4b54, 0x603e, 0x6378, 0x6a69, 0x7c32, 0x6074,
01425 0x4f60, 0x6e25, 0x7a2a, 0x6643, 0x6132, 0x4a2b, 0x6364, 0x693b,
01426 0x6256, 0x7372, 0x6e56, 0x6a32, 0x5076, 0x6c59, 0x5a4b, 0x4f28,
01427 0x5d23, 0x585b, 0x794e, 0x6955, 0x6351, 0x523c, 0x582c, 0x734c,
01428 0x4d7b, 0x7656, 0x6775, 0x686f, 0x6379, 0x523b, 0x7373, 0x637b,
01429 0x5e50, 0x4e30, 0x5677, 0x7159, 0x7541, 0x5c44, 0x753b, 0x5e51,
01430 0x5c66, 0x5e52, 0x6d62, 0x6e76, 0x6a4f, 0x706e, 0x637c, 0x535f,
01431 0x5374, 0x6133, 0x6134, 0x7453, 0x5f46, 0x6956, 0x5b2b, 0x7626,
01432 0x6339, 0x6b45, 0x7429, 0x4d36, 0x5279, 0x5a2d, 0x5263, 0x4f51,
01433 0x4b5c, 0x4c7a, 0x4f5d, 0x6829, 0x633b, 0x633a, 0x605a, 0x6e77,
01434 0x5c33, 0x5375, 0x5726, 0x7635, 0x575b, 0x6155, 0x546a, 0x5f23,
01435 0x7d5f, 0x5077, 0x6d54, 0x4b2a, 0x645b, 0x617b, 0x4b22, 0x5360,
01436 0x643f, 0x7b40, 0x5a3e, 0x644d, 0x5639, 0x6f40, 0x617c, 0x7639,
01437 0x5f47, 0x6431, 0x5c67, 0x5c68, 0x7a56, 0x5376, 0x715a, 0x7a72,
01438 0x627d, 0x554f, 0x5078, 0x4d5f, 0x754b, 0x6470, 0x4b2b, 0x5744,
01439 0x627e, 0x5d5a, 0x5a2e, 0x4a6e, 0x5539, 0x6321, 0x6863, 0x732b,
01440 0x4f29, 0x5377, 0x5471, 0x4e64, 0x6872, 0x6575, 0x672e, 0x563a,
01441 0x5f6c, 0x6440, 0x6864, 0x5835, 0x645c, 0x7439, 0x7136, 0x625e,
01442 0x6135, 0x4d6f, 0x7127, 0x4e65, 0x4b5d, 0x5963, 0x732c, 0x5079,
01443 0x6c2b, 0x5e53, 0x7769, 0x7975, 0x615e, 0x4b6e, 0x633c, 0x7856,
01444 0x5b6e, 0x7d71, 0x7736, 0x745e, 0x726d, 0x5b59, 0x7028, 0x617d,
01445 0x5e54, 0x602c, 0x6d63, 0x5361, 0x5f48, 0x5936, 0x7d2c, 0x6f53,
01446 0x6441, 0x786b, 0x5b2c, 0x7c46, 0x582d, 0x763a, 0x5b5f, 0x5353,
01447 0x7847, 0x4a4e, 0x7841, 0x5234, 0x5c34, 0x7a39, 0x4a4f, 0x7c33,
01448 0x6a6a, 0x6a6b, 0x507a, 0x6d64, 0x5d67, 0x5f49, 0x5f6d, 0x6e3c,
01449 0x6f41, 0x4c52, 0x5d24, 0x5f4a, 0x5378, 0x7128, 0x4d37, 0x6f54,
01450 0x645d, 0x5f6e, 0x4b2c, 0x693c, 0x6a6c, 0x5f4b, 0x793f, 0x562f,
01451 0x5546, 0x4f2a, 0x4e29, 0x5678, 0x7137, 0x6e78, 0x5959, 0x735f,
01452 0x7848, 0x4e46, 0x5566, 0x7466, 0x6645, 0x6f55, 0x4b6f, 0x7c5f,
01453 0x5c27, 0x5667, 0x7849, 0x6352, 0x633d, 0x4f61, 0x7040, 0x6c5a,
01454 0x5d57, 0x7b70, 0x6c2c, 0x7029, 0x7a57, 0x7b41, 0x5240, 0x6530,
01455 0x6d65, 0x4b2d, 0x7930, 0x7725, 0x4b2e, 0x5a2f, 0x5836, 0x5327,
01456 0x7b32, 0x7d44, 0x6c2d, 0x7b21, 0x6569, 0x696e, 0x7374, 0x7873,
01457 0x7041, 0x5e2f, 0x7830, 0x7360, 0x672f, 0x5b2d, 0x6635, 0x7928,
01458 0x5d58, 0x6859, 0x6f56, 0x5362, 0x625f, 0x7c60, 0x5748, 0x7d2d,
01459 0x5f6f, 0x4c53, 0x5379, 0x5470, 0x5b47, 0x5e55, 0x7074, 0x5550,
01460 0x6559, 0x7c47, 0x5c56, 0x6260, 0x5a30, 0x7323, 0x536c, 0x744b,
01461 0x7d45, 0x637d, 0x7931, 0x507b, 0x6c5b, 0x753c, 0x7224, 0x584e,
01462 0x584f, 0x7577, 0x7661, 0x5237, 0x7b6c, 0x5d48, 0x6468, 0x5241,
01463 0x7857, 0x563b, 0x5e56, 0x773d, 0x6c2e, 0x5061, 0x6075, 0x6a33,
01464 0x4e56, 0x4c25, 0x6c76, 0x6261, 0x633e, 0x7c48, 0x4d70, 0x7976,
01465 0x5f70, 0x653f, 0x4e3f, 0x7c61, 0x6d30, 0x7d51, 0x763b, 0x794f,
01466 0x6b5a, 0x4a41, 0x5238, 0x4d71, 0x6353, 0x7d66, 0x666d, 0x637a,
01467 0x702a, 0x7950, 0x7c62, 0x7827, 0x6165, 0x6e79, 0x6776, 0x6a6d,
01468 0x7c34, 0x7542, 0x575c, 0x7075, 0x5d68, 0x536d, 0x757c, 0x5a3f,
01469 0x4c7b, 0x537a, 0x7424, 0x6f57, 0x5443, 0x7b63, 0x7b6d, 0x602d,
01470 0x6a6e, 0x7b33, 0x6442, 0x7667, 0x525d, 0x5f4c, 0x7c49, 0x6529,
01471 0x6076, 0x7633, 0x617e, 0x4b70, 0x6a6f, 0x6a70, 0x5a40, 0x7834,
01472 0x6b72, 0x6443, 0x6957, 0x6471, 0x4a6f, 0x4e57, 0x7c4a, 0x7361,
01473 0x4b44, 0x6365, 0x4b45, 0x6a34, 0x693d, 0x5749, 0x6b5b, 0x6d31,
01474 0x4c43, 0x773e, 0x7c4b, 0x7874, 0x5937, 0x7353, 0x7354, 0x7764,
01475 0x7751, 0x5837, 0x4e31, 0x4a42, 0x7b34, 0x4b46, 0x7076, 0x5567,
01476 0x6a50, 0x4c54, 0x4b2f, 0x742a, 0x692f, 0x7543, 0x6958, 0x5d69,
01477 0x7173, 0x557b, 0x5e3b, 0x747b, 0x7d72, 0x7d72, 0x7726, 0x5d49,
01478 0x5453, 0x4c28, 0x5a41, 0x4c55, 0x5964, 0x7a4a, 0x6563, 0x533c,
01479 0x4a70, 0x5044, 0x4a50, 0x7a2b, 0x6b6b, 0x6778, 0x5965, 0x5157,
01480 0x7324, 0x547b, 0x7c63, 0x7a58, 0x7355, 0x4f2b, 0x6b73, 0x557c,
01481 0x5354, 0x4d7c, 0x5966, 0x6279, 0x6221, 0x6b54, 0x6077, 0x6432,
01482 0x4c7c, 0x7b64, 0x742b, 0x503d, 0x4a71, 0x6f38, 0x5740, 0x6e7a,
01483 0x7d74, 0x5363, 0x7b42, 0x5568, 0x5b2e, 0x6136, 0x7837, 0x603f,
01484 0x7b43, 0x5d6a, 0x6222, 0x6e26, 0x7668, 0x7675, 0x5d4a, 0x5062,
01485 0x5d26, 0x5d6b, 0x6479, 0x632f, 0x507c, 0x747c, 0x4c3c, 0x776a,
01486 0x6564, 0x5f71, 0x7761, 0x7977, 0x6f39, 0x7858, 0x7929, 0x7859,
01487 0x6e3d, 0x5846, 0x6463, 0x754e, 0x5d59, 0x5967, 0x5239, 0x5543,
01488 0x5a65, 0x5a50, 0x5159, 0x4e58, 0x4b5e, 0x742c, 0x5a7b, 0x7669,
01489 0x6873, 0x4f2c, 0x7070, 0x747d, 0x5b48, 0x4e40, 0x6354, 0x514f,
01490 0x7175, 0x4d72, 0x4f6b, 0x4d38, 0x6326, 0x515a, 0x7225, 0x7226,
01491 0x644e, 0x537b, 0x7129, 0x7249, 0x6f58, 0x6649, 0x5838, 0x7a73,
01492 0x7335, 0x7824, 0x5173, 0x6648, 0x785a, 0x5c69, 0x5e57, 0x4b5f,
01493 0x4f6c, 0x745f, 0x5174, 0x523a, 0x5f72, 0x6137, 0x6223, 0x537c,
01494 0x6d66, 0x5b49, 0x647a, 0x4f5e, 0x4e50, 0x5553, 0x7375, 0x772e,
01495 0x6f48, 0x4d73, 0x754f, 0x6573, 0x7042, 0x4a51, 0x6a71, 0x5026,
01496 0x595a, 0x702b, 0x6b67, 0x6540, 0x7c35, 0x6444, 0x4c29, 0x7d46,
01497 0x6a35, 0x652a, 0x5f3a, 0x615f, 0x5a51, 0x6138, 0x6874, 0x537d,
01498 0x6224, 0x724a, 0x5a66, 0x7733, 0x7d4d, 0x7336, 0x6e57, 0x7544,
01499 0x5824, 0x7227, 0x5938, 0x5939, 0x6f49, 0x564e, 0x774b, 0x5f2e,
01500 0x6875, 0x5235, 0x5355, 0x744c, 0x5a7c, 0x5968, 0x776b, 0x7549,
01501 0x733c, 0x5a52, 0x5335, 0x6836, 0x564f, 0x743a, 0x7749, 0x4c2a,
01502 0x7043, 0x4c56, 0x5053, 0x533d, 0x5b7b, 0x4b60, 0x5364, 0x7677,
01503 0x553a, 0x734d, 0x4b61, 0x6b74, 0x742d, 0x7c2a, 0x776c, 0x6876,
01504 0x5a67, 0x774c, 0x6541, 0x606e, 0x557d, 0x4e66, 0x7c2b, 0x553b,
01505 0x7228, 0x6225, 0x4d39, 0x6a72, 0x4b47, 0x4d74, 0x5b2f, 0x6f59,
01506 0x4d3a, 0x7c79, 0x5f73, 0x4e67, 0x5a42, 0x4f2d, 0x6779, 0x7828,
01507 0x7362, 0x4a72, 0x5f24, 0x5444, 0x4c57, 0x6542, 0x4d3b, 0x6f5a,
01508 0x6e58, 0x5d27, 0x6226, 0x6040, 0x5630, 0x784a, 0x7c7a, 0x597e,
01509 0x5e30, 0x5d6c, 0x5a68, 0x5460, 0x5679, 0x4d57, 0x5e58, 0x7278,
01510 0x6456, 0x5045, 0x742e, 0x5d28, 0x6d45, 0x7356, 0x5e59, 0x6366,
```

```
01511 0x5328, 0x5b30, 0x655a, 0x633f, 0x5b31, 0x5569, 0x6041, 0x6f5b,
01512 0x7069, 0x5732, 0x507d, 0x5969, 0x507e, 0x6c6d, 0x5329, 0x7229,
01513 0x7044, 0x6262, 0x696f, 0x7951, 0x6959, 0x685a, 0x5a43, 0x5a44,
01514 0x5445, 0x677a, 0x4d60, 0x6330, 0x5b32, 0x7b44, 0x7363, 0x5925,
01515 0x7b67, 0x5d4b, 0x5054, 0x6636, 0x602e, 0x7d5a, 0x5c35, 0x6078,
01516 0x6731, 0x7570, 0x585c, 0x6d46, 0x6139, 0x6340, 0x7940, 0x6970,
01517 0x595b, 0x7364, 0x5c36, 0x6469, 0x7045, 0x6341, 0x7c4c, 0x7c4d,
01518 0x724b, 0x724c, 0x644f, 0x715b, 0x7a59, 0x7138, 0x7d75, 0x6079,
01519 0x677b, 0x7c37, 0x7c64, 0x7b45, 0x6367, 0x5839, 0x7678, 0x5c45,
01520 0x4c58, 0x602f, 0x7467, 0x6f5c, 0x4f7c, 0x6f5d, 0x722a, 0x7d3e,
01521 0x4a2c, 0x7d3b, 0x7d47, 0x6732, 0x6a51, 0x5f74, 0x516c, 0x645e,
01522 0x6543, 0x5926, 0x4d3c, 0x7365, 0x6d55, 0x593a, 0x6d67, 0x7b35,
01523 0x786c, 0x6067, 0x4c59, 0x5446, 0x6725, 0x5575, 0x533e, 0x7c7b,
01524 0x6472, 0x5f75, 0x6878, 0x786d, 0x4e47, 0x7d76, 0x6858, 0x4d58,
01525 0x6756, 0x4c5a, 0x4a63, 0x5f76, 0x7047, 0x7046, 0x583a, 0x7174,
01526 0x7470, 0x754c, 0x7c65, 0x6a45, 0x6a73, 0x5d5b, 0x5c57, 0x5e7d,
01527 0x7279, 0x5547, 0x5850, 0x7048, 0x5121, 0x5122, 0x5954, 0x5668,
01528 0x594a, 0x5a31, 0x5847, 0x5c62, 0x734e, 0x7574, 0x7139, 0x5a53,
01529 0x766a, 0x4f75, 0x7d2e, 0x4a52, 0x5f34, 0x575d, 0x7a3a, 0x6e27,
01530 0x753d, 0x7875, 0x6d68, 0x5461, 0x5123, 0x6156, 0x7978, 0x5b4a,
01531 0x4b79, 0x5454, 0x595c, 0x6e3e, 0x776d, 0x526e, 0x6166, 0x7779,
01532 0x5d6d, 0x685b, 0x5b33, 0x5177, 0x6030, 0x5462, 0x7657, 0x5779,
01533 0x585d, 0x4d7d, 0x722b, 0x4d3d, 0x7842, 0x722c, 0x4a2d, 0x4a2e,
01534 0x4f2e, 0x6342, 0x5c37, 0x5b5a, 0x593b, 0x4a73, 0x7653, 0x6678,
01535 0x6a75, 0x6a76, 0x7679, 0x4f2f, 0x4a53, 0x4a2f, 0x5230, 0x713a,
01536 0x5733, 0x6343, 0x737d, 0x5e5a, 0x5e5b, 0x6f5e, 0x6263, 0x6e7b,
01537 0x5f77, 0x574a, 0x4e68, 0x5b5b, 0x713b, 0x6971, 0x7a37, 0x5046,
01538 0x4c2b, 0x6e28, 0x4b7a, 0x7979, 0x4c7d, 0x537e, 0x6450, 0x726e,
01539 0x5455, 0x5f4d, 0x7c38, 0x5150, 0x724d, 0x7752, 0x4a54, 0x5559,
01540 0x585e, 0x4d59, 0x6e29, 0x763c, 0x4c5b, 0x7049, 0x7c7c, 0x6849,
01541 0x747e, 0x677c, 0x575e, 0x5e5c, 0x702c, 0x4c7e, 0x4d61, 0x613a,
01542 0x5b6f, 0x5a32, 0x5125, 0x5c38, 0x5876, 0x5124, 0x4d62, 0x5c6a,
01543 0x7077, 0x704a, 0x503e, 0x5d5c, 0x5456, 0x5356, 0x6d50, 0x4d21,
01544 0x5f35, 0x5f78, 0x5421, 0x4e32, 0x684a, 0x6b75, 0x6355, 0x7550,
01545 0x7521, 0x5927, 0x652b, 0x664b, 0x7571, 0x6545, 0x7923, 0x605b,
01546 0x766b, 0x4b71, 0x596a, 0x7522, 0x5751, 0x5178, 0x6a78, 0x6a79,
01547 0x5a33, 0x6f5f, 0x716f, 0x6576, 0x6e3f, 0x6264, 0x503f, 0x7a2c,
01548 0x7551, 0x6733, 0x693e, 0x724e, 0x5b34, 0x7c4e, 0x5d6e, 0x6734,
01549 0x5734, 0x7734, 0x4d3e, 0x5a69, 0x4f30, 0x7759, 0x7366, 0x4e59,
01550 0x4e2a, 0x4b48, 0x5027, 0x704b, 0x5047, 0x6445, 0x5b60, 0x555a,
01551 0x5727, 0x6e40, 0x7876, 0x7552, 0x6d69, 0x593c, 0x6546, 0x7523,
01552 0x5a54, 0x6227, 0x7b7c, 0x715c, 0x4a74, 0x687a, 0x4e69, 0x6978,
01553 0x6265, 0x5039, 0x5472, 0x5126, 0x5f4e, 0x7c74, 0x532a, 0x4c2c,
01554 0x6f60, 0x6565, 0x5055, 0x5b7c, 0x7c66, 0x4b7e, 0x6d6a, 0x5e31,
01555 0x7963, 0x5422, 0x4f76, 0x5650, 0x556a, 0x716e, 0x7a4b, 0x6521,
01556 0x5531, 0x4f6d, 0x6d6b, 0x5532, 0x553c, 0x7d62, 0x732d, 0x7d5b,
01557 0x6930, 0x5127, 0x7d63, 0x4e33, 0x7d64, 0x7a4e, 0x4a30, 0x7727,
01558 0x4f31, 0x6622, 0x7c36, 0x722d, 0x6f61, 0x732e, 0x5c46, 0x596b,
01559 0x6860, 0x6128, 0x5576, 0x4f7d, 0x5e5d, 0x5951, 0x646a, 0x724f,
01560 0x773f, 0x6266, 0x6228, 0x6356, 0x6d51, 0x6979, 0x5631, 0x5e32,
01561 0x6068, 0x532b, 0x6b5c, 0x5f2f, 0x4a43, 0x6e7c, 0x7d43, 0x6b76,
01562 0x4f32, 0x596c, 0x593d, 0x585f, 0x5438, 0x6b3e, 0x5d6f, 0x5d70,
01563 0x5d71, 0x5d72, 0x593e, 0x7b46, 0x4f33, 0x6e7d, 0x642b, 0x5a45,
01564 0x586c, 0x5128, 0x6229, 0x5e3c, 0x6735, 0x5b70, 0x6f62, 0x7170,
01565 0x4f34, 0x5b71, 0x6031, 0x5f25, 0x7952, 0x677d, 0x6623, 0x7b71,
01566 0x4b30, 0x722e, 0x4d67, 0x685c, 0x6757, 0x7740, 0x5063, 0x5a21,
01567 0x4c3d, 0x5129, 0x5d4c, 0x637e, 0x512a, 0x682a, 0x6a36, 0x797a,
01568 0x664c, 0x7658, 0x5447, 0x594b, 0x5952, 0x534b, 0x5877, 0x5a29,
01569 0x7578, 0x5e5e, 0x722f, 0x7829, 0x5848, 0x6e41, 0x7941, 0x5d73,
01570 0x6a7a, 0x763d, 0x613b, 0x4d3f, 0x7454, 0x664d, 0x7c4f, 0x7b22,
01571 0x605c, 0x743b, 0x5a55, 0x7932, 0x7b72, 0x5b76, 0x5e5f, 0x5b72,
01572 0x785c, 0x776e, 0x6b68, 0x527a, 0x713c, 0x7a5a, 0x5a6a, 0x5a46,
01573 0x7741, 0x6736, 0x6547, 0x562c, 0x5c47, 0x6129, 0x622a, 0x5526,
01574 0x5457, 0x7250, 0x6a7b, 0x605d, 0x7b73, 0x713d, 0x6267, 0x7d57,
01575 0x4e48, 0x6a37, 0x7c40, 0x7d67, 0x776f, 0x5735, 0x6f3a, 0x715d,
01576 0x5e33, 0x684b, 0x785d, 0x7b47, 0x5548, 0x575f, 0x5d29, 0x6931,
01577 0x7a2d, 0x7659, 0x7a74, 0x782a, 0x666e, 0x4c5c, 0x613c, 0x606f,
01578 0x693f, 0x7c7d, 0x664e, 0x6157, 0x664f, 0x7471, 0x6473, 0x647b,
01579 0x7964, 0x6f63, 0x4f6e, 0x763e, 0x6032, 0x7c7e, 0x512b, 0x577a,
01580 0x7b48, 0x6257, 0x5423, 0x7078, 0x5728, 0x6167, 0x5333, 0x6f64,
01581 0x5745, 0x6b62, 0x7c67, 0x6422, 0x6268, 0x6650, 0x7b68, 0x7468,
01582 0x6574, 0x743c, 0x7455, 0x5f36, 0x7c39, 0x6e42, 0x4a75, 0x6f65,
01583 0x4b62, 0x5424, 0x5e60, 0x5a7d, 0x6446, 0x683e, 0x605e, 0x7634,
01584 0x6a52, 0x797b, 0x6042, 0x4a64, 0x6737, 0x6a7d, 0x595d, 0x5a34,
01585 0x6e2a, 0x7b69, 0x5b4b, 0x5a35, 0x713e, 0x532c, 0x7b49, 0x5f4f,
01586 0x5340, 0x6357, 0x6f66, 0x7c50, 0x6940, 0x7553, 0x6c5c, 0x7737,
01587 0x6a38, 0x5179, 0x5c48, 0x6a39, 0x715e, 0x5736, 0x4f35, 0x5928,
01588 0x6c6e, 0x5d2a, 0x4d22, 0x682e, 0x613d, 0x7251, 0x6941, 0x527c,
01589 0x5b35, 0x7367, 0x587e, 0x7c51, 0x6d32, 0x742f, 0x7b23, 0x7c41,
01590 0x6e2b, 0x5425, 0x7472, 0x6e59, 0x7b4a, 0x4d63, 0x583b, 0x655b,
01591 0x7877, 0x7654, 0x5729, 0x4b49, 0x6651, 0x704c, 0x582e, 0x7953,
01592 0x557e, 0x583c, 0x7230, 0x622b, 0x7368, 0x6f42, 0x6d6c, 0x6738,
01593 0x5a7e, 0x4c3e, 0x727c, 0x5a6b, 0x6258, 0x6d56, 0x5651, 0x6033,
01594 0x7c52, 0x6b48, 0x5341, 0x704d, 0x4f77, 0x6d52, 0x5458, 0x5c49,
01595 0x5771, 0x5f3b, 0x7325, 0x744d, 0x713f, 0x7831, 0x697a, 0x7b4b,
01596 0x4a55, 0x7954, 0x774a, 0x5648, 0x7c68, 0x733d, 0x6e7e, 0x677e,
01597 0x5342, 0x5336, 0x4c2d, 0x767a, 0x5632, 0x5258, 0x6758, 0x6325,
```

```
01598 0x6739, 0x702d, 0x7b4c, 0x6b21, 0x5426, 0x7b4d, 0x553d, 0x715f,
01599 0x767b, 0x5e34, 0x556b, 0x6548, 0x7b24, 0x5439, 0x5e61, 0x6423,
01600 0x5737, 0x786e, 0x5e35, 0x5652, 0x7955, 0x673a, 0x6b55, 0x5577,
01601 0x6f67, 0x613e, 0x7a2e, 0x5669, 0x566e, 0x673b, 0x6c4b, 0x5533,
01602 0x4e34, 0x7b25, 0x616e, 0x7728, 0x7b4e, 0x583d, 0x7b7d, 0x7c69,
01603 0x4f36, 0x6d47, 0x6e2c, 0x4c5d, 0x7627, 0x667a, 0x7524, 0x7d5c,
01604 0x6d33, 0x4e49, 0x6f68, 0x613f, 0x7a5b, 0x4b63, 0x7729, 0x7b26,
01605 0x5c39, 0x7140, 0x6d48, 0x6f43, 0x562d, 0x7d4e, 0x6821, 0x7b74,
01606 0x5527, 0x7176, 0x6653, 0x4c5e, 0x7832, 0x5c6b, 0x7d36, 0x656a,
01607 0x7160, 0x5b4c, 0x5d4d, 0x5448, 0x596d, 0x7525, 0x667b, 0x6654,
01608 0x7d48, 0x5621, 0x7d3f, 0x7c53, 0x6f21, 0x673c, 0x516e, 0x6655,
01609 0x6972, 0x5f30, 0x5860, 0x7c3a, 0x7d2f, 0x704e, 0x5b61, 0x6549,
01610 0x6d34, 0x6043, 0x6358, 0x697b, 0x6a28, 0x7d37, 0x7b27, 0x6942,
01611 0x7d77, 0x6259, 0x5c6c, 0x6822, 0x6670, 0x7d78, 0x7d79, 0x763f,
01612 0x6727, 0x6657, 0x5473, 0x5449, 0x567a, 0x5772, 0x6140, 0x5b62,
01613 0x6658, 0x673d, 0x704f, 0x733e, 0x622c, 0x7537, 0x6070, 0x7d38,
01614 0x6368, 0x5427, 0x687c, 0x7a52, 0x786f, 0x5653, 0x5534, 0x7050,
01615 0x7770, 0x6e33, 0x6a3a, 0x6a53, 0x6d49, 0x5d2b, 0x652c, 0x7d21,
01616 0x5f50, 0x6c33, 0x5f51, 0x6d6d, 0x7838, 0x777a, 0x782b, 0x7460,
01617 0x543a, 0x6433, 0x695a, 0x5e36, 0x593f, 0x5940, 0x566f, 0x594c,
01618 0x5a2a, 0x5f65, 0x7765, 0x4c32, 0x5f79, 0x5760, 0x543b, 0x7d7a,
01619 0x4c33, 0x5b73, 0x5f52, 0x4e4a, 0x6e5a, 0x6464, 0x7b4f, 0x4f37,
01620 0x6e43, 0x4e6a, 0x622d, 0x5761, 0x7a75, 0x5549, 0x782c, 0x6759,
01621 0x7369, 0x586d, 0x6344, 0x7071, 0x6865, 0x607a, 0x6e44, 0x595e,
01622 0x6b22, 0x6b23, 0x7c42, 0x6a3b, 0x682b, 0x5e62, 0x6d6f, 0x6823,
01623 0x4f71, 0x543c, 0x7c6a, 0x673e, 0x7c72, 0x5634, 0x622e, 0x5337,
01624 0x7a4c, 0x7a5c, 0x6d35, 0x6163, 0x682c, 0x685d, 0x6f69, 0x743d,
01625 0x4f38, 0x695b, 0x512c, 0x5a47, 0x6b49, 0x684c, 0x5e37, 0x563c,
01626 0x5365, 0x7a5d, 0x5a56, 0x4a31, 0x5a48, 0x5f26, 0x7933, 0x7252,
01627 0x4a44, 0x4e4b, 0x4d75, 0x7d30, 0x5528, 0x7141, 0x6269, 0x5c4a,
01628 0x6c34, 0x7a40, 0x7b28, 0x5028, 0x5a6c, 0x596e, 0x607b, 0x6f6a,
01629 0x7a5e, 0x6044, 0x4f39, 0x554a, 0x5762, 0x622f, 0x5738, 0x684d,
01630 0x765a, 0x6f22, 0x625a, 0x767c, 0x7b50, 0x512d, 0x4d64, 0x512e,
01631 0x5c6d, 0x684e, 0x7079, 0x4e35, 0x667c, 0x577b, 0x5056, 0x5d75,
01632 0x7771, 0x767d, 0x5b77, 0x7b6a, 0x695c, 0x5941, 0x7572, 0x6045,
01633 0x6a54, 0x7942, 0x6a3c, 0x5245, 0x7b51, 0x6740, 0x6b25, 0x5f7a,
01634 0x6322, 0x5739, 0x6943, 0x687d, 0x682f, 0x7253, 0x7b29, 0x5825,
01635 0x554b, 0x5048, 0x512f, 0x5763, 0x6046, 0x5622, 0x6d70, 0x5773,
01636 0x7c54, 0x5a57, 0x4c5f, 0x7254, 0x5130, 0x4c60, 0x5b7d, 0x733f,
01637 0x7051, 0x7c3b, 0x6230, 0x6625, 0x625b, 0x5f5e, 0x6047, 0x726f,
01638 0x4c61, 0x566a, 0x6742, 0x4e36, 0x7340, 0x4d7e, 0x7b52, 0x7878,
01639 0x777b, 0x683f, 0x6837, 0x6d36, 0x5c3a, 0x4c34, 0x7177, 0x6838,
01640 0x4a76, 0x6424, 0x7456, 0x5f66, 0x5f27, 0x5f67, 0x6141, 0x6944,
01641 0x5c4b, 0x6945, 0x6f23, 0x6b26, 0x4b23, 0x6369, 0x517b, 0x6f24,
01642 0x6f6b, 0x5034, 0x4d23, 0x6866, 0x6f25, 0x534c, 0x5a6d, 0x573a,
01643 0x7255, 0x7565, 0x596f, 0x7934, 0x5554, 0x7d4f, 0x5b63, 0x7161,
01644 0x6c36, 0x7b7e, 0x5357, 0x5131, 0x4b31, 0x5132, 0x4b32, 0x7142,
01645 0x7461, 0x7935, 0x6143, 0x6142, 0x6b77, 0x5f28, 0x4b4a, 0x6639,
01646 0x785e, 0x792a, 0x4a77, 0x6d37, 0x5338, 0x7256, 0x5459, 0x6e45,
01647 0x7270, 0x4a32, 0x5c3b, 0x7178, 0x6c37, 0x654a, 0x7640, 0x7d5d,
01648 0x5463, 0x4c62, 0x7754, 0x5765, 0x5343, 0x5826, 0x7641, 0x5d76,
01649 0x4d40, 0x655c, 0x654b, 0x6144, 0x6830, 0x7430, 0x736a, 0x5a6e,
01650 0x573b, 0x6231, 0x572a, 0x567b, 0x645f, 0x4a56, 0x6b28, 0x5b7e,
01651 0x7642, 0x6f3b, 0x547d, 0x6048, 0x6839, 0x6f26, 0x4d24, 0x5474,
01652 0x5b21, 0x5b5c, 0x5b5d, 0x6e5c, 0x4b4b, 0x7c55, 0x4e6b, 0x4d41,
01653 0x7b53, 0x792b, 0x7554, 0x5929, 0x695d, 0x5b4d, 0x5d4e, 0x6743,
01654 0x6c4c, 0x796c, 0x4b4c, 0x607c, 0x5428, 0x6d53, 0x586e, 0x7257,
01655 0x4a78, 0x5a6f, 0x5654, 0x594d, 0x586e, 0x7241, 0x5f53, 0x5a70,
01656 0x626a, 0x607d, 0x5878, 0x772f, 0x5a36, 0x4a57, 0x7258, 0x5879,
01657 0x7a5f, 0x4f6f, 0x5942, 0x7052, 0x6451, 0x7337, 0x7a60, 0x6f6c,
01658 0x6232, 0x543d, 0x594e, 0x7462, 0x5429, 0x4d42, 0x675a, 0x7259,
01659 0x592a, 0x583e, 0x5c2d, 0x626b, 0x567c, 0x4a79, 0x545a, 0x7457,
01660 0x4c21, 0x4f3a, 0x7538, 0x5943, 0x5068, 0x6345, 0x6b78, 0x7231,
01661 0x4f3b, 0x532d, 0x6861, 0x4e6c, 0x6034, 0x5e63, 0x5d77, 0x7232,
01662 0x7376, 0x765b, 0x577e, 0x785f, 0x7772, 0x5029, 0x665a, 0x7526,
01663 0x573c, 0x4c63, 0x665b, 0x5d5d, 0x5133, 0x6f6d, 0x565e, 0x6474,
01664 0x616f, 0x5d78, 0x684f, 0x4a65, 0x5c21, 0x6035, 0x7c2c, 0x7c2d,
01665 0x5827, 0x6d38, 0x5b36, 0x5670, 0x732f, 0x4d25, 0x5a71, 0x5828,
01666 0x4c64, 0x5134, 0x4a58, 0x5a72, 0x7527, 0x7528, 0x6626, 0x556c,
01667 0x5578, 0x5a73, 0x6346, 0x5e64, 0x5e65, 0x5135, 0x5136, 0x5137,
01668 0x7233, 0x695e, 0x7053, 0x7234, 0x7054, 0x4b64, 0x7b54, 0x7566,
01669 0x636a, 0x5e66, 0x5f54, 0x7879, 0x702e, 0x5138, 0x565f, 0x5057,
01670 0x7c21, 0x6f6e, 0x5c58, 0x695f, 0x655d, 0x7d7b, 0x6049, 0x5649,
01671 0x542a, 0x654c, 0x6960, 0x5058, 0x7c22, 0x543e, 0x6233, 0x5e67,
01672 0x5c3c, 0x5236, 0x7555, 0x4e21, 0x7529, 0x5d79, 0x5d7a, 0x7055,
01673 0x765f, 0x725a, 0x646b, 0x7271, 0x6c39, 0x7d7c, 0x612a, 0x4a59,
01674 0x6f6f, 0x752a, 0x6c79, 0x782d, 0x7242, 0x7643, 0x5752, 0x7922,
01675 0x7056, 0x707a, 0x7660, 0x6973, 0x7243, 0x542b, 0x4a33, 0x4d26,
01676 0x4d43, 0x4d5a, 0x594f, 0x7644, 0x6e5d, 0x6744, 0x6234, 0x5f62,
01677 0x675b, 0x6831, 0x7c2e, 0x654d, 0x7a6b, 0x4f3c, 0x4f62, 0x4d76,
01678 0x6f70, 0x743e, 0x544d, 0x7338, 0x6921, 0x7272, 0x736b, 0x7057,
01679 0x4f57, 0x4f5f, 0x6840, 0x6841, 0x4f63, 0x6922, 0x502a, 0x7341,
01680 0x502b, 0x5464, 0x6f3c, 0x5821, 0x595f, 0x7357, 0x5c3d, 0x4c65,
01681 0x6d71, 0x7162, 0x545b, 0x6235, 0x4a66, 0x532e, 0x4c66, 0x7153,
01682 0x7567, 0x4a5a, 0x7b6e, 0x6145, 0x5f69, 0x6e5e, 0x7742, 0x5822,
01683 0x5d2c, 0x702f, 0x563d, 0x612b, 0x7936, 0x5475, 0x5049, 0x6f27,
01684 0x626c, 0x5b6a, 0x4e4c, 0x7568, 0x7755, 0x534d, 0x737e, 0x5035,
```

```

01685 0x607e, 0x5f7b, 0x665d, 0x6824, 0x4b4d, 0x6f28, 0x6e34, 0x5a58,
01686 0x5139, 0x5f29, 0x7330, 0x4c44, 0x4e37, 0x6f29, 0x5f55, 0x6d57,
01687 0x6e46, 0x6f3d, 0x7c56, 0x5b74, 0x6f2a, 0x7839, 0x7569, 0x6359,
01688 0x6146, 0x543f, 0x5e68, 0x706a, 0x7342, 0x532f, 0x4a5b, 0x7c57,
01689 0x6d58, 0x6147, 0x7458, 0x5633, 0x5d2d, 0x553e, 0x7143, 0x6e5f,
01690 0x555d, 0x4d27, 0x502c, 0x513a, 0x7144, 0x6533, 0x7b75, 0x6961,
01691 0x7058, 0x6f44, 0x6158, 0x7154, 0x6d72, 0x555b, 0x555c, 0x7344,
01692 0x4b57, 0x6236, 0x6f71, 0x7b55, 0x5358, 0x5d50, 0x7059, 0x4b33,
01693 0x555d, 0x4d27, 0x502c, 0x513a, 0x7144, 0x6533, 0x7b75, 0x6961,
01694 0x7d60, 0x7c3c, 0x5a22, 0x5a23, 0x5221, 0x526f, 0x626d, 0x5e69,
01695 0x4e5c, 0x7235, 0x5064, 0x5d51, 0x6148, 0x5b37, 0x5f63, 0x6d39,
01696 0x7145, 0x734f, 0x572b, 0x612c, 0x636b, 0x6e47, 0x6149, 0x4a7a,
01697 0x707b, 0x7a61, 0x705a, 0x4c67, 0x5a74, 0x4c3f, 0x4e6d, 0x5529,
01698 0x7a62, 0x5065, 0x6b56, 0x6c5f, 0x5f7c, 0x7756, 0x5e6a, 0x4b34,
01699 0x6f3e, 0x4c35, 0x4f3d, 0x6f72, 0x6237, 0x4c68, 0x707c, 0x5660,
01700 0x7146, 0x6238, 0x6b2b, 0x4b35, 0x5851, 0x744e, 0x7377, 0x5746,
01701 0x513b, 0x772a, 0x6d4a, 0x5753, 0x587a, 0x7645, 0x514c, 0x5d7c,
01702 0x5f7d, 0x7965, 0x604a, 0x727d, 0x5330, 0x7473, 0x5a49, 0x665e,
01703 0x783a, 0x6850, 0x587b, 0x6a55, 0x5623, 0x7646, 0x725b, 0x647c,
01704 0x6832, 0x5a5a, 0x725c, 0x7b56, 0x6932, 0x6e2d, 0x7a63, 0x5c6e,
01705 0x756a, 0x6660, 0x707d, 0x572c, 0x7545, 0x6e60, 0x5b65, 0x5d5e,
01706 0x5970, 0x6923, 0x7179, 0x7244, 0x604b, 0x6924, 0x6239, 0x6331,
01707 0x7c6b, 0x4d28, 0x4c36, 0x705b, 0x663a, 0x4d29, 0x7343, 0x6159,
01708 0x6f2b, 0x6745, 0x6069, 0x7345, 0x5440, 0x553f, 0x5d2e, 0x797c,
01709 0x4c40, 0x6522, 0x4e38, 0x5852, 0x7956, 0x712a, 0x4e51, 0x7647,
01710 0x5b6b, 0x5f7e, 0x5861, 0x7773, 0x5767, 0x547e, 0x513c, 0x654f,
01711 0x4b36, 0x5a38, 0x4d44, 0x563e, 0x623a, 0x4f58, 0x604c, 0x6b79,
01712 0x7d7d, 0x5768, 0x4b58, 0x6962, 0x683a, 0x6347, 0x6c4d, 0x6c4e,
01713 0x563f, 0x6327, 0x5f56, 0x7d68, 0x6e61, 0x7628, 0x5d7d, 0x783b,
01714 0x6851, 0x7957, 0x4e6e, 0x6c4f, 0x6925, 0x5655, 0x4d45, 0x6d3a,
01715 0x513d, 0x4f3e, 0x6c3b, 0x5231, 0x4c69, 0x5944, 0x697c, 0x513e,
01716 0x6c3c, 0x652d, 0x7730, 0x4c6a, 0x5344, 0x5640, 0x567d, 0x6121,
01717 0x5e3d, 0x7629, 0x5a24, 0x5624, 0x7546, 0x6122, 0x6946, 0x7245,
01718 0x7469, 0x566c, 0x6b53, 0x6c3d, 0x625c, 0x5e6b, 0x705c, 0x6b3f,
01719 0x574e, 0x513f, 0x752b, 0x797d, 0x4a5c, 0x4d46, 0x7236, 0x5d7e,
01720 0x4c37, 0x5b38, 0x5069, 0x4e5d, 0x6b40, 0x7d22, 0x784b, 0x6a56,
01721 0x7130, 0x5b4e, 0x7743, 0x5b4f, 0x4b24, 0x7860, 0x7b57, 0x6b4a,
01722 0x6021, 0x4e4d, 0x545c, 0x7d58, 0x5276, 0x7237, 0x7a76, 0x762a,
01723 0x7a77, 0x5866, 0x7431, 0x6852, 0x4a45, 0x4c6b, 0x626e, 0x623b,
01724 0x772d, 0x7861, 0x736c, 0x5e21, 0x647d, 0x636c, 0x5d2f, 0x5d30,
01725 0x4b37, 0x6853, 0x6123, 0x5260, 0x707e, 0x6926, 0x4b72, 0x6d73,
01726 0x5c59, 0x604d, 0x775a, 0x5b39, 0x4c2e, 0x5a5b, 0x4d47, 0x5d31,
01727 0x582f, 0x6323, 0x4e6f, 0x7273, 0x7833, 0x604e, 0x757d, 0x6b6c,
01728 0x5345, 0x7c6c, 0x525b, 0x546b, 0x5e22, 0x6566, 0x7030, 0x5544,
01729 0x6d74, 0x636d, 0x6842, 0x6d75, 0x577c, 0x6d3b, 0x762b, 0x7238,
01730 0x7648, 0x5366, 0x725d, 0x4f3f, 0x6b2c, 0x4f40, 0x6628, 0x7d69,
01731 0x4f41, 0x605f, 0x5e6c, 0x6022, 0x743f, 0x626f, 0x5971, 0x7147,
01732 0x4b38, 0x797e, 0x5b3a, 0x5a75, 0x766c, 0x5a5c, 0x7a64, 0x604f,
01733 0x5d32, 0x6629, 0x6f73, 0x736d, 0x6b7a, 0x7966, 0x4a5d, 0x555e,
01734 0x4a5e, 0x5f64, 0x667d, 0x752c, 0x6475, 0x6963, 0x6d4b, 0x4f64,
01735 0x5853, 0x5d33, 0x546c, 0x7239, 0x5f37, 0x4b4e, 0x7b58, 0x5059,
01736 0x5d52, 0x7774, 0x675c, 0x6425, 0x7c23, 0x5b3b, 0x723a, 0x697d,
01737 0x504a, 0x7556, 0x5945, 0x6434, 0x6d27, 0x6a3d, 0x667e, 0x7744,
01738 0x752d, 0x5960, 0x4a34, 0x7862, 0x4f42, 0x6c3e, 0x6534, 0x4d48,
01739 0x6e48, 0x6748, 0x4d49, 0x7937, 0x7168, 0x5972, 0x5b75, 0x4a35,
01740 0x5946, 0x5849, 0x592b, 0x6d3c, 0x5854, 0x5c5a, 0x623c, 0x7c6d,
01741 0x6c60, 0x527e, 0x6947, 0x662a, 0x6270, 0x7a3b, 0x752e, 0x7b2a,
01742 0x6c7b, 0x6c3f, 0x7c58, 0x5465, 0x7943, 0x6e62, 0x5769, 0x6d76,
01743 0x5e6d, 0x4c6c, 0x636e, 0x6854, 0x7a78, 0x5d34, 0x6435, 0x5830,
01744 0x5855, 0x746a, 0x4e39, 0x5661, 0x4f52, 0x5036, 0x4e22, 0x736e,
01745 0x7378, 0x5c4c, 0x504b, 0x7c24, 0x4d4a, 0x5754, 0x5e23, 0x6460,
01746 0x6e49, 0x625d, 0x757e, 0x542c, 0x5551, 0x5870, 0x7843, 0x6a57,
01747 0x7557, 0x583f, 0x7d40, 0x6b2d, 0x552a, 0x6728, 0x6e4a, 0x4a67,
01748 0x7863, 0x545d, 0x6a58, 0x7b59, 0x6d77, 0x6535, 0x502d, 0x7171,
01749 0x623d, 0x6348, 0x5955, 0x5f2a, 0x5b3c, 0x7864, 0x717a, 0x6536,
01750 0x736f, 0x7b5a, 0x6160, 0x592c, 0x756b, 0x6036, 0x6948, 0x4b4f,
01751 0x6349, 0x5e6e, 0x623e, 0x5c6f, 0x5625, 0x6271, 0x567e, 0x5921,
01752 0x5840, 0x5c5b, 0x6d3d, 0x5f38, 0x6a25, 0x572d, 0x7379, 0x6d78,
01753 0x7547, 0x614a, 0x6b63, 0x725e, 0x784c, 0x6a59, 0x5346, 0x5b66,
01754 0x752f, 0x4e70, 0x697e, 0x7b36, 0x6272, 0x4f72, 0x7739, 0x5973,
01755 0x614b, 0x5a5d, 0x5a39, 0x6b7b, 0x4b39, 0x6d79, 0x6060, 0x7440,
01756 0x7d3c, 0x5f31, 0x636f, 0x6023, 0x7d39, 0x7031, 0x4d4b, 0x6d3e,
01757 0x5540, 0x6370, 0x6d7a, 0x6964, 0x556d, 0x675d, 0x5476, 0x6537,
01758 0x5b67, 0x623f, 0x6e4b, 0x5774, 0x705d, 0x4e2b, 0x675e, 0x5656,
01759 0x614c, 0x6833, 0x656e, 0x5c22, 0x6050, 0x5535, 0x5521, 0x7b5b,
01760 0x794b, 0x4b73, 0x7425, 0x7a48, 0x5657, 0x6965, 0x7b5c, 0x7d50,
01761 0x7b76, 0x5a25, 0x5b3d, 0x6c62, 0x4d77, 0x705e, 0x7649, 0x5e6f,
01762 0x5331, 0x7c6e, 0x6843, 0x7148, 0x4e71, 0x796d, 0x7274, 0x6436,
01763 0x7539, 0x5c70, 0x6371, 0x6825, 0x723b, 0x5e24, 0x5a4c, 0x4a69,
01764 0x635a, 0x7c59, 0x6a5a, 0x7944, 0x6324, 0x7b5d, 0x6f4a, 0x6844,
01765 0x554c, 0x6b57, 0x592d, 0x7b2b, 0x5359, 0x5522, 0x765e, 0x5a76,
01766 0x6051, 0x6928, 0x7579, 0x7a2f, 0x6b7c, 0x606a, 0x6332, 0x5545,
01767 0x7163, 0x556e, 0x4d4c, 0x6d59, 0x5841, 0x7a6c, 0x716b, 0x7a3c,
01768 0x6662, 0x7a65, 0x627a, 0x4a36, 0x6437, 0x6a5b, 0x757a, 0x7b2c,
01769 0x4f43, 0x6b7d, 0x787a, 0x5f39, 0x6171, 0x5224, 0x757b, 0x505a,
01770 0x505b, 0x6a3e, 0x5931, 0x4a37, 0x5367, 0x7865, 0x5332, 0x6240,
01771 0x725f, 0x4d65, 0x792c, 0x4d4d, 0x6e2e, 0x562e, 0x576a, 0x6760,

```

```
01772 0x6b2e, 0x4f59, 0x5c4d, 0x6d7b, 0x5e70, 0x576b, 0x5e25, 0x5f57,
01773 0x5b50, 0x5b51, 0x5523, 0x7032, 0x5c5c, 0x4a68, 0x7866, 0x5c4e,
01774 0x6a5c, 0x5b52, 0x6933, 0x775b, 0x6328, 0x572e, 0x6061, 0x4b3a,
01775 0x6551, 0x505c, 0x5541, 0x584a, 0x6329, 0x6024, 0x6929, 0x5347,
01776 0x5c5d, 0x782e, 0x4c38, 0x502e, 0x5872, 0x634a, 0x4c2f, 0x542d,
01777 0x7651, 0x504c, 0x4a46, 0x5542, 0x4e3a, 0x4a47, 0x7a30, 0x5f58,
01778 0x753a, 0x656b, 0x6f74, 0x5d35, 0x4d2a, 0x6372, 0x7b77, 0x7750,
01779 0x7d3a, 0x7d61, 0x767e, 0x5140, 0x6845, 0x6438, 0x6168, 0x4c41,
01780 0x526d, 0x5b3e, 0x6062, 0x7a49, 0x614d, 0x4a38, 0x7260, 0x7149,
01781 0x5e71, 0x705f, 0x7844, 0x6e4c, 0x5e72, 0x6749, 0x6273, 0x6761,
01782 0x634b, 0x634c, 0x4f78, 0x6f2c, 0x7d7e, 0x7c25, 0x7a31, 0x5f59,
01783 0x6052, 0x745a, 0x714a, 0x4e23, 0x723c, 0x6c63, 0x6025, 0x772b,
01784 0x6b2f, 0x655e, 0x6124, 0x4d2b, 0x5974, 0x6826, 0x4d4e, 0x6169,
01785 0x7c6f, 0x6063, 0x6241, 0x4e24, 0x5e26, 0x6b7e, 0x6b5d, 0x7060,
01786 0x745b, 0x6274, 0x5348, 0x746b, 0x6e35, 0x7558, 0x555f, 0x5665,
01787 0x6b30, 0x7463, 0x634d, 0x7474, 0x7a32, 0x6f75, 0x4a5f, 0x6b31,
01788 0x6d3f, 0x7d49, 0x6426, 0x7924, 0x7033, 0x656c, 0x5167, 0x5947,
01789 0x6457, 0x6a5d, 0x5477, 0x5a3a, 0x5a4d, 0x794c, 0x615a, 0x5b3f,
01790 0x4c45, 0x6c50, 0x4b3b, 0x5e73, 0x692a, 0x5948, 0x6e63, 0x573d,
01791 0x4f44, 0x504d, 0x7c26, 0x717b, 0x7d52, 0x5141, 0x635b, 0x5349,
01792 0x5c4f, 0x4c6d, 0x5e27, 0x663b, 0x6c21, 0x4c39, 0x7b5e, 0x6762,
01793 0x5441, 0x5c28, 0x6242, 0x7358, 0x6553, 0x7359, 0x7346, 0x4d5b,
01794 0x4d2c, 0x7c43, 0x5467, 0x5142, 0x7925, 0x6855, 0x634e, 0x544a,
01795 0x5f5a, 0x7b5f, 0x6763, 0x787b, 0x634f, 0x7530, 0x5867, 0x5949,
01796 0x782f, 0x6f76, 0x5d36, 0x6e2f, 0x4d78, 0x5e38, 0x7c27, 0x777c,
01797 0x7731, 0x4e3b, 0x7421, 0x6e4d, 0x612e, 0x6c43, 0x4f7e, 0x783f,
01798 0x5862, 0x5368, 0x5e28, 0x7464, 0x6c42, 0x5975, 0x7945, 0x5d53,
01799 0x5671, 0x6c7c, 0x7c70, 0x6d40, 0x4a39, 0x6e64, 0x7261, 0x5e39,
01800 0x5672, 0x5e74, 0x5f5b, 0x5b53, 0x7a67, 0x5863, 0x7441, 0x5d37,
01801 0x7275, 0x542e, 0x5673, 0x5d38, 0x4f45, 0x5f5f, 0x723c, 0x7621,
01802 0x6b4b, 0x717c, 0x7347, 0x606b, 0x6d7c, 0x615b, 0x6e65, 0x5e75,
01803 0x7a53, 0x714b, 0x502f, 0x5d39, 0x5143, 0x7531, 0x6a4e, 0x7061,
01804 0x762c, 0x7559, 0x706b, 0x5d3a, 0x723f, 0x7745, 0x5b22, 0x7276,
01805 0x4a3a, 0x7775, 0x4b65, 0x6e66, 0x6053, 0x4e25, 0x5658, 0x542f,
01806 0x6949, 0x534e, 0x7442, 0x4b66, 0x7121, 0x6b32, 0x7122, 0x6b33,
01807 0x7034, 0x4b74, 0x5430, 0x7332, 0x7b37, 0x756c, 0x6e67, 0x7432,
01808 0x756d, 0x4f73, 0x7062, 0x6e4e, 0x714c, 0x6538, 0x5775, 0x6373,
01809 0x4f65, 0x4f46, 0x7333, 0x6458, 0x4f79, 0x4f5a, 0x7a4d, 0x6663,
01810 0x7262, 0x756e, 0x4a3b, 0x635c, 0x4e72, 0x5659, 0x6e30, 0x7465,
01811 0x5842, 0x5c50, 0x4c6e, 0x5560, 0x764a, 0x7d4a, 0x5856, 0x744f,
01812 0x5626, 0x5c3e, 0x5b54, 0x5747, 0x727e, 0x714d, 0x6243, 0x5c5e,
01813 0x5c5f, 0x6f2d, 0x662b, 0x795d, 0x6a3f, 0x6f2e, 0x7450, 0x4e73,
01814 0x662c, 0x4e5e, 0x5579, 0x6374, 0x4d50, 0x5538, 0x777d, 0x5c29,
01815 0x5e76, 0x5c2a, 0x7263, 0x6934, 0x525c, 0x6966, 0x6376, 0x674a,
01816 0x504e, 0x5a77, 0x4a3c, 0x6e68, 0x5a5e, 0x7277, 0x627b, 0x4c26,
01817 0x5a3b, 0x6e69, 0x755a, 0x775c, 0x616a, 0x4e41, 0x5431, 0x7d31,
01818 0x663d, 0x7b2d, 0x7867, 0x614e, 0x7762, 0x756f, 0x4f47, 0x5432,
01819 0x4c6f, 0x5468, 0x6e4f, 0x7757, 0x6026, 0x5641, 0x615c, 0x7063,
01820 0x7164, 0x5c71, 0x5627, 0x7475, 0x714e, 0x7264, 0x5030, 0x6c6f,
01821 0x793a, 0x6b35, 0x546d, 0x6244, 0x6967, 0x6b34, 0x6a21, 0x783c,
01822 0x4e26, 0x7946, 0x7c5a, 0x5433, 0x5339, 0x6a5e, 0x692b, 0x6161,
01823 0x534f, 0x7476, 0x6a40, 0x614f, 0x4c3a, 0x6e6a, 0x7064, 0x7334,
01824 0x546e, 0x7240, 0x7165, 0x7443, 0x6054, 0x6b36, 0x5721, 0x4b68,
01825 0x792d, 0x692d, 0x5864, 0x7a33, 0x6245, 0x7c3d, 0x6c44, 0x5831,
01826 0x5c2b, 0x5524, 0x6b69, 0x683b, 0x5857, 0x7b2e, 0x5161, 0x5b40,
01827 0x753e, 0x5e77, 0x4a7b, 0x7746, 0x4f48, 0x6150, 0x6e50, 0x6974,
01828 0x4e74, 0x554d, 0x4f5b, 0x5d3b, 0x4e2c, 0x6968, 0x5434, 0x6447,
01829 0x755b, 0x7a41, 0x5e29, 0x5478, 0x6f77, 0x5333, 0x6b37, 0x6f78,
01830 0x755c, 0x6d4c, 0x5b55, 0x714f, 0x7150, 0x7532, 0x592e, 0x552c,
01831 0x6246, 0x7d23, 0x7b65, 0x5f2b, 0x6275, 0x762d, 0x7533, 0x7035,
01832 0x6125, 0x755d, 0x6c22, 0x6d7d, 0x7534, 0x7b38, 0x5b23, 0x564a,
01833 0x4b59, 0x6554, 0x737a, 0x6b38, 0x6037, 0x576c, 0x716c, 0x652f,
01834 0x5561, 0x576d, 0x5151, 0x6172, 0x6f79, 0x5d3c, 0x765c, 0x7065,
01835 0x7444, 0x6969, 0x737b, 0x546f, 0x4c22, 0x777e, 0x5f3c, 0x6b4d,
01836 0x5037, 0x5642, 0x682d, 0x6f2f, 0x4b25, 0x4b69, 0x7a68, 0x4c46,
01837 0x6667, 0x6a47, 0x5b24, 0x4f49, 0x627c, 0x6f7a, 0x6b5e, 0x7548,
01838 0x545e, 0x6055, 0x6f30, 0x6247, 0x592f, 0x7967, 0x6765, 0x4f4a,
01839 0x6151, 0x6248, 0x6f7b, 0x7a79, 0x5c72, 0x6027, 0x7868, 0x4b6a,
01840 0x4b3c, 0x5662, 0x755e, 0x755f, 0x6e36, 0x6276, 0x534a, 0x6f7c,
01841 0x5144, 0x6f31, 0x5145, 0x505e, 0x5961, 0x6038, 0x4d51, 0x7339,
01842 0x674c, 0x5628, 0x4e27, 0x5435, 0x6448, 0x5334, 0x6b39, 0x4b75,
01843 0x765d, 0x7123, 0x4c47, 0x694a, 0x6170, 0x7560, 0x7b2f, 0x4b51,
01844 0x7b60, 0x7265, 0x6c70, 0x706c, 0x6e6b, 0x694b, 0x4c70, 0x572f,
01845 0x7321, 0x7c75, 0x7124, 0x6056, 0x6f32, 0x7451, 0x7721, 0x7151,
01846 0x4a7c, 0x4a7d, 0x4e4e, 0x7348, 0x733a, 0x6d7e, 0x5a26, 0x606c,
01847 0x784d, 0x4b52, 0x6b4e, 0x7958, 0x7959, 0x4a60, 0x5a4a, 0x4b26,
01848 0x4a48, 0x796e, 0x796f, 0x5b6c, 0x5031, 0x556f, 0x6673, 0x6722, 0x6459,
01849 0x6461, 0x7c44, 0x796f, 0x4f74, 0x7766, 0x4e3c, 0x7445, 0x5c23,
01850 0x5d3d, 0x7446, 0x7821, 0x6856, 0x5b41, 0x7066, 0x6439, 0x766d,
01851 0x792e, 0x5d3e, 0x5730, 0x5868, 0x4b3d, 0x795a, 0x784e, 0x7970,
01852 0x606d, 0x6333, 0x7433, 0x6a42, 0x7266, 0x7036, 0x5b56, 0x6b64,
01853 0x7267, 0x5755, 0x5436, 0x7968, 0x5741, 0x6555, 0x696a, 0x574c,
01854 0x5369, 0x6249, 0x7c5b, 0x4d2d, 0x4c30, 0x6a22, 0x6476, 0x5040,
01855 0x7037, 0x6e21, 0x5776, 0x624a, 0x624b, 0x7a4f, 0x6b5f, 0x564b,
01856 0x7434, 0x6d4d, 0x6452, 0x6a29, 0x643a, 0x7322, 0x4d52, 0x764b,
01857 0x7166, 0x6d41, 0x683c, 0x6e51, 0x7067, 0x624c, 0x642a, 0x7561,
01858 0x6d5a, 0x576e, 0x5171, 0x696b, 0x696c, 0x6064, 0x5a27, 0x5d54,
```

```
01859 0x6a23, 0x5643, 0x5674, 0x5a5f, 0x6f33, 0x624d, 0x6f7d, 0x7268,
01860 0x6f45, 0x6767, 0x577d, 0x674e, 0x5f5c, 0x7947, 0x5976, 0x5f2c,
01861 0x565a, 0x5c24, 0x7038, 0x557a, 0x6477, 0x5644, 0x746c, 0x6f7e,
01862 0x7021, 0x5e2a, 0x5a3c, 0x587c, 0x7a54, 0x6c65, 0x7c28, 0x6c66,
01863 0x584b, 0x7b39, 0x6453, 0x4d79, 0x4f53, 0x4a6a, 0x4f54, 0x783d,
01864 0x7447, 0x6a5f, 0x795b, 0x5437, 0x6b65, 0x6152, 0x6a24, 0x7a42,
01865 0x7b61, 0x7a6d, 0x7022, 0x4c71, 0x7a23, 0x6277, 0x624e, 0x6975,
01866 0x616b, 0x6768, 0x6857, 0x5a78, 0x544b, 0x7776, 0x5645, 0x5469,
01867 0x7a7a, 0x4c72, 0x775d, 0x5e3a, 0x4e28, 0x7039, 0x647e, 0x6449,
01868 0x6454, 0x6a43, 0x6f34, 0x573e, 0x7b62, 0x4d53, 0x6f35, 0x7a69,
01869 0x7926, 0x5f3d, 0x7747, 0x787d, 0x787c, 0x5e2b, 0x5b68, 0x635d,
01870 0x6162, 0x5146, 0x7650, 0x6b66, 0x5a79, 0x6c47, 0x5e78, 0x7869,
01871 0x635e, 0x4e75, 0x7a43, 0x6557, 0x6c48, 0x7349, 0x643b, 0x662e,
01872 0x6f36, 0x5c3f, 0x4e3d, 0x5843, 0x504f, 0x4f7a, 0x734a, 0x6057,
01873 0x5147, 0x692e, 0x683d, 0x7a44, 0x624f, 0x7a45, 0x7938, 0x5c60,
01874 0x7b30, 0x5829, 0x655f, 0x7927, 0x766e, 0x764c, 0x6278, 0x6c71,
01875 0x5a60, 0x7152, 0x524c, 0x4f4b, 0x4a3d, 0x5d3f, 0x766f, 0x5e79,
01876 0x7a34, 0x552d, 0x7167, 0x5e3e, 0x5c40, 0x5148, 0x5149, 0x783e,
01877 0x4b76, 0x5479, 0x7562, 0x6153, 0x5869, 0x787e, 0x4f4c, 0x7d24,
01878 0x4e76, 0x7a50, 0x4c73, 0x663e, 0x762e, 0x5570, 0x514a, 0x7c3e,
01879 0x5571, 0x4d69, 0x7a35, 0x6250, 0x7477, 0x4d54, 0x6723, 0x5b25,
01880 0x6251, 0x5722, 0x7763, 0x6a26, 0x5021, 0x4e5a, 0x7b6b, 0x5b26,
01881 0x5b5e, 0x5865, 0x6a60, 0x582a, 0x6560, 0x565b, 0x6f46, 0x786a,
01882 0x6455, 0x4e77, 0x6058, 0x576f, 0x746d, 0x4d66, 0x4c74, 0x7563,
01883 0x644a, 0x5c61, 0x7948, 0x7c3f, 0x6827, 0x5844, 0x4b3e, 0x5c2e,
01884 0x5777, 0x7068, 0x5d40, 0x4f4d, 0x5c73, 0x5930, 0x6669, 0x643c,
01885 0x6a44, 0x646c, 0x6465, 0x7b78, 0x4c3b, 0x643d, 0x4d5c, 0x5977,
01886 0x5d5f, 0x6d4e, 0x5950, 0x6523, 0x794d, 0x4d2e, 0x4f4e, 0x762f,
01887 0x7d53, 0x6b6d, 0x565c, 0x6524, 0x5536, 0x565d, 0x7969, 0x6724,
01888 0x5663, 0x514b, 0x5664, 0x5572, 0x5e7a, 0x5778, 0x586a, 0x4f55,
01889 0x587d, 0x582b, 0x7d4b, 0x7c5c, 0x6028, 0x5573, 0x7d59, 0x4c23,
01890 0x5979, 0x536a, 0x7575, 0x6f47, 0x535a, 0x5a3d, 0x6828, 0x5c2f,
01891 0x7023, 0x4d55, 0x6029, 0x5e2c, 0x703a, 0x6e31, 0x6e32, 0x764d,
01892 0x6e52, 0x5646, 0x6065, 0x733b, 0x6561, 0x644b, 0x5723, 0x5b42,
01893 0x4a7e, 0x4f4f, 0x3021, 0x3022, 0x3023, 0x3024, 0x3025, 0x3026,
01894 0x3027, 0x3028, 0x3029, 0x302a, 0x302b, 0x302c, 0x302d, 0x302e,
01895 0x302f, 0x3030, 0x3031, 0x3032, 0x3033, 0x3034, 0x3035, 0x3036,
01896 0x3037, 0x3038, 0x3039, 0x303a, 0x303b, 0x303c, 0x303d, 0x303e,
01897 0x303f, 0x3040, 0x3041, 0x3042, 0x3043, 0x3044, 0x3045, 0x3046,
01898 0x3047, 0x3048, 0x3049, 0x304a, 0x304b, 0x304c, 0x304d, 0x304e,
01899 0x304f, 0x3050, 0x3051, 0x3052, 0x3053, 0x3054, 0x3055, 0x3056,
01900 0x3057, 0x3058, 0x3059, 0x305a, 0x305b, 0x305c, 0x305d, 0x305e,
01901 0x305f, 0x3060, 0x3061, 0x3062, 0x3063, 0x3064, 0x3065, 0x3066,
01902 0x3067, 0x3068, 0x3069, 0x306a, 0x306b, 0x306c, 0x306d, 0x306e,
01903 0x306f, 0x3070, 0x3071, 0x3072, 0x3073, 0x3074, 0x3075, 0x3076,
01904 0x3077, 0x3078, 0x3079, 0x307a, 0x307b, 0x307c, 0x307d, 0x307e,
01905 0x3121, 0x3122, 0x3123, 0x3124, 0x3125, 0x3126, 0x3127, 0x3128,
01906 0x3129, 0x312a, 0x312b, 0x312c, 0x312d, 0x312e, 0x312f, 0x3130,
01907 0x3131, 0x3132, 0x3133, 0x3134, 0x3135, 0x3136, 0x3137, 0x3138,
01908 0x3139, 0x313a, 0x313b, 0x313c, 0x313d, 0x313e, 0x313f, 0x3140,
01909 0x3141, 0x3142, 0x3143, 0x3144, 0x3145, 0x3146, 0x3147, 0x3148,
01910 0x3149, 0x314a, 0x314b, 0x314c, 0x314d, 0x314e, 0x314f, 0x3150,
01911 0x3151, 0x3152, 0x3153, 0x3154, 0x3155, 0x3156, 0x3157, 0x3158,
01912 0x3159, 0x315a, 0x315b, 0x315c, 0x315d, 0x315e, 0x315f, 0x3160,
01913 0x3161, 0x3162, 0x3163, 0x3164, 0x3165, 0x3166, 0x3167, 0x3168,
01914 0x3169, 0x316a, 0x316b, 0x316c, 0x316d, 0x316e, 0x316f, 0x3170,
01915 0x3171, 0x3172, 0x3173, 0x3174, 0x3175, 0x3176, 0x3177, 0x3178,
01916 0x3179, 0x317a, 0x317b, 0x317c, 0x317d, 0x317e, 0x3221, 0x3222,
01917 0x3223, 0x3224, 0x3225, 0x3226, 0x3227, 0x3228, 0x3229, 0x322a,
01918 0x322b, 0x322c, 0x322d, 0x322e, 0x322f, 0x3230, 0x3231, 0x3232,
01919 0x3233, 0x3234, 0x3235, 0x3236, 0x3237, 0x3238, 0x3239, 0x323a,
01920 0x323b, 0x323c, 0x323d, 0x323e, 0x323f, 0x3240, 0x3241, 0x3242,
01921 0x3243, 0x3244, 0x3245, 0x3246, 0x3247, 0x3248, 0x3249, 0x324a,
01922 0x324b, 0x324c, 0x324d, 0x324e, 0x324f, 0x3250, 0x3251, 0x3252,
01923 0x3253, 0x3254, 0x3255, 0x3256, 0x3257, 0x3258, 0x3259, 0x325a,
01924 0x325b, 0x325c, 0x325d, 0x325e, 0x325f, 0x3260, 0x3261, 0x3262,
01925 0x3263, 0x3264, 0x3265, 0x3266, 0x3267, 0x3268, 0x3269, 0x326a,
01926 0x326b, 0x326c, 0x326d, 0x326e, 0x326f, 0x3270, 0x3271, 0x3272,
01927 0x3273, 0x3274, 0x3275, 0x3276, 0x3277, 0x3278, 0x3279, 0x327a,
01928 0x327b, 0x327c, 0x327d, 0x327e, 0x3321, 0x3322, 0x3323, 0x3324,
01929 0x3325, 0x3326, 0x3327, 0x3328, 0x3329, 0x332a, 0x332b, 0x332c,
01930 0x332d, 0x332e, 0x332f, 0x3330, 0x3331, 0x3332, 0x3333, 0x3334,
01931 0x3335, 0x3336, 0x3337, 0x3338, 0x3339, 0x333a, 0x333b, 0x333c,
01932 0x333d, 0x333e, 0x333f, 0x3340, 0x3341, 0x3342, 0x3343, 0x3344,
01933 0x3345, 0x3346, 0x3347, 0x3348, 0x3349, 0x334a, 0x334b, 0x334c,
01934 0x334d, 0x334e, 0x334f, 0x3350, 0x3351, 0x3352, 0x3353, 0x3354,
01935 0x3355, 0x3356, 0x3357, 0x3358, 0x3359, 0x335a, 0x335b, 0x335c,
01936 0x335d, 0x335e, 0x335f, 0x3360, 0x3361, 0x3362, 0x3363, 0x3364,
01937 0x3365, 0x3366, 0x3367, 0x3368, 0x3369, 0x336a, 0x336b, 0x336c,
01938 0x336d, 0x336e, 0x336f, 0x3370, 0x3371, 0x3372, 0x3373, 0x3374,
01939 0x3375, 0x3376, 0x3377, 0x3378, 0x3379, 0x337a, 0x337b, 0x337c,
01940 0x337d, 0x337e, 0x3421, 0x3422, 0x3423, 0x3424, 0x3425, 0x3426,
01941 0x3427, 0x3428, 0x3429, 0x342a, 0x342b, 0x342c, 0x342d, 0x342e,
01942 0x342f, 0x3430, 0x3431, 0x3432, 0x3433, 0x3434, 0x3435, 0x3436,
01943 0x3437, 0x3438, 0x3439, 0x343a, 0x343b, 0x343c, 0x343d, 0x343e,
01944 0x343f, 0x3440, 0x3441, 0x3442, 0x3443, 0x3444, 0x3445, 0x3446,
01945 0x3447, 0x3448, 0x3449, 0x344a, 0x344b, 0x344c, 0x344d, 0x344e,
```


01946 0x344f, 0x3450, 0x3451, 0x3452, 0x3453, 0x3454, 0x3455, 0x3456,
01947 0x3457, 0x3458, 0x3459, 0x345a, 0x345b, 0x345c, 0x345d, 0x345e,
01948 0x345f, 0x3460, 0x3461, 0x3462, 0x3463, 0x3464, 0x3465, 0x3466,
01949 0x3467, 0x3468, 0x3469, 0x346a, 0x346b, 0x346c, 0x346d, 0x346e,
01950 0x346f, 0x3470, 0x3471, 0x3472, 0x3473, 0x3474, 0x3475, 0x3476,
01951 0x3477, 0x3478, 0x3479, 0x347a, 0x347b, 0x347c, 0x347d, 0x347e,
01952 0x3521, 0x3522, 0x3523, 0x3524, 0x3525, 0x3526, 0x3527, 0x3528,
01953 0x3529, 0x352a, 0x352b, 0x352c, 0x352d, 0x352e, 0x352f, 0x3530,
01954 0x3531, 0x3532, 0x3533, 0x3534, 0x3535, 0x3536, 0x3537, 0x3538,
01955 0x3539, 0x353a, 0x353b, 0x353c, 0x353d, 0x353e, 0x353f, 0x3540,
01956 0x3541, 0x3542, 0x3543, 0x3544, 0x3545, 0x3546, 0x3547, 0x3548,
01957 0x3549, 0x354a, 0x354b, 0x354c, 0x354d, 0x354e, 0x354f, 0x3550,
01958 0x3551, 0x3552, 0x3553, 0x3554, 0x3555, 0x3556, 0x3557, 0x3558,
01959 0x3559, 0x355a, 0x355b, 0x355c, 0x355d, 0x355e, 0x355f, 0x3560,
01960 0x3561, 0x3562, 0x3563, 0x3564, 0x3565, 0x3566, 0x3567, 0x3568,
01961 0x3569, 0x356a, 0x356b, 0x356c, 0x356d, 0x356e, 0x356f, 0x3570,
01962 0x3571, 0x3572, 0x3573, 0x3574, 0x3575, 0x3576, 0x3577, 0x3578,
01963 0x3579, 0x357a, 0x357b, 0x357c, 0x357d, 0x357e, 0x357f, 0x3580,
01964 0x3623, 0x3624, 0x3625, 0x3626, 0x3627, 0x3628, 0x3629, 0x362a,
01965 0x362b, 0x362c, 0x362d, 0x362e, 0x362f, 0x3630, 0x3631, 0x3632,
01966 0x3633, 0x3634, 0x3635, 0x3636, 0x3637, 0x3638, 0x3639, 0x363a,
01967 0x363b, 0x363c, 0x363d, 0x363e, 0x363f, 0x3640, 0x3641, 0x3642,
01968 0x3643, 0x3644, 0x3645, 0x3646, 0x3647, 0x3648, 0x3649, 0x364a,
01969 0x364b, 0x364c, 0x364d, 0x364e, 0x364f, 0x3650, 0x3651, 0x3652,
01970 0x3653, 0x3654, 0x3655, 0x3656, 0x3657, 0x3658, 0x3659, 0x365a,
01971 0x365b, 0x365c, 0x365d, 0x365e, 0x365f, 0x3660, 0x3661, 0x3662,
01972 0x3663, 0x3664, 0x3665, 0x3666, 0x3667, 0x3668, 0x3669, 0x366a,
01973 0x366b, 0x366c, 0x366d, 0x366e, 0x366f, 0x3670, 0x3671, 0x3672,
01974 0x3673, 0x3674, 0x3675, 0x3676, 0x3677, 0x3678, 0x3679, 0x367a,
01975 0x367b, 0x367c, 0x367d, 0x367e, 0x367f, 0x3680, 0x3681, 0x3682,
01976 0x3725, 0x3726, 0x3727, 0x3728, 0x3729, 0x372a, 0x372b, 0x372c,
01977 0x372d, 0x372e, 0x372f, 0x3730, 0x3731, 0x3732, 0x3733, 0x3734,
01978 0x3735, 0x3736, 0x3737, 0x3738, 0x3739, 0x373a, 0x373b, 0x373c,
01979 0x373d, 0x373e, 0x373f, 0x3740, 0x3741, 0x3742, 0x3743, 0x3744,
01980 0x3745, 0x3746, 0x3747, 0x3748, 0x3749, 0x374a, 0x374b, 0x374c,
01981 0x374d, 0x374e, 0x374f, 0x3750, 0x3751, 0x3752, 0x3753, 0x3754,
01982 0x3755, 0x3756, 0x3757, 0x3758, 0x3759, 0x375a, 0x375b, 0x375c,
01983 0x375d, 0x375e, 0x375f, 0x3760, 0x3761, 0x3762, 0x3763, 0x3764,
01984 0x3765, 0x3766, 0x3767, 0x3768, 0x3769, 0x376a, 0x376b, 0x376c,
01985 0x376d, 0x376e, 0x376f, 0x3770, 0x3771, 0x3772, 0x3773, 0x3774,
01986 0x3775, 0x3776, 0x3777, 0x3778, 0x3779, 0x377a, 0x377b, 0x377c,
01987 0x377d, 0x377e, 0x377f, 0x3780, 0x3781, 0x3782, 0x3783, 0x3784,
01988 0x3827, 0x3828, 0x3829, 0x382a, 0x382b, 0x382c, 0x382d, 0x382e,
01989 0x382f, 0x3830, 0x3831, 0x3832, 0x3833, 0x3834, 0x3835, 0x3836,
01990 0x3837, 0x3838, 0x3839, 0x383a, 0x383b, 0x383c, 0x383d, 0x383e,
01991 0x383f, 0x3840, 0x3841, 0x3842, 0x3843, 0x3844, 0x3845, 0x3846,
01992 0x3847, 0x3848, 0x3849, 0x384a, 0x384b, 0x384c, 0x384d, 0x384e,
01993 0x384f, 0x3850, 0x3851, 0x3852, 0x3853, 0x3854, 0x3855, 0x3856,
01994 0x3857, 0x3858, 0x3859, 0x385a, 0x385b, 0x385c, 0x385d, 0x385e,
01995 0x385f, 0x3860, 0x3861, 0x3862, 0x3863, 0x3864, 0x3865, 0x3866,
01996 0x3867, 0x3868, 0x3869, 0x386a, 0x386b, 0x386c, 0x386d, 0x386e,
01997 0x386f, 0x3870, 0x3871, 0x3872, 0x3873, 0x3874, 0x3875, 0x3876,
01998 0x3877, 0x3878, 0x3879, 0x387a, 0x387b, 0x387c, 0x387d, 0x387e,
01999 0x3921, 0x3922, 0x3923, 0x3924, 0x3925, 0x3926, 0x3927, 0x3928,
02000 0x3929, 0x392a, 0x392b, 0x392c, 0x392d, 0x392e, 0x392f, 0x3930,
02001 0x3931, 0x3932, 0x3933, 0x3934, 0x3935, 0x3936, 0x3937, 0x3938,
02002 0x3939, 0x393a, 0x393b, 0x393c, 0x393d, 0x393e, 0x393f, 0x3940,
02003 0x3941, 0x3942, 0x3943, 0x3944, 0x3945, 0x3946, 0x3947, 0x3948,
02004 0x3949, 0x394a, 0x394b, 0x394c, 0x394d, 0x394e, 0x394f, 0x3950,
02005 0x3951, 0x3952, 0x3953, 0x3954, 0x3955, 0x3956, 0x3957, 0x3958,
02006 0x3959, 0x395a, 0x395b, 0x395c, 0x395d, 0x395e, 0x395f, 0x3960,
02007 0x3961, 0x3962, 0x3963, 0x3964, 0x3965, 0x3966, 0x3967, 0x3968,
02008 0x3969, 0x396a, 0x396b, 0x396c, 0x396d, 0x396e, 0x396f, 0x3970,
02009 0x3971, 0x3972, 0x3973, 0x3974, 0x3975, 0x3976, 0x3977, 0x3978,
02010 0x3979, 0x397a, 0x397b, 0x397c, 0x397d, 0x397e, 0x397f, 0x3980,
02011 0x3a23, 0x3a24, 0x3a25, 0x3a26, 0x3a27, 0x3a28, 0x3a29, 0x3a2a,
02012 0x3a2b, 0x3a2c, 0x3a2d, 0x3a2e, 0x3a2f, 0x3a30, 0x3a31, 0x3a32,
02013 0x3a33, 0x3a34, 0x3a35, 0x3a36, 0x3a37, 0x3a38, 0x3a39, 0x3a3a,
02014 0x3a3b, 0x3a3c, 0x3a3d, 0x3a3e, 0x3a3f, 0x3a40, 0x3a41, 0x3a42,
02015 0x3a43, 0x3a44, 0x3a45, 0x3a46, 0x3a47, 0x3a48, 0x3a49, 0x3a4a,
02016 0x3a4b, 0x3a4c, 0x3a4d, 0x3a4e, 0x3a4f, 0x3a50, 0x3a51, 0x3a52,
02017 0x3a53, 0x3a54, 0x3a55, 0x3a56, 0x3a57, 0x3a58, 0x3a59, 0x3a5a,
02018 0x3a5b, 0x3a5c, 0x3a5d, 0x3a5e, 0x3a5f, 0x3a60, 0x3a61, 0x3a62,
02019 0x3a63, 0x3a64, 0x3a65, 0x3a66, 0x3a67, 0x3a68, 0x3a69, 0x3a6a,
02020 0x3a6b, 0x3a6c, 0x3a6d, 0x3a6e, 0x3a6f, 0x3a70, 0x3a71, 0x3a72,
02021 0x3a73, 0x3a74, 0x3a75, 0x3a76, 0x3a77, 0x3a78, 0x3a79, 0x3a7a,
02022 0x3a7b, 0x3a7c, 0x3a7d, 0x3a7e, 0x3a7f, 0x3a80, 0x3a81, 0x3a82,
02023 0x3b25, 0x3b26, 0x3b27, 0x3b28, 0x3b29, 0x3b2a, 0x3b2b, 0x3b2c,
02024 0x3b2d, 0x3b2e, 0x3b2f, 0x3b30, 0x3b31, 0x3b32, 0x3b33, 0x3b34,
02025 0x3b35, 0x3b36, 0x3b37, 0x3b38, 0x3b39, 0x3b3a, 0x3b3b, 0x3b3c,
02026 0x3b3d, 0x3b3e, 0x3b3f, 0x3b40, 0x3b41, 0x3b42, 0x3b43, 0x3b44,
02027 0x3b45, 0x3b46, 0x3b47, 0x3b48, 0x3b49, 0x3b4a, 0x3b4b, 0x3b4c,
02028 0x3b4d, 0x3b4e, 0x3b4f, 0x3b50, 0x3b51, 0x3b52, 0x3b53, 0x3b54,
02029 0x3b55, 0x3b56, 0x3b57, 0x3b58, 0x3b59, 0x3b5a, 0x3b5b, 0x3b5c,
02030 0x3b5d, 0x3b5e, 0x3b5f, 0x3b60, 0x3b61, 0x3b62, 0x3b63, 0x3b64,
02031 0x3b65, 0x3b66, 0x3b67, 0x3b68, 0x3b69, 0x3b6a, 0x3b6b, 0x3b6c,
02032 0x3b6d, 0x3b6e, 0x3b6f, 0x3b70, 0x3b71, 0x3b72, 0x3b73, 0x3b74,

```
02033 0x3b75, 0x3b76, 0x3b77, 0x3b78, 0x3b79, 0x3b7a, 0x3b7b, 0x3b7c,
02034 0x3b7d, 0x3b7e, 0x3c21, 0x3c22, 0x3c23, 0x3c24, 0x3c25, 0x3c26,
02035 0x3c27, 0x3c28, 0x3c29, 0x3c2a, 0x3c2b, 0x3c2c, 0x3c2d, 0x3c2e,
02036 0x3c2f, 0x3c30, 0x3c31, 0x3c32, 0x3c33, 0x3c34, 0x3c35, 0x3c36,
02037 0x3c37, 0x3c38, 0x3c39, 0x3c3a, 0x3c3b, 0x3c3c, 0x3c3d, 0x3c3e,
02038 0x3c3f, 0x3c40, 0x3c41, 0x3c42, 0x3c43, 0x3c44, 0x3c45, 0x3c46,
02039 0x3c47, 0x3c48, 0x3c49, 0x3c4a, 0x3c4b, 0x3c4c, 0x3c4d, 0x3c4e,
02040 0x3c4f, 0x3c50, 0x3c51, 0x3c52, 0x3c53, 0x3c54, 0x3c55, 0x3c56,
02041 0x3c57, 0x3c58, 0x3c59, 0x3c5a, 0x3c5b, 0x3c5c, 0x3c5d, 0x3c5e,
02042 0x3c5f, 0x3c60, 0x3c61, 0x3c62, 0x3c63, 0x3c64, 0x3c65, 0x3c66,
02043 0x3c67, 0x3c68, 0x3c69, 0x3c6a, 0x3c6b, 0x3c6c, 0x3c6d, 0x3c6e,
02044 0x3c6f, 0x3c70, 0x3c71, 0x3c72, 0x3c73, 0x3c74, 0x3c75, 0x3c76,
02045 0x3c77, 0x3c78, 0x3c79, 0x3c7a, 0x3c7b, 0x3c7c, 0x3c7d, 0x3c7e,
02046 0x3d21, 0x3d22, 0x3d23, 0x3d24, 0x3d25, 0x3d26, 0x3d27, 0x3d28,
02047 0x3d29, 0x3d2a, 0x3d2b, 0x3d2c, 0x3d2d, 0x3d2e, 0x3d2f, 0x3d30,
02048 0x3d31, 0x3d32, 0x3d33, 0x3d34, 0x3d35, 0x3d36, 0x3d37, 0x3d38,
02049 0x3d39, 0x3d3a, 0x3d3b, 0x3d3c, 0x3d3d, 0x3d3e, 0x3d3f, 0x3d40,
02050 0x3d41, 0x3d42, 0x3d43, 0x3d44, 0x3d45, 0x3d46, 0x3d47, 0x3d48,
02051 0x3d49, 0x3d4a, 0x3d4b, 0x3d4c, 0x3d4d, 0x3d4e, 0x3d4f, 0x3d50,
02052 0x3d51, 0x3d52, 0x3d53, 0x3d54, 0x3d55, 0x3d56, 0x3d57, 0x3d58,
02053 0x3d59, 0x3d5a, 0x3d5b, 0x3d5c, 0x3d5d, 0x3d5e, 0x3d5f, 0x3d60,
02054 0x3d61, 0x3d62, 0x3d63, 0x3d64, 0x3d65, 0x3d66, 0x3d67, 0x3d68,
02055 0x3d69, 0x3d6a, 0x3d6b, 0x3d6c, 0x3d6d, 0x3d6e, 0x3d6f, 0x3d70,
02056 0x3d71, 0x3d72, 0x3d73, 0x3d74, 0x3d75, 0x3d76, 0x3d77, 0x3d78,
02057 0x3d79, 0x3d7a, 0x3d7b, 0x3d7c, 0x3d7d, 0x3d7e, 0x3e21, 0x3e22,
02058 0x3e23, 0x3e24, 0x3e25, 0x3e26, 0x3e27, 0x3e28, 0x3e29, 0x3e2a,
02059 0x3e2b, 0x3e2c, 0x3e2d, 0x3e2e, 0x3e2f, 0x3e30, 0x3e31, 0x3e32,
02060 0x3e33, 0x3e34, 0x3e35, 0x3e36, 0x3e37, 0x3e38, 0x3e39, 0x3e3a,
02061 0x3e3b, 0x3e3c, 0x3e3d, 0x3e3e, 0x3e3f, 0x3e40, 0x3e41, 0x3e42,
02062 0x3e43, 0x3e44, 0x3e45, 0x3e46, 0x3e47, 0x3e48, 0x3e49, 0x3e4a,
02063 0x3e4b, 0x3e4c, 0x3e4d, 0x3e4e, 0x3e4f, 0x3e50, 0x3e51, 0x3e52,
02064 0x3e53, 0x3e54, 0x3e55, 0x3e56, 0x3e57, 0x3e58, 0x3e59, 0x3e5a,
02065 0x3e5b, 0x3e5c, 0x3e5d, 0x3e5e, 0x3e5f, 0x3e60, 0x3e61, 0x3e62,
02066 0x3e63, 0x3e64, 0x3e65, 0x3e66, 0x3e67, 0x3e68, 0x3e69, 0x3e6a,
02067 0x3e6b, 0x3e6c, 0x3e6d, 0x3e6e, 0x3e6f, 0x3e70, 0x3e71, 0x3e72,
02068 0x3e73, 0x3e74, 0x3e75, 0x3e76, 0x3e77, 0x3e78, 0x3e79, 0x3e7a,
02069 0x3e7b, 0x3e7c, 0x3e7d, 0x3e7e, 0x3f21, 0x3f22, 0x3f23, 0x3f24,
02070 0x3f25, 0x3f26, 0x3f27, 0x3f28, 0x3f29, 0x3f2a, 0x3f2b, 0x3f2c,
02071 0x3f2d, 0x3f2e, 0x3f2f, 0x3f30, 0x3f31, 0x3f32, 0x3f33, 0x3f34,
02072 0x3f35, 0x3f36, 0x3f37, 0x3f38, 0x3f39, 0x3f3a, 0x3f3b, 0x3f3c,
02073 0x3f3d, 0x3f3e, 0x3f3f, 0x3f40, 0x3f41, 0x3f42, 0x3f43, 0x3f44,
02074 0x3f45, 0x3f46, 0x3f47, 0x3f48, 0x3f49, 0x3f4a, 0x3f4b, 0x3f4c,
02075 0x3f4d, 0x3f4e, 0x3f4f, 0x3f50, 0x3f51, 0x3f52, 0x3f53, 0x3f54,
02076 0x3f55, 0x3f56, 0x3f57, 0x3f58, 0x3f59, 0x3f5a, 0x3f5b, 0x3f5c,
02077 0x3f5d, 0x3f5e, 0x3f5f, 0x3f60, 0x3f61, 0x3f62, 0x3f63, 0x3f64,
02078 0x3f65, 0x3f66, 0x3f67, 0x3f68, 0x3f69, 0x3f6a, 0x3f6b, 0x3f6c,
02079 0x3f6d, 0x3f6e, 0x3f6f, 0x3f70, 0x3f71, 0x3f72, 0x3f73, 0x3f74,
02080 0x3f75, 0x3f76, 0x3f77, 0x3f78, 0x3f79, 0x3f7a, 0x3f7b, 0x3f7c,
02081 0x3f7d, 0x3f7e, 0x4021, 0x4022, 0x4023, 0x4024, 0x4025, 0x4026,
02082 0x4027, 0x4028, 0x4029, 0x402a, 0x402b, 0x402c, 0x402d, 0x402e,
02083 0x402f, 0x4030, 0x4031, 0x4032, 0x4033, 0x4034, 0x4035, 0x4036,
02084 0x4037, 0x4038, 0x4039, 0x403a, 0x403b, 0x403c, 0x403d, 0x403e,
02085 0x403f, 0x4040, 0x4041, 0x4042, 0x4043, 0x4044, 0x4045, 0x4046,
02086 0x4047, 0x4048, 0x4049, 0x404a, 0x404b, 0x404c, 0x404d, 0x404e,
02087 0x404f, 0x4050, 0x4051, 0x4052, 0x4053, 0x4054, 0x4055, 0x4056,
02088 0x4057, 0x4058, 0x4059, 0x405a, 0x405b, 0x405c, 0x405d, 0x405e,
02089 0x405f, 0x4060, 0x4061, 0x4062, 0x4063, 0x4064, 0x4065, 0x4066,
02090 0x4067, 0x4068, 0x4069, 0x406a, 0x406b, 0x406c, 0x406d, 0x406e,
02091 0x406f, 0x4070, 0x4071, 0x4072, 0x4073, 0x4074, 0x4075, 0x4076,
02092 0x4077, 0x4078, 0x4079, 0x407a, 0x407b, 0x407c, 0x407d, 0x407e,
02093 0x4121, 0x4122, 0x4123, 0x4124, 0x4125, 0x4126, 0x4127, 0x4128,
02094 0x4129, 0x412a, 0x412b, 0x412c, 0x412d, 0x412e, 0x412f, 0x4130,
02095 0x4131, 0x4132, 0x4133, 0x4134, 0x4135, 0x4136, 0x4137, 0x4138,
02096 0x4139, 0x413a, 0x413b, 0x413c, 0x413d, 0x413e, 0x413f, 0x4140,
02097 0x4141, 0x4142, 0x4143, 0x4144, 0x4145, 0x4146, 0x4147, 0x4148,
02098 0x4149, 0x414a, 0x414b, 0x414c, 0x414d, 0x414e, 0x414f, 0x4150,
02099 0x4151, 0x4152, 0x4153, 0x4154, 0x4155, 0x4156, 0x4157, 0x4158,
02100 0x4159, 0x415a, 0x415b, 0x415c, 0x415d, 0x415e, 0x415f, 0x4160,
02101 0x4161, 0x4162, 0x4163, 0x4164, 0x4165, 0x4166, 0x4167, 0x4168,
02102 0x4169, 0x416a, 0x416b, 0x416c, 0x416d, 0x416e, 0x416f, 0x4170,
02103 0x4171, 0x4172, 0x4173, 0x4174, 0x4175, 0x4176, 0x4177, 0x4178,
02104 0x4179, 0x417a, 0x417b, 0x417c, 0x417d, 0x417e, 0x4221, 0x4222,
02105 0x4223, 0x4224, 0x4225, 0x4226, 0x4227, 0x4228, 0x4229, 0x422a,
02106 0x422b, 0x422c, 0x422d, 0x422e, 0x422f, 0x4230, 0x4231, 0x4232,
02107 0x4233, 0x4234, 0x4235, 0x4236, 0x4237, 0x4238, 0x4239, 0x423a,
02108 0x423b, 0x423c, 0x423d, 0x423e, 0x423f, 0x4240, 0x4241, 0x4242,
02109 0x4243, 0x4244, 0x4245, 0x4246, 0x4247, 0x4248, 0x4249, 0x424a,
02110 0x424b, 0x424c, 0x424d, 0x424e, 0x424f, 0x4250, 0x4251, 0x4252,
02111 0x4253, 0x4254, 0x4255, 0x4256, 0x4257, 0x4258, 0x4259, 0x425a,
02112 0x425b, 0x425c, 0x425d, 0x425e, 0x425f, 0x4260, 0x4261, 0x4262,
02113 0x4263, 0x4264, 0x4265, 0x4266, 0x4267, 0x4268, 0x4269, 0x426a,
02114 0x426b, 0x426c, 0x426d, 0x426e, 0x426f, 0x4270, 0x4271, 0x4272,
02115 0x4273, 0x4274, 0x4275, 0x4276, 0x4277, 0x4278, 0x4279, 0x427a,
02116 0x427b, 0x427c, 0x427d, 0x427e, 0x4321, 0x4322, 0x4323, 0x4324,
02117 0x4325, 0x4326, 0x4327, 0x4328, 0x4329, 0x432a, 0x432b, 0x432c,
02118 0x432d, 0x432e, 0x432f, 0x4330, 0x4331, 0x4332, 0x4333, 0x4334,
02119 0x4335, 0x4336, 0x4337, 0x4338, 0x4339, 0x433a, 0x433b, 0x433c,
```

```
02120 0x433d, 0x433e, 0x433f, 0x4340, 0x4341, 0x4342, 0x4343, 0x4344,
02121 0x4345, 0x4346, 0x4347, 0x4348, 0x4349, 0x434a, 0x434b, 0x434c,
02122 0x434d, 0x434e, 0x434f, 0x4350, 0x4351, 0x4352, 0x4353, 0x4354,
02123 0x4355, 0x4356, 0x4357, 0x4358, 0x4359, 0x435a, 0x435b, 0x435c,
02124 0x435d, 0x435e, 0x435f, 0x4360, 0x4361, 0x4362, 0x4363, 0x4364,
02125 0x4365, 0x4366, 0x4367, 0x4368, 0x4369, 0x436a, 0x436b, 0x436c,
02126 0x436d, 0x436e, 0x436f, 0x4370, 0x4371, 0x4372, 0x4373, 0x4374,
02127 0x4375, 0x4376, 0x4377, 0x4378, 0x4379, 0x437a, 0x437b, 0x437c,
02128 0x437d, 0x437e, 0x4421, 0x4422, 0x4423, 0x4424, 0x4425, 0x4426,
02129 0x4427, 0x4428, 0x4429, 0x442a, 0x442b, 0x442c, 0x442d, 0x442e,
02130 0x442f, 0x4430, 0x4431, 0x4432, 0x4433, 0x4434, 0x4435, 0x4436,
02131 0x4437, 0x4438, 0x4439, 0x443a, 0x443b, 0x443c, 0x443d, 0x443e,
02132 0x443f, 0x4440, 0x4441, 0x4442, 0x4443, 0x4444, 0x4445, 0x4446,
02133 0x4447, 0x4448, 0x4449, 0x444a, 0x444b, 0x444c, 0x444d, 0x444e,
02134 0x444f, 0x4450, 0x4451, 0x4452, 0x4453, 0x4454, 0x4455, 0x4456,
02135 0x4457, 0x4458, 0x4459, 0x445a, 0x445b, 0x445c, 0x445d, 0x445e,
02136 0x445f, 0x4460, 0x4461, 0x4462, 0x4463, 0x4464, 0x4465, 0x4466,
02137 0x4467, 0x4468, 0x4469, 0x446a, 0x446b, 0x446c, 0x446d, 0x446e,
02138 0x446f, 0x4470, 0x4471, 0x4472, 0x4473, 0x4474, 0x4475, 0x4476,
02139 0x4477, 0x4478, 0x4479, 0x447a, 0x447b, 0x447c, 0x447d, 0x447e,
02140 0x4521, 0x4522, 0x4523, 0x4524, 0x4525, 0x4526, 0x4527, 0x4528,
02141 0x4529, 0x452a, 0x452b, 0x452c, 0x452d, 0x452e, 0x452f, 0x4530,
02142 0x4531, 0x4532, 0x4533, 0x4534, 0x4535, 0x4536, 0x4537, 0x4538,
02143 0x4539, 0x453a, 0x453b, 0x453c, 0x453d, 0x453e, 0x453f, 0x4540,
02144 0x4541, 0x4542, 0x4543, 0x4544, 0x4545, 0x4546, 0x4547, 0x4548,
02145 0x4549, 0x454a, 0x454b, 0x454c, 0x454d, 0x454e, 0x454f, 0x4550,
02146 0x4551, 0x4552, 0x4553, 0x4554, 0x4555, 0x4556, 0x4557, 0x4558,
02147 0x4559, 0x455a, 0x455b, 0x455c, 0x455d, 0x455e, 0x455f, 0x4560,
02148 0x4561, 0x4562, 0x4563, 0x4564, 0x4565, 0x4566, 0x4567, 0x4568,
02149 0x4569, 0x456a, 0x456b, 0x456c, 0x456d, 0x456e, 0x456f, 0x4570,
02150 0x4571, 0x4572, 0x4573, 0x4574, 0x4575, 0x4576, 0x4577, 0x4578,
02151 0x4579, 0x457a, 0x457b, 0x457c, 0x457d, 0x457e, 0x4621, 0x4622,
02152 0x4623, 0x4624, 0x4625, 0x4626, 0x4627, 0x4628, 0x4629, 0x462a,
02153 0x462b, 0x462c, 0x462d, 0x462e, 0x462f, 0x4630, 0x4631, 0x4632,
02154 0x4633, 0x4634, 0x4635, 0x4636, 0x4637, 0x4638, 0x4639, 0x463a,
02155 0x463b, 0x463c, 0x463d, 0x463e, 0x463f, 0x4640, 0x4641, 0x4642,
02156 0x4643, 0x4644, 0x4645, 0x4646, 0x4647, 0x4648, 0x4649, 0x464a,
02157 0x464b, 0x464c, 0x464d, 0x464e, 0x464f, 0x4650, 0x4651, 0x4652,
02158 0x4653, 0x4654, 0x4655, 0x4656, 0x4657, 0x4658, 0x4659, 0x465a,
02159 0x465b, 0x465c, 0x465d, 0x465e, 0x465f, 0x4660, 0x4661, 0x4662,
02160 0x4663, 0x4664, 0x4665, 0x4666, 0x4667, 0x4668, 0x4669, 0x466a,
02161 0x466b, 0x466c, 0x466d, 0x466e, 0x466f, 0x4670, 0x4671, 0x4672,
02162 0x4673, 0x4674, 0x4675, 0x4676, 0x4677, 0x4678, 0x4679, 0x467a,
02163 0x467b, 0x467c, 0x467d, 0x467e, 0x4721, 0x4722, 0x4723, 0x4724,
02164 0x4725, 0x4726, 0x4727, 0x4728, 0x4729, 0x472a, 0x472b, 0x472c,
02165 0x472d, 0x472e, 0x472f, 0x4730, 0x4731, 0x4732, 0x4733, 0x4734,
02166 0x4735, 0x4736, 0x4737, 0x4738, 0x4739, 0x473a, 0x473b, 0x473c,
02167 0x473d, 0x473e, 0x473f, 0x4740, 0x4741, 0x4742, 0x4743, 0x4744,
02168 0x4745, 0x4746, 0x4747, 0x4748, 0x4749, 0x474a, 0x474b, 0x474c,
02169 0x474d, 0x474e, 0x474f, 0x4750, 0x4751, 0x4752, 0x4753, 0x4754,
02170 0x4755, 0x4756, 0x4757, 0x4758, 0x4759, 0x475a, 0x475b, 0x475c,
02171 0x475d, 0x475e, 0x475f, 0x4760, 0x4761, 0x4762, 0x4763, 0x4764,
02172 0x4765, 0x4766, 0x4767, 0x4768, 0x4769, 0x476a, 0x476b, 0x476c,
02173 0x476d, 0x476e, 0x476f, 0x4770, 0x4771, 0x4772, 0x4773, 0x4774,
02174 0x4775, 0x4776, 0x4777, 0x4778, 0x4779, 0x477a, 0x477b, 0x477c,
02175 0x477d, 0x477e, 0x4821, 0x4822, 0x4823, 0x4824, 0x4825, 0x4826,
02176 0x4827, 0x4828, 0x4829, 0x482a, 0x482b, 0x482c, 0x482d, 0x482e,
02177 0x482f, 0x4830, 0x4831, 0x4832, 0x4833, 0x4834, 0x4835, 0x4836,
02178 0x4837, 0x4838, 0x4839, 0x483a, 0x483b, 0x483c, 0x483d, 0x483e,
02179 0x483f, 0x4840, 0x4841, 0x4842, 0x4843, 0x4844, 0x4845, 0x4846,
02180 0x4847, 0x4848, 0x4849, 0x484a, 0x484b, 0x484c, 0x484d, 0x484e,
02181 0x484f, 0x4850, 0x4851, 0x4852, 0x4853, 0x4854, 0x4855, 0x4856,
02182 0x4857, 0x4858, 0x4859, 0x485a, 0x485b, 0x485c, 0x485d, 0x485e,
02183 0x485f, 0x4860, 0x4861, 0x4862, 0x4863, 0x4864, 0x4865, 0x4866,
02184 0x4867, 0x4868, 0x4869, 0x486a, 0x486b, 0x486c, 0x486d, 0x486e,
02185 0x486f, 0x4870, 0x4871, 0x4872, 0x4873, 0x4874, 0x4875, 0x4876,
02186 0x4877, 0x4878, 0x4879, 0x487a, 0x487b, 0x487c, 0x487d, 0x487e,
02187 0x4b50, 0x4b56, 0x4b67, 0x4d4f, 0x4d68, 0x4e2d, 0x4f7b, 0x5022,
02188 0x5038, 0x5050, 0x505d, 0x5154, 0x5155, 0x5158, 0x515b, 0x515c,
02189 0x515d, 0x515e, 0x515f, 0x5160, 0x5162, 0x5163, 0x5164, 0x5165,
02190 0x5166, 0x5168, 0x5169, 0x516a, 0x516b, 0x516d, 0x516f, 0x5170,
02191 0x5172, 0x5176, 0x517a, 0x517c, 0x517d, 0x517e, 0x5222, 0x5223,
02192 0x5227, 0x5228, 0x5229, 0x522a, 0x522b, 0x522d, 0x5232, 0x523e,
02193 0x5242, 0x5243, 0x5244, 0x5246, 0x5247, 0x5248, 0x5249, 0x524a,
02194 0x524b, 0x524d, 0x524e, 0x524f, 0x5250, 0x5251, 0x5252, 0x5253,
02195 0x5254, 0x5255, 0x5256, 0x5257, 0x5259, 0x525a, 0x525e, 0x525f,
02196 0x5261, 0x5262, 0x5264, 0x5265, 0x5266, 0x5267, 0x5268, 0x5269,
02197 0x526a, 0x526b, 0x5270, 0x5271, 0x5272, 0x5273, 0x5274, 0x5275,
02198 0x5277, 0x5278, 0x5466, 0x547c, 0x5525, 0x552b, 0x552e, 0x5638,
02199 0x564d, 0x574b, 0x5764, 0x5b64, 0x5c25, 0x5d25, 0x5d55,
02200 0x5d74, 0x5e7c, 0x5e7e, 0x5f33, 0x5f61, 0x5f68, 0x6071, 0x612d,
02201 0x616d, 0x6375, 0x6421, 0x6429, 0x652e, 0x6531, 0x6532, 0x6539,
02202 0x653b, 0x653c, 0x6544, 0x654e, 0x6550, 0x6552, 0x6556, 0x657a,
02203 0x657b, 0x657c, 0x657e, 0x6621, 0x6624, 0x6627, 0x662d, 0x662f,
02204 0x6630, 0x6631, 0x6633, 0x6637, 0x6638, 0x663c, 0x6644, 0x6646,
02205 0x6647, 0x664a, 0x6652, 0x6656, 0x6659, 0x665c, 0x665f, 0x6661,
02206 0x6664, 0x6665, 0x6666, 0x6668, 0x666a, 0x666b, 0x666c, 0x666f,
```

```

02207 0x6671, 0x6672, 0x6675, 0x6676, 0x6677, 0x6679, 0x6721, 0x6726,
02208 0x6729, 0x672a, 0x672c, 0x672d, 0x6730, 0x673f, 0x6741, 0x6746,
02209 0x6747, 0x674b, 0x674d, 0x674f, 0x6750, 0x6753, 0x675f, 0x6764,
02210 0x6766, 0x6777, 0x6867, 0x6868, 0x6870, 0x6871, 0x6877, 0x6879,
02211 0x687b, 0x687e, 0x6927, 0x692c, 0x694c, 0x6977, 0x6a41, 0x6a65,
02212 0x6a74, 0x6a77, 0x6a7c, 0x6a7e, 0x6b24, 0x6b27, 0x6b29, 0x6b2a,
02213 0x6b3a, 0x6b3b, 0x6b3d, 0x6b41, 0x6b42, 0x6b46, 0x6b47, 0x6b4c,
02214 0x6b4f, 0x6b50, 0x6b51, 0x6b52, 0x6b58, 0x6c26, 0x6c27, 0x6c2a,
02215 0x6c2f, 0x6c30, 0x6c31, 0x6c32, 0x6c35, 0x6c38, 0x6c3a, 0x6c40,
02216 0x6c41, 0x6c45, 0x6c46, 0x6c49, 0x6c4a, 0x6c55, 0x6c5d, 0x6c5e,
02217 0x6c61, 0x6c64, 0x6c67, 0x6c68, 0x6c77, 0x6c78, 0x6c7a, 0x6d21,
02218 0x6d22, 0x6d23, 0x6d6e, 0x6e5b, 0x723d, 0x727a, 0x7331, 0x7427,
02219 0x746e, 0x7674, 0x7676, 0x7738, 0x7748, 0x7753, 0x785b, 0x7870,
02220 0x7a21, 0x7a22, 0x7a66, 0x7c29, 0x2321, 0x2322, 0x2323, 0x2324,
02221 0x2325, 0x2326, 0x2327, 0x2328, 0x2329, 0x232a, 0x232b, 0x232c,
02222 0x232d, 0x232e, 0x232f, 0x2330, 0x2331, 0x2332, 0x2333, 0x2334,
02223 0x2335, 0x2336, 0x2337, 0x2338, 0x2339, 0x233a, 0x233b, 0x233c,
02224 0x233d, 0x233e, 0x233f, 0x2340, 0x2341, 0x2342, 0x2343, 0x2344,
02225 0x2345, 0x2346, 0x2347, 0x2348, 0x2349, 0x234a, 0x234b, 0x234c,
02226 0x234d, 0x234e, 0x234f, 0x2350, 0x2351, 0x2352, 0x2353, 0x2354,
02227 0x2355, 0x2356, 0x2357, 0x2358, 0x2359, 0x235a, 0x235b, 0x212c,
02228 0x235d, 0x235e, 0x235f, 0x2360, 0x2361, 0x2362, 0x2363, 0x2364,
02229 0x2365, 0x2366, 0x2367, 0x2368, 0x2369, 0x236a, 0x236b, 0x236c,
02230 0x236d, 0x236e, 0x236f, 0x2370, 0x2371, 0x2372, 0x2373, 0x2374,
02231 0x2375, 0x2376, 0x2377, 0x2378, 0x2379, 0x237a, 0x237b, 0x237c,
02232 0x237d, 0x2226, 0x214b, 0x214c, 0x217e, 0x237e, 0x214d, 0x235c,
02233 };
02234
02235 static const Summary16 ksc5601_uni2indx_page00[70] = {
02236 /* 0x0000 */
02237 { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 },
02238 { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x0000 },
02239 { 0, 0x0000 }, { 0, 0x0000 }, { 0, 0x2592 }, { 6, 0xf7df },
02240 { 20, 0x0040 }, { 21, 0xc181 }, { 26, 0x0040 }, { 27, 0x4181 },
02241 /* 0x0100 */
02242 { 31, 0x0000 }, { 31, 0x0002 }, { 32, 0x00c0 }, { 34, 0x810e },
02243 { 39, 0x0e07 }, { 45, 0x000c }, { 47, 0x00c0 }, { 49, 0x0000 },
02244 { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 },
02245 { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 },
02246 /* 0x0200 */
02247 { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 },
02248 { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 },
02249 { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 }, { 49, 0x0000 },
02250 { 49, 0x0080 }, { 50, 0x2f01 }, { 56, 0x0000 }, { 56, 0x0000 },
02251 /* 0x0300 */
02252 { 56, 0x0000 }, { 56, 0x0000 }, { 56, 0x0000 }, { 56, 0x0000 },
02253 { 56, 0x0000 }, { 56, 0x0000 }, { 56, 0x0000 }, { 56, 0x0000 },
02254 { 56, 0x0000 }, { 56, 0xffff }, { 71, 0x03fb }, { 80, 0xffff },
02255 { 95, 0x03fb }, { 104, 0x0000 }, { 104, 0x0000 }, { 104, 0x0000 },
02256 /* 0x0400 */
02257 { 104, 0x0002 }, { 105, 0xffff }, { 121, 0xffff }, { 137, 0xffff },
02258 { 153, 0xffff }, { 169, 0x0002 },
02259 };
02260 static const Summary16 ksc5601_uni2indx_page20[103] = {
02261 /* 0x2000 */
02262 { 170, 0x0000 }, { 170, 0x3320 }, { 175, 0x0063 }, { 179, 0x080d },
02263 { 183, 0x0000 }, { 183, 0x0000 }, { 183, 0x0000 }, { 183, 0x8010 },
02264 { 185, 0x001e }, { 189, 0x0000 }, { 189, 0x0000 }, { 189, 0x0000 },
02265 { 189, 0x0000 }, { 189, 0x0000 }, { 189, 0x0000 }, { 189, 0x0000 },
02266 /* 0x2100 */
02267 { 189, 0x0208 }, { 191, 0x0048 }, { 193, 0x0846 }, { 197, 0x0000 },
02268 { 197, 0x0000 }, { 197, 0x7818 }, { 203, 0x03ff }, { 213, 0x03ff },
02269 { 223, 0x0000 }, { 223, 0x03ff }, { 233, 0x0000 }, { 233, 0x0000 },
02270 { 233, 0x0000 }, { 233, 0x0014 }, { 235, 0x0000 }, { 235, 0x0000 },
02271 /* 0x2200 */
02272 { 235, 0x898d }, { 242, 0x6402 }, { 246, 0x5fa1 }, { 255, 0x3030 },
02273 { 259, 0x0000 }, { 259, 0x0004 }, { 260, 0x0c33 }, { 266, 0x0000 },
02274 { 266, 0x00cc }, { 270, 0x0200 }, { 271, 0x0020 }, { 272, 0x0000 },
02275 { 272, 0x0000 }, { 272, 0x0000 }, { 272, 0x0000 }, { 272, 0x0000 },
02276 /* 0x2300 */
02277 { 272, 0x0000 }, { 272, 0x0004 }, { 273, 0x0000 }, { 273, 0x0000 },
02278 { 273, 0x0000 }, { 273, 0x0000 }, { 273, 0x0000 }, { 273, 0x0000 },
02279 { 273, 0x0000 }, { 273, 0x0000 }, { 273, 0x0000 }, { 273, 0x0000 },
02280 { 273, 0x0000 }, { 273, 0x0000 }, { 273, 0x0000 }, { 273, 0x0000 },
02281 /* 0x2400 */
02282 { 273, 0x0000 }, { 273, 0x0000 }, { 273, 0x0000 }, { 273, 0x0000 },
02283 { 273, 0x0000 }, { 273, 0x0000 }, { 273, 0x7fff }, { 288, 0xffff },
02284 { 300, 0x0007 }, { 303, 0xf000 }, { 307, 0xffff }, { 323, 0x003f },
02285 { 329, 0x0000 }, { 329, 0xffff }, { 345, 0x03ff }, { 355, 0x0000 },
02286 /* 0x2500 */
02287 { 355, 0xf00f }, { 363, 0xffff }, { 379, 0xffff }, { 395, 0xffff },
02288 { 411, 0xffff }, { 423, 0x0000 }, { 423, 0x0000 }, { 423, 0x0000 },
02289 { 423, 0x0000 }, { 423, 0x0004 }, { 424, 0x03fb }, { 433, 0x30cc },
02290 { 439, 0xc9c3 }, { 447, 0x0003 }, { 449, 0x0000 }, { 449, 0x0000 },
02291 /* 0x2600 */
02292 { 449, 0xc060 }, { 453, 0x5000 }, { 455, 0x0000 }, { 455, 0x0000 },
02293 { 455, 0x0005 }, { 457, 0x0000 }, { 457, 0x37bb },

```

```
02294 };
02295 static const Summary16 ksc5601_uni2indx_page30[62] = {
02296     /* 0x3000 */
02297     { 468, 0xff0f }, { 480, 0x003b }, { 485, 0x0000 }, { 485, 0x0000 },
02298     { 485, 0xffff }, { 500, 0xffff }, { 516, 0xffff }, { 532, 0xffff },
02299     { 548, 0xffff }, { 564, 0x000f }, { 568, 0xffff }, { 583, 0xffff },
02300     { 599, 0xffff }, { 615, 0xffff }, { 631, 0xffff }, { 647, 0x007f },
02301     /* 0x3100 */
02302     { 654, 0x0000 }, { 654, 0x0000 }, { 654, 0x0000 }, { 654, 0xffff },
02303     { 669, 0xffff }, { 685, 0xffff }, { 701, 0xffff }, { 717, 0xffff },
02304     { 733, 0x7fff }, { 748, 0x0000 }, { 748, 0x0000 }, { 748, 0x0000 },
02305     { 748, 0x0000 }, { 748, 0x0000 }, { 748, 0x0000 }, { 748, 0x0000 },
02306     /* 0x3200 */
02307     { 748, 0xffff }, { 764, 0x1fff }, { 777, 0x0000 }, { 777, 0x0000 },
02308     { 777, 0x0000 }, { 777, 0x0000 }, { 777, 0xffff }, { 793, 0x8fff },
02309     { 806, 0x0000 }, { 806, 0x0000 }, { 806, 0x0000 }, { 806, 0x0000 },
02310     { 806, 0x0000 }, { 806, 0x0000 }, { 806, 0x0000 }, { 806, 0x0000 },
02311     /* 0x3300 */
02312     { 806, 0x0000 }, { 806, 0x0000 }, { 806, 0x0000 }, { 806, 0x0000 },
02313     { 806, 0x0000 }, { 806, 0x0000 }, { 806, 0x0000 }, { 806, 0x0000 },
02314     { 806, 0xffff }, { 819, 0xffff }, { 835, 0xffff }, { 851, 0xffff },
02315     { 867, 0x87ff }, { 879, 0x3949 },
02316 };
02317 static const Summary16 ksc5601_uni2indx_page4e[1306] = {
02318     /* 0x4e00 */
02319     { 886, 0x2f8b }, { 895, 0x4372 }, { 902, 0x2000 }, { 903, 0x0b04 },
02320     { 907, 0xe82c }, { 914, 0xe340 }, { 920, 0x2800 }, { 922, 0x40c8 },
02321     { 926, 0x5944 }, { 932, 0x4937 }, { 940, 0x7976 }, { 950, 0x0440 },
02322     { 952, 0x2c93 }, { 959, 0xa3f0 }, { 967, 0x0038 }, { 970, 0x08c5 },
02323     /* 0x4f00 */
02324     { 975, 0xee02 }, { 982, 0x0003 }, { 984, 0x8000 }, { 985, 0x3550 },
02325     { 991, 0xe1c8 }, { 998, 0x1e23 }, { 1005, 0x8200 }, { 1007, 0xc449 },
02326     { 1013, 0xad5a }, { 1022, 0x2942 }, { 1027, 0xc000 }, { 1029, 0x8060 },
02327     { 1032, 0x461c }, { 1038, 0xa49a }, { 1045, 0xc003 }, { 1049, 0x052a },
02328     /* 0x5000 */
02329     { 1054, 0x2a44 }, { 1059, 0xd646 }, { 1067, 0x3dda }, { 1077, 0x0800 },
02330     { 1078, 0x8388 }, { 1083, 0x1420 }, { 1086, 0x0020 }, { 1087, 0x0170 },
02331     { 1091, 0x2021 }, { 1094, 0x0302 }, { 1097, 0x3000 }, { 1099, 0x40ac },
02332     { 1104, 0x8620 }, { 1108, 0x4462 }, { 1113, 0x20a0 }, { 1116, 0x8a00 },
02333     /* 0x5100 */
02334     { 1119, 0x0253 }, { 1124, 0x8004 }, { 1126, 0x0402 }, { 1128, 0x1484 },
02335     { 1132, 0x7bfb }, { 1145, 0x1004 }, { 1147, 0x7fa4 }, { 1157, 0x11e2 },
02336     { 1163, 0x2441 }, { 1167, 0x00a4 }, { 1170, 0x1421 }, { 1174, 0x20c0 },
02337     { 1177, 0x3a50 }, { 1183, 0x7000 }, { 1186, 0x0002 }, { 1187, 0x2743 },
02338     /* 0x5200 */
02339     { 1194, 0x45c9 }, { 1201, 0x2082 }, { 1204, 0x4630 }, { 1209, 0x0fc1 },
02340     { 1216, 0x3c88 }, { 1222, 0x2850 }, { 1226, 0x8602 }, { 1230, 0xa024 },
02341     { 1234, 0x2388 }, { 1239, 0x8806 }, { 1243, 0x0e19 }, { 1249, 0x4000 },
02342     { 1250, 0x22aa }, { 1256, 0xeb64 }, { 1265, 0x001c }, { 1268, 0xcd28 },
02343     /* 0x5300 */
02344     { 1275, 0xa120 }, { 1279, 0x02e1 }, { 1284, 0x840b }, { 1289, 0x8200 },
02345     { 1291, 0x279b }, { 1300, 0x549e }, { 1308, 0x8141 }, { 1312, 0xa0b3 },
02346     { 1319, 0x0010 }, { 1320, 0x8508 }, { 1324, 0x2061 }, { 1328, 0x0800 },
02347     { 1329, 0x2f08 }, { 1335, 0x08d0 }, { 1339, 0xbe3e }, { 1350, 0x010f },
02348     /* 0x5400 */
02349     { 1355, 0xf718 }, { 1364, 0xa803 }, { 1369, 0x0a41 }, { 1373, 0x5b08 },
02350     { 1379, 0x0504 }, { 1382, 0x0002 }, { 1383, 0x0500 }, { 1385, 0x382a },
02351     { 1391, 0x5041 }, { 1395, 0x0001 }, { 1396, 0x1910 }, { 1400, 0x2108 },
02352     { 1403, 0x0313 }, { 1408, 0x0000 }, { 1408, 0x6122 }, { 1413, 0x0404 },
02353     /* 0x5500 */
02354     { 1415, 0x40d0 }, { 1419, 0x1001 }, { 1421, 0x8000 }, { 1422, 0x4022 },
02355     { 1425, 0x8050 }, { 1428, 0x4048 }, { 1431, 0x0008 }, { 1432, 0x1000 },
02356     { 1433, 0x06d1 }, { 1439, 0x3700 }, { 1444, 0x5e80 }, { 1450, 0x0000 },
02357     { 1450, 0x00a0 }, { 1452, 0x9410 }, { 1456, 0x0018 }, { 1458, 0x6000 },
02358     /* 0x5600 */
02359     { 1460, 0x0240 }, { 1462, 0x0090 }, { 1464, 0x8000 }, { 1465, 0x0054 },
02360     { 1468, 0x0000 }, { 1468, 0x0008 }, { 1469, 0x0900 }, { 1471, 0x0010 },
02361     { 1472, 0x0040 }, { 1473, 0x0000 }, { 1473, 0x5020 }, { 1476, 0x1010 },
02362     { 1478, 0x2400 }, { 1480, 0x4c02 }, { 1484, 0x0001 }, { 1485, 0x0601 },
02363     /* 0x5700 */
02364     { 1488, 0x2918 }, { 1493, 0x814c }, { 1498, 0x2100 }, { 1500, 0x0801 },
02365     { 1502, 0x6485 }, { 1508, 0x0003 }, { 1510, 0x4452 }, { 1515, 0x1021 },
02366     { 1518, 0x0904 }, { 1521, 0x0008 }, { 1522, 0x000d }, { 1525, 0x0000 },
02367     { 1525, 0x4988 }, { 1530, 0x8000 }, { 1531, 0x0001 }, { 1532, 0x1691 },
02368     /* 0x5800 */
02369     { 1538, 0x0765 }, { 1545, 0x4000 }, { 1546, 0x8492 }, { 1551, 0x0433 },
02370     { 1556, 0x8c00 }, { 1559, 0x4592 }, { 1565, 0x0016 }, { 1568, 0x5220 },
02371     { 1572, 0x0228 }, { 1575, 0xd008 }, { 1579, 0x4300 }, { 1582, 0x4c08 },
02372     { 1586, 0x40a2 }, { 1590, 0xc32a }, { 1597, 0x9810 }, { 1601, 0x2e00 },
02373     /* 0x5900 */
02374     { 1605, 0x8000 }, { 1606, 0x1670 }, { 1612, 0x6e84 }, { 1619, 0x4082 },
02375     { 1622, 0xc390 }, { 1628, 0x04b3 }, { 1634, 0x7c85 }, { 1642, 0x2118 },
02376     { 1646, 0x041c }, { 1650, 0x02c8 }, { 1654, 0x1120 }, { 1657, 0x4a00 },
02377     { 1660, 0x0a48 }, { 1664, 0x361b }, { 1672, 0x5540 }, { 1677, 0x8900 },
02378     /* 0x5a00 */
02379     { 1680, 0x000a }, { 1682, 0x9902 }, { 1687, 0x0221 }, { 1690, 0x1040 },
02380     { 1692, 0x0242 }, { 1695, 0x0400 }, { 1696, 0x0044 }, { 1698, 0x0000 },
```

```

02381 { 1698, 0x0000 }, { 1698, 0x0c04 }, { 1701, 0x0010 }, { 1702, 0x0000 },
02382 { 1702, 0x1216 }, { 1707, 0x0000 }, { 1707, 0x0242 }, { 1710, 0x0000 },
02383 /* 0x5b00 */
02384 { 1710, 0x1a20 }, { 1714, 0x0040 }, { 1715, 0x0400 }, { 1716, 0x0000 },
02385 { 1716, 0x0009 }, { 1718, 0xb5b3 }, { 1728, 0x0a18 }, { 1732, 0x1523 },
02386 { 1738, 0x9ba0 }, { 1745, 0x1fe8 }, { 1754, 0x507c }, { 1761, 0x8379 },
02387 { 1769, 0x10fd }, { 1777, 0xc09d }, { 1784, 0xdbf6 }, { 1796, 0x0560 },
02388 /* 0x5c00 */
02389 { 1800, 0xef92 }, { 1810, 0x0242 }, { 1813, 0x0110 }, { 1815, 0xdf02 },
02390 { 1823, 0x6961 }, { 1830, 0x0822 }, { 1833, 0x9035 }, { 1839, 0x0202 },
02391 { 1841, 0x0000 }, { 1841, 0x0003 }, { 1843, 0x1a02 }, { 1847, 0x45aa },
02392 { 1854, 0x0001 }, { 1855, 0x0200 }, { 1856, 0x8101 }, { 1859, 0x2851 },
02393 /* 0x5d00 */
02394 { 1864, 0x6080 }, { 1867, 0x02d2 }, { 1872, 0x0280 }, { 1874, 0x0000 },
02395 { 1874, 0x1800 }, { 1876, 0x0001 }, { 1877, 0x9200 }, { 1880, 0x0000 },
02396 { 1880, 0x0880 }, { 1882, 0x2000 }, { 1883, 0x0405 }, { 1886, 0x3500 },
02397 { 1890, 0x2000 }, { 1891, 0x6044 }, { 1895, 0x49e6 }, { 1903, 0x609e },
02398 /* 0x5e00 */
02399 { 1910, 0x104c }, { 1914, 0x2a42 }, { 1919, 0x2820 }, { 1922, 0xa148 },
02400 { 1927, 0x10b1 }, { 1932, 0x8020 }, { 1934, 0x000e }, { 1937, 0x7b9c },
02401 { 1947, 0x8490 }, { 1951, 0x14a0 }, { 1955, 0x28c1 }, { 1960, 0x41e0 },
02402 { 1965, 0x0704 }, { 1969, 0x8c49 }, { 1975, 0x100d }, { 1979, 0x0cc8 },
02403 /* 0x5f00 */
02404 { 1984, 0x8412 }, { 1988, 0x89ba }, { 1996, 0x02c0 }, { 1999, 0x1422 },
02405 { 2003, 0x5500 }, { 2007, 0x0ac0 }, { 2011, 0x3ec4 }, { 2019, 0x9283 },
02406 { 2025, 0x1ca3 }, { 2032, 0x4387 }, { 2039, 0x4703 }, { 2045, 0x22a0 },
02407 { 2049, 0x3028 }, { 2053, 0x03c0 }, { 2057, 0x0801 }, { 2059, 0xa020 },
02408 /* 0x6000 */
02409 { 2062, 0x8000 }, { 2063, 0x3044 }, { 2067, 0x85a3 }, { 2074, 0x0000 },
02410 { 2074, 0x200e }, { 2078, 0x2225 }, { 2083, 0xb73c }, { 2093, 0x0001 },
02411 { 2094, 0x3220 }, { 2098, 0x8c50 }, { 2103, 0x0099 }, { 2107, 0x315d },
02412 { 2115, 0x00a0 }, { 2117, 0x9402 }, { 2121, 0x0003 }, { 2123, 0x0e4b },
02413 /* 0x6100 */
02414 { 2130, 0xe342 }, { 2137, 0x8c20 }, { 2141, 0x0080 }, { 2142, 0xd091 },
02415 { 2148, 0x1d94 }, { 2155, 0xa328 }, { 2161, 0x499c }, { 2168, 0x60c1 },
02416 { 2173, 0x4406 }, { 2177, 0x0713 }, { 2183, 0x5a90 }, { 2189, 0x4444 },
02417 { 2193, 0x0f88 }, { 2199, 0x0000 }, { 2199, 0x0040 }, { 2200, 0x95c4 },
02418 /* 0x6200 */
02419 { 2207, 0x7581 }, { 2214, 0x8447 }, { 2220, 0x4402 }, { 2223, 0xc053 },
02420 { 2229, 0x2b83 }, { 2236, 0x0108 }, { 2238, 0x4000 }, { 2239, 0x9242 },
02421 { 2244, 0x0611 }, { 2248, 0x09a6 }, { 2254, 0x0800 }, { 2255, 0x3222 },
02422 { 2260, 0xb384 }, { 2267, 0x1bdd }, { 2277, 0xf000 }, { 2281, 0xc08a },
02423 /* 0x6300 */
02424 { 2286, 0x0282 }, { 2289, 0x0002 }, { 2290, 0x8800 }, { 2292, 0x6c00 },
02425 { 2296, 0x9200 }, { 2299, 0x0021 }, { 2301, 0x4180 }, { 2304, 0x8c84 },
02426 { 2309, 0x1308 }, { 2313, 0x0944 }, { 2317, 0x07a7 }, { 2325, 0x0000 },
02427 { 2325, 0x8051 }, { 2329, 0x0c41 }, { 2333, 0x6002 }, { 2336, 0x00d0 },
02428 /* 0x6400 */
02429 { 2339, 0xa000 }, { 2341, 0x10d0 }, { 2345, 0x3004 }, { 2348, 0x4400 },
02430 { 2350, 0x0000 }, { 2350, 0x0100 }, { 2351, 0x8201 }, { 2354, 0x0700 },
02431 { 2357, 0x0100 }, { 2358, 0x440e }, { 2363, 0x6830 }, { 2368, 0x0805 },
02432 { 2371, 0x64b2 }, { 2378, 0x0514 }, { 2382, 0x10e6 }, { 2388, 0x4414 },
02433 /* 0x6500 */
02434 { 2392, 0x0011 }, { 2394, 0x2100 }, { 2396, 0x9c08 }, { 2401, 0xc0c0 },
02435 { 2408, 0xe120 }, { 2413, 0x40c2 }, { 2417, 0x304c }, { 2422, 0x41b4 },
02436 { 2428, 0x10ac }, { 2433, 0x9a83 }, { 2440, 0x98b2 }, { 2447, 0x3281 },
02437 { 2452, 0x9822 }, { 2457, 0x0084 }, { 2459, 0x3369 }, { 2467, 0xb0c12 },
02438 /* 0x6600 */
02439 { 2474, 0xd6c0 }, { 2481, 0xc03b }, { 2488, 0xala1 }, { 2494, 0x0c53 },
02440 { 2500, 0x8a1e }, { 2507, 0xea00 }, { 2512, 0xcbf0 }, { 2521, 0x05d8 },
02441 { 2527, 0x4390 }, { 2532, 0x21c3 }, { 2538, 0x4805 }, { 2542, 0x4a1c },
02442 { 2548, 0x02d0 }, { 2552, 0x3240 }, { 2556, 0x0041 }, { 2558, 0xd79d },
02443 /* 0x6700 */
02444 { 2569, 0x2b09 }, { 2575, 0xe8b0 }, { 2582, 0x7dc0 }, { 2590, 0x2452 },
02445 { 2595, 0xc240 }, { 2599, 0xd04b }, { 2606, 0xa000 }, { 2608, 0xc8ab },
02446 { 2616, 0x8a80 }, { 2620, 0x34a9 }, { 2627, 0x8000 }, { 2628, 0x41c9 },
02447 { 2634, 0x8010 }, { 2636, 0x241f }, { 2643, 0x9200 }, { 2646, 0x487b },
02448 /* 0x6800 */
02449 { 2654, 0x0000 }, { 2654, 0x00cc }, { 2658, 0x8406 }, { 2662, 0x3300 },
02450 { 2666, 0x410f }, { 2672, 0x001b }, { 2676, 0x2000 }, { 2677, 0x8040 },
02451 { 2679, 0x8022 }, { 2682, 0xa098 }, { 2687, 0xa186 }, { 2693, 0x006b },
02452 { 2698, 0x2a30 }, { 2703, 0x85a4 }, { 2709, 0x4181 }, { 2713, 0x0604 },
02453 /* 0x6900 */
02454 { 2716, 0x6021 }, { 2720, 0x0004 }, { 2721, 0x0080 }, { 2722, 0xa001 },
02455 { 2725, 0x0400 }, { 2726, 0x46b8 }, { 2733, 0xe90f }, { 2742, 0x03a0 },
02456 { 2746, 0x0000 }, { 2746, 0x1820 }, { 2749, 0x40a0 }, { 2752, 0x0810 },
02457 { 2754, 0x380a }, { 2759, 0x0001 }, { 2760, 0x0500 }, { 2762, 0xa800 },
02458 /* 0x6a00 */
02459 { 2765, 0x0404 }, { 2767, 0xc28a }, { 2773, 0x000a }, { 2775, 0x2720 },
02460 { 2780, 0x0910 }, { 2783, 0x830c }, { 2788, 0x0802 }, { 2790, 0x0000 },
02461 { 2790, 0x6211 }, { 2795, 0x1080 }, { 2797, 0x000c }, { 2799, 0x0808 },
02462 { 2801, 0x000c }, { 2803, 0x0c08 }, { 2806, 0x0000 }, { 2806, 0x0840 },
02463 /* 0x6b00 */
02464 { 2808, 0x1410 }, { 2811, 0x0044 }, { 2813, 0x000b }, { 2816, 0x6404 },
02465 { 2820, 0x50c0 }, { 2824, 0x8001 }, { 2826, 0x047e }, { 2833, 0x8984 },
02466 { 2838, 0x0658 }, { 2843, 0x4140 }, { 2846, 0xc000 }, { 2848, 0x94a4 },
02467 { 2854, 0xa862 }, { 2860, 0x09dc }, { 2867, 0x1800 }, { 2869, 0x0000 },

```

```
02468 /* 0x6c00 */
02469 { 2869, 0x8100 }, { 2871, 0x000a }, { 2873, 0x0008 }, { 2874, 0x4190 },
02470 { 2878, 0x4007 }, { 2882, 0xe4a1 }, { 2889, 0x2501 }, { 2893, 0x6445 },
02471 { 2899, 0x11ee }, { 2907, 0x0e7d }, { 2916, 0x4800 }, { 2918, 0xfb08 },
02472 { 2926, 0x1616 }, { 2932, 0x08a8 }, { 2936, 0xc92e }, { 2944, 0x0009 },
02473 /* 0x6d00 */
02474 { 2946, 0x1800 }, { 2948, 0x4a82 }, { 2953, 0x06a0 }, { 2957, 0x6b64 },
02475 { 2965, 0x0002 }, { 2966, 0x1600 }, { 2969, 0x5648 }, { 2975, 0x8390 },
02476 { 2980, 0x73a0 }, { 2987, 0x002a }, { 2990, 0x8000 }, { 2991, 0x002a },
02477 { 2993, 0x88f9 }, { 3001, 0x4702 }, { 3006, 0x4d02 }, { 3011, 0x0faa },
02478 /* 0x6e00 */
02479 { 3019, 0x0000 }, { 3019, 0x8e80 }, { 3024, 0xb87b }, { 3034, 0x7554 },
02480 { 3042, 0x2418 }, { 3046, 0xd940 }, { 3052, 0xc880 }, { 3056, 0x040c },
02481 { 3059, 0x0000 }, { 3059, 0xb041 }, { 3064, 0x8c24 }, { 3069, 0x0442 },
02482 { 3072, 0x5a34 }, { 3079, 0x001a }, { 3082, 0x8000 }, { 3083, 0xc110 },
02483 /* 0x6f00 */
02484 { 3087, 0x8046 }, { 3091, 0x0032 }, { 3094, 0x180d }, { 3099, 0x8106 },
02485 { 3103, 0x0002 }, { 3104, 0xcd92 }, { 3112, 0x6014 }, { 3116, 0x7401 },
02486 { 3121, 0x6112 }, { 3126, 0x0091 }, { 3129, 0xc098 }, { 3134, 0x420a },
02487 { 3138, 0x040f }, { 3143, 0x8420 }, { 3146, 0x9a13 }, { 3153, 0x4002 },
02488 /* 0x7000 */
02489 { 3155, 0x8a62 }, { 3161, 0xfd22 }, { 3170, 0x8188 }, { 3174, 0x4080 },
02490 { 3176, 0x1000 }, { 3177, 0x2103 }, { 3181, 0x0808 }, { 3183, 0x3101 },
02491 { 3187, 0x4420 }, { 3190, 0x0704 }, { 3194, 0xb812 }, { 3200, 0x0388 },
02492 { 3204, 0x8900 }, { 3207, 0xa300 }, { 3211, 0x0000 }, { 3211, 0x2202 },
02493 /* 0x7100 */
02494 { 3214, 0x1210 }, { 3217, 0x4600 }, { 3220, 0x0042 }, { 3222, 0x0041 },
02495 { 3224, 0x5680 }, { 3229, 0x5241 }, { 3234, 0x52f0 }, { 3241, 0x2000 },
02496 { 3242, 0x8610 }, { 3246, 0x8214 }, { 3250, 0x1004 }, { 3252, 0x4602 },
02497 { 3256, 0x430a }, { 3261, 0x8035 }, { 3266, 0x60e0 }, { 3271, 0xd800 },
02498 /* 0x7200 */
02499 { 3275, 0x0041 }, { 3277, 0x0801 }, { 3279, 0x3400 }, { 3282, 0x6c65 },
02500 { 3290, 0x11c1 }, { 3295, 0xab04 }, { 3301, 0x0286 }, { 3305, 0x2204 },
02501 { 3308, 0x0003 }, { 3310, 0x0000 }, { 3310, 0x9084 }, { 3314, 0x0000 },
02502 { 3314, 0x4015 }, { 3318, 0x0281 }, { 3321, 0x0202 }, { 3323, 0x3300 },
02503 /* 0x7300 */
02504 { 3327, 0x0400 }, { 3328, 0x3840 }, { 3332, 0x0e20 }, { 3336, 0xc0c0 },
02505 { 3340, 0x0030 }, { 3342, 0x0085 }, { 3345, 0x0500 }, { 3347, 0x0d25 },
02506 { 3353, 0x4ad0 }, { 3359, 0x81d0 }, { 3364, 0x2280 }, { 3367, 0x020c },
02507 { 3370, 0xb605 }, { 3377, 0x6240 }, { 3381, 0x2679 }, { 3389, 0x6280 },
02508 /* 0x7400 */
02509 { 3393, 0x02ea }, { 3399, 0x0808 }, { 3401, 0xdd67 }, { 3412, 0x8579 },
02510 { 3420, 0x081b }, { 3425, 0xdea0 }, { 3433, 0x8735 }, { 3441, 0x4000 },
02511 { 3442, 0x0a8c }, { 3447, 0xd100 }, { 3451, 0x05aa }, { 3457, 0xa225 },
02512 { 3463, 0x8440 }, { 3466, 0x1510 }, { 3470, 0x404d }, { 3475, 0x0080 },
02513 /* 0x7500 */
02514 { 3476, 0x0012 }, { 3478, 0x8d22 }, { 3484, 0x1968 }, { 3490, 0x058f },
02515 { 3497, 0x9080 }, { 3500, 0x3a1a }, { 3507, 0x8464 }, { 3512, 0x8561 },
02516 { 3518, 0xcc00 }, { 3524, 0x2002 }, { 3526, 0x0820 }, { 3528, 0x732e },
02517 { 3537, 0x20a4 }, { 3541, 0x0b34 }, { 3547, 0x0004 }, { 3548, 0x1415 },
02518 /* 0x7600 */
02519 { 3553, 0x2001 }, { 3555, 0x8200 }, { 3557, 0x0057 }, { 3562, 0x0800 },
02520 { 3563, 0x5004 }, { 3566, 0x0044 }, { 3568, 0x1212 }, { 3572, 0x7905 },
02521 { 3579, 0x40d0 }, { 3583, 0x0009 }, { 3585, 0x4000 }, { 3586, 0x8400 },
02522 { 3588, 0x054c }, { 3593, 0xd844 }, { 3599, 0x409a }, { 3604, 0x5114 },
02523 /* 0x7700 */
02524 { 3609, 0x0b12 }, { 3614, 0x4000 }, { 3615, 0x0201 }, { 3617, 0x1580 },
02525 { 3621, 0x2001 }, { 3623, 0x0800 }, { 3624, 0x084a }, { 3628, 0xc200 },
02526 { 3631, 0x0800 }, { 3632, 0x4002 }, { 3634, 0x3020 }, { 3637, 0x9809 },
02527 { 3642, 0x0000 }, { 3642, 0x1880 }, { 3645, 0xe22c }, { 3652, 0x0008 },
02528 /* 0x7800 */
02529 { 3653, 0x0004 }, { 3654, 0x0004 }, { 3655, 0x10e0 }, { 3659, 0x0014 },
02530 { 3661, 0x8020 }, { 3663, 0x2000 }, { 3664, 0x9800 }, { 3667, 0x1000 },
02531 { 3668, 0x7082 }, { 3673, 0x0082 }, { 3675, 0x0288 }, { 3678, 0x1c00 },
02532 { 3681, 0x4c22 }, { 3686, 0x0001 }, { 3687, 0x9100 }, { 3690, 0x0820 },
02533 /* 0x7900 */
02534 { 3692, 0x4002 }, { 3694, 0x0040 }, { 3695, 0x1c00 }, { 3698, 0x4400 },
02535 { 3700, 0x0383 }, { 3705, 0x7cc1 }, { 3713, 0x2121 }, { 3717, 0x8400 },
02536 { 3719, 0xe002 }, { 3723, 0x0002 }, { 3724, 0x44c0 }, { 3728, 0xe20a },
02537 { 3734, 0x0e03 }, { 3739, 0x8126 }, { 3744, 0x02d0 }, { 3748, 0x0800 },
02538 /* 0x7a00 */
02539 { 3749, 0x2921 }, { 3754, 0x9690 }, { 3760, 0x4001 }, { 3762, 0xb8c2 },
02540 { 3769, 0x6241 }, { 3774, 0x0080 }, { 3775, 0x0a06 }, { 3779, 0xa651 },
02541 { 3786, 0x0112 }, { 3789, 0x812c }, { 3794, 0xc600 }, { 3798, 0x0400 },
02542 { 3799, 0x0cb0 }, { 3804, 0xa280 }, { 3808, 0xa429 }, { 3814, 0x8640 },
02543 /* 0x7b00 */
02544 { 3818, 0x8000 }, { 3819, 0x4a02 }, { 3823, 0x3041 }, { 3827, 0x0200 },
02545 { 3828, 0xba40 }, { 3834, 0x0057 }, { 3839, 0x5001 }, { 3842, 0x2020 },
02546 { 3844, 0x8880 }, { 3847, 0x24b0 }, { 3852, 0x2002 }, { 3854, 0x0112 },
02547 { 3857, 0x02d3 }, { 3863, 0x0004 }, { 3864, 0x0211 }, { 3867, 0x0000 },
02548 /* 0x7c00 */
02549 { 3867, 0x0080 }, { 3868, 0x4004 }, { 3870, 0x0c82 }, { 3874, 0xe000 },
02550 { 3877, 0x3008 }, { 3880, 0x0000 }, { 3880, 0x1011 }, { 3883, 0x0008 },
02551 { 3884, 0x0208 }, { 3886, 0x81a4 }, { 3891, 0x40a0 }, { 3894, 0x420e },
02552 { 3899, 0x0400 }, { 3900, 0xc040 }, { 3903, 0x0081 }, { 3905, 0x4800 },
02553 /* 0x7d00 */
02554 { 3907, 0x2df5 }, { 3917, 0x0f91 }, { 3924, 0xd807 }, { 3931, 0x0629 },
```

```

02555 { 3936, 0x007c }, { 3941, 0x4001 }, { 3943, 0x4546 }, { 3949, 0x824e },
02556 { 3955, 0xc000 }, { 3957, 0x1008 }, { 3959, 0x3005 }, { 3963, 0xed36 },
02557 { 3973, 0x0c80 }, { 3976, 0x6540 }, { 3981, 0x930b }, { 3988, 0x0810 },
02558 /* 0x7e00 */
02559 { 3990, 0x0600 }, { 3992, 0xe820 }, { 3997, 0xc80a }, { 4002, 0x6082 },
02560 { 4006, 0x00ca }, { 4010, 0x4034 }, { 4014, 0x2e02 }, { 4019, 0x1201 },
02561 { 4022, 0x9004 }, { 4025, 0x1948 }, { 4030, 0x0000 }, { 4030, 0x0000 },
02562 { 4030, 0x0000 }, { 4030, 0x0000 }, { 4030, 0x0000 }, { 4030, 0x0000 },
02563 /* 0x7f00 */
02564 { 4030, 0x0000 }, { 4030, 0x0000 }, { 4030, 0x0000 }, { 4030, 0x0540 },
02565 { 4033, 0x1000 }, { 4034, 0x0031 }, { 4037, 0x4c00 }, { 4040, 0x02a5 },
02566 { 4045, 0x5520 }, { 4050, 0x4410 }, { 4053, 0x0310 }, { 4056, 0x2304 },
02567 { 4060, 0x5422 }, { 4065, 0x8034 }, { 4069, 0x0a03 }, { 4073, 0x1201 },
02568 /* 0x8000 */
02569 { 4076, 0x126b }, { 4083, 0x01a1 }, { 4087, 0x2000 }, { 4088, 0xa048 },
02570 { 4092, 0x0448 }, { 4095, 0x4540 }, { 4099, 0x8000 }, { 4100, 0xe08d },
02571 { 4107, 0x1af0 }, { 4114, 0x2840 }, { 4117, 0x8626 }, { 4123, 0x0416 },
02572 { 4127, 0x5018 }, { 4131, 0x4c00 }, { 4134, 0x0032 }, { 4137, 0x2112 },
02573 /* 0x8100 */
02574 { 4141, 0x05e4 }, { 4147, 0x0d00 }, { 4150, 0x8a08 }, { 4154, 0x4200 },
02575 { 4156, 0x4800 }, { 4158, 0x0033 }, { 4162, 0x0860 }, { 4165, 0x8703 },
02576 { 4171, 0x8501 }, { 4175, 0x3400 }, { 4178, 0x0109 }, { 4181, 0xe428 },
02577 { 4187, 0x2045 }, { 4191, 0x8100 }, { 4193, 0x25a8 }, { 4199, 0x5c18 },
02578 /* 0x8200 */
02579 { 4205, 0x35a0 }, { 4211, 0xd804 }, { 4216, 0x1c02 }, { 4220, 0x02e0 },
02580 { 4224, 0x00a1 }, { 4227, 0x0200 }, { 4228, 0xc050 }, { 4232, 0x4146 },
02581 { 4237, 0x6800 }, { 4240, 0xa604 }, { 4245, 0xf260 }, { 4252, 0xbb8a },
02582 { 4261, 0x0000 }, { 4261, 0xc8b6 }, { 4269, 0x00e2 }, { 4273, 0x6002 },
02583 /* 0x8300 */
02584 { 4276, 0x023e }, { 4282, 0x0080 }, { 4283, 0x8900 }, { 4286, 0x0372 },
02585 { 4292, 0x8681 }, { 4297, 0x0006 }, { 4299, 0x0000 }, { 4299, 0x0888 },
02586 { 4302, 0x4600 }, { 4305, 0x4140 }, { 4308, 0x0e04 }, { 4312, 0x2000 },
02587 { 4313, 0x1622 }, { 4318, 0x1048 }, { 4321, 0x8a00 }, { 4324, 0x2217 },
02588 /* 0x8400 */
02589 { 4330, 0x7418 }, { 4336, 0x0000 }, { 4336, 0x1200 }, { 4338, 0x2102 },
02590 { 4341, 0x0200 }, { 4342, 0x0880 }, { 4344, 0x984a }, { 4350, 0x0420 },
02591 { 4352, 0x0000 }, { 4352, 0x1211 }, { 4356, 0x0002 }, { 4357, 0x9904 },
02592 { 4362, 0x2a55 }, { 4369, 0x0402 }, { 4371, 0x5000 }, { 4373, 0x1010 },
02593 /* 0x8500 */
02594 { 4375, 0x0000 }, { 4375, 0x459a }, { 4382, 0xb02a }, { 4388, 0xa000 },
02595 { 4390, 0x420a }, { 4394, 0x0208 }, { 4396, 0x2708 }, { 4401, 0x0000 },
02596 { 4401, 0x8090 }, { 4404, 0x0812 }, { 4407, 0x8740 }, { 4412, 0x0401 },
02597 { 4414, 0xe202 }, { 4419, 0x3020 }, { 4422, 0x0630 }, { 4426, 0x8c80 },
02598 /* 0x8600 */
02599 { 4430, 0x04c4 }, { 4434, 0x04c0 }, { 4437, 0x2000 }, { 4438, 0x8000 },
02600 { 4439, 0x4000 }, { 4440, 0xd831 }, { 4447, 0x0080 }, { 4448, 0x0200 },
02601 { 4449, 0x1400 }, { 4451, 0x0008 }, { 4452, 0x0218 }, { 4455, 0x0000 },
02602 { 4455, 0x0880 }, { 4457, 0x8a10 }, { 4461, 0x2010 }, { 4463, 0x4000 },
02603 /* 0x8700 */
02604 { 4464, 0x010d }, { 4468, 0x1500 }, { 4471, 0x0000 }, { 4471, 0x0000 },
02605 { 4471, 0x4000 }, { 4472, 0x80a0 }, { 4475, 0x0140 }, { 4477, 0x0150 },
02606 { 4480, 0x2004 }, { 4482, 0x8000 }, { 4483, 0x0004 }, { 4484, 0x0408 },
02607 { 4486, 0x0010 }, { 4487, 0x0000 }, { 4487, 0x9001 }, { 4490, 0x4a04 },
02608 /* 0x8800 */
02609 { 4494, 0x0020 }, { 4495, 0x8000 }, { 4496, 0x000c }, { 4498, 0x0842 },
02610 { 4501, 0x3041 }, { 4505, 0x2a8c }, { 4511, 0x090e }, { 4516, 0xc085 },
02611 { 4521, 0x2906 }, { 4526, 0x40c4 }, { 4530, 0x0800 }, { 4531, 0x0010 },
02612 { 4532, 0x8006 }, { 4535, 0xb230 }, { 4541, 0x0102 }, { 4543, 0x2138 },
02613 /* 0x8900 */
02614 { 4548, 0x0080 }, { 4549, 0x030d }, { 4554, 0x0420 }, { 4556, 0x0940 },
02615 { 4559, 0x0012 }, { 4561, 0x8000 }, { 4562, 0x0410 }, { 4564, 0x8004 },
02616 { 4566, 0x88ca }, { 4572, 0x0048 }, { 4574, 0x0602 }, { 4577, 0x2404 },
02617 { 4580, 0x0001 }, { 4581, 0x0004 }, { 4582, 0x0008 }, { 4583, 0x0110 },
02618 /* 0x8a00 */
02619 { 4585, 0x550d }, { 4592, 0xa9c8 }, { 4599, 0x2428 }, { 4603, 0x0c52 },
02620 { 4608, 0x0000 }, { 4608, 0x4831 }, { 4613, 0x624d }, { 4620, 0x022f },
02621 { 4626, 0x30a0 }, { 4630, 0x4128 }, { 4634, 0x057b }, { 4642, 0xd205 },
02622 { 4648, 0xa894 }, { 4654, 0x1844 }, { 4658, 0x6cc2 }, { 4665, 0x45c2 },
02623 /* 0x8b00 */
02624 { 4671, 0x4017 }, { 4676, 0x2ed1 }, { 4684, 0x1901 }, { 4688, 0x0208 },
02625 { 4690, 0xc202 }, { 4694, 0x1500 }, { 4697, 0x9040 }, { 4700, 0x2091 },
02626 { 4704, 0x0401 }, { 4706, 0x044d }, { 4711, 0x0000 }, { 4711, 0x0000 },
02627 { 4711, 0x0000 }, { 4711, 0x0000 }, { 4711, 0x0000 }, { 4711, 0x0000 },
02628 /* 0x8c00 */
02629 { 4711, 0x0000 }, { 4711, 0x0000 }, { 4711, 0x0000 }, { 4711, 0x8080 },
02630 { 4713, 0x1542 }, { 4718, 0x0420 }, { 4720, 0x0c02 }, { 4723, 0x0600 },
02631 { 4725, 0x1404 }, { 4728, 0x6000 }, { 4730, 0x9f87 }, { 4740, 0xb9d9 },
02632 { 4750, 0x059f }, { 4758, 0x540a }, { 4763, 0x245d }, { 4770, 0x3810 },
02633 /* 0x8d00 */
02634 { 4774, 0x25b0 }, { 4780, 0x0048 }, { 4782, 0x0000 }, { 4782, 0x0000 },
02635 { 4782, 0x0000 }, { 4782, 0x0000 }, { 4782, 0x0850 }, { 4785, 0x0099 },
02636 { 4789, 0x0420 }, { 4791, 0x0200 }, { 4792, 0x0108 }, { 4794, 0x4408 },
02637 { 4797, 0x9840 }, { 4801, 0x2800 }, { 4803, 0x810a }, { 4807, 0x0008 },
02638 /* 0x8e00 */
02639 { 4808, 0x8400 }, { 4810, 0x4001 }, { 4812, 0x0400 }, { 4813, 0x0021 },
02640 { 4815, 0x0794 }, { 4821, 0x8200 }, { 4823, 0x0001 }, { 4824, 0x0050 },
02641 { 4826, 0x2482 }, { 4830, 0x0000 }, { 4830, 0x1c00 }, { 4833, 0x0000 },

```



```
02642 { 4833, 0x3c01 }, { 4838, 0x8004 }, { 4840, 0x0800 }, { 4841, 0x4900 },
02643 /* 0x8f00 */
02644 { 4844, 0x0228 }, { 4847, 0xf83c }, { 4856, 0x86c0 }, { 4861, 0xcb08 },
02645 { 4867, 0x6230 }, { 4872, 0xa000 }, { 4874, 0x0004 }, { 4875, 0x0000 },
02646 { 4875, 0x0000 }, { 4875, 0x1800 }, { 4877, 0xa148 }, { 4882, 0x0007 },
02647 { 4885, 0x4024 }, { 4888, 0x0012 }, { 4890, 0x2c40 }, { 4894, 0x2285 },
02648 /* 0x9000 */
02649 { 4899, 0xa96f }, { 4909, 0xe6b3 }, { 4919, 0x400f }, { 4924, 0x5126 },
02650 { 4930, 0x6c86 }, { 4937, 0x723b }, { 4946, 0xe20b }, { 4953, 0xb5a4 },
02651 { 4961, 0x859f }, { 4970, 0x0222 }, { 4973, 0x854c }, { 4979, 0x0123 },
02652 { 4983, 0x0402 }, { 4985, 0x4000 }, { 4986, 0x2102 }, { 4989, 0x2020 },
02653 /* 0x9100 */
02654 { 4991, 0x0004 }, { 4992, 0x0224 }, { 4995, 0x2080 }, { 4997, 0x0004 },
02655 { 4998, 0x7e00 }, { 5004, 0x0004 }, { 5005, 0x1604 }, { 5009, 0x01a0 },
02656 { 5012, 0x2a80 }, { 5016, 0x1004 }, { 5018, 0xd800 }, { 5022, 0x0032 },
02657 { 5025, 0xfa81 }, { 5033, 0x3183 }, { 5039, 0x0488 }, { 5042, 0x0020 },
02658 /* 0x9200 */
02659 { 5043, 0x2000 }, { 5044, 0x4087 }, { 5049, 0x0000 }, { 5049, 0x8410 },
02660 { 5052, 0x0221 }, { 5055, 0x4880 }, { 5058, 0x0074 }, { 5062, 0x0000 },
02661 { 5062, 0x0029 }, { 5065, 0x114a }, { 5070, 0x0000 }, { 5070, 0x02c8 },
02662 { 5074, 0x9000 }, { 5076, 0x0004 }, { 5077, 0x0410 }, { 5079, 0x1100 },
02663 /* 0x9300 */
02664 { 5081, 0x0010 }, { 5082, 0xc501 }, { 5087, 0xc957 }, { 5096, 0x0000 },
02665 { 5096, 0x2d00 }, { 5100, 0x0810 }, { 5102, 0x4000 }, { 5103, 0x5020 },
02666 { 5106, 0x1000 }, { 5107, 0x0450 }, { 5110, 0x3088 }, { 5114, 0x0001 },
02667 { 5115, 0x0008 }, { 5116, 0x4002 }, { 5118, 0x0012 }, { 5120, 0x0040 },
02668 /* 0x9400 */
02669 { 5121, 0x0010 }, { 5122, 0x0100 }, { 5123, 0x0820 }, { 5125, 0x0120 },
02670 { 5127, 0x0010 }, { 5128, 0x0806 }, { 5131, 0x0000 }, { 5131, 0xa000 },
02671 { 5133, 0x0000 }, { 5133, 0x0000 }, { 5133, 0x0000 }, { 5133, 0x0000 },
02672 { 5133, 0x0000 }, { 5133, 0x0000 }, { 5133, 0x0000 }, { 5133, 0x0000 },
02673 /* 0x9500 */
02674 { 5133, 0x0000 }, { 5133, 0x0000 }, { 5133, 0x0000 }, { 5133, 0x0000 },
02675 { 5133, 0x0000 }, { 5133, 0x0000 }, { 5133, 0x0000 }, { 5133, 0x0080 },
02676 { 5134, 0x8a09 }, { 5139, 0x011e }, { 5144, 0x2138 }, { 5149, 0x1802 },
02677 { 5152, 0x0480 }, { 5154, 0x1070 }, { 5158, 0x0006 }, { 5160, 0x0000 },
02678 /* 0x9600 */
02679 { 5160, 0x0000 }, { 5160, 0x1000 }, { 5161, 0x4402 }, { 5164, 0x8804 },
02680 { 5167, 0x3815 }, { 5173, 0xf801 }, { 5179, 0x041c }, { 5183, 0x21e9 },
02681 { 5190, 0x6c60 }, { 5196, 0x1b30 }, { 5202, 0x0588 }, { 5206, 0x0882 },
02682 { 5209, 0x7af3 }, { 5220, 0x1a60 }, { 5225, 0x870c }, { 5231, 0x0ac5 },
02683 /* 0x9700 */
02684 { 5237, 0x00c1 }, { 5240, 0x524a }, { 5246, 0x0080 }, { 5247, 0x2205 },
02685 { 5251, 0x0114 }, { 5254, 0x5042 }, { 5258, 0x2206 }, { 5262, 0x0490 },
02686 { 5265, 0xa800 }, { 5268, 0x0000 }, { 5268, 0x2901 }, { 5272, 0x0000 },
02687 { 5272, 0x0840 }, { 5274, 0x1008 }, { 5276, 0x0000 }, { 5276, 0x8848 },
02688 /* 0x9800 */
02689 { 5280, 0x156f }, { 5289, 0x018f }, { 5295, 0x2000 }, { 5296, 0x0b01 },
02690 { 5300, 0x7040 }, { 5304, 0x4510 }, { 5308, 0x88a0 }, { 5312, 0x0000 },
02691 { 5312, 0x0000 }, { 5312, 0x0000 }, { 5312, 0x8100 }, { 5314, 0x0002 },
02692 { 5315, 0x0090 }, { 5317, 0x9800 }, { 5320, 0xe006 }, { 5325, 0x7010 },
02693 /* 0x9900 */
02694 { 5329, 0x1608 }, { 5333, 0x4109 }, { 5337, 0x0101 }, { 5339, 0x0000 },
02695 { 5339, 0x3a20 }, { 5344, 0x0096 }, { 5348, 0x0000 }, { 5348, 0x0000 },
02696 { 5348, 0x0000 }, { 5348, 0x2240 }, { 5351, 0x7120 }, { 5356, 0x021a },
02697 { 5360, 0x0002 }, { 5361, 0xa227 }, { 5368, 0x2000 }, { 5369, 0x8002 },
02698 /* 0x9a00 */
02699 { 5371, 0xc102 }, { 5375, 0x0200 }, { 5376, 0x0800 }, { 5377, 0x00c1 },
02700 { 5380, 0x2029 }, { 5384, 0x8ca0 }, { 5389, 0x0624 }, { 5393, 0x0000 },
02701 { 5393, 0x0000 }, { 5393, 0x0000 }, { 5393, 0x0100 }, { 5394, 0x0100 },
02702 { 5395, 0x0000 }, { 5395, 0x0118 }, { 5398, 0x4020 }, { 5400, 0x0000 },
02703 /* 0x9b00 */
02704 { 5400, 0x0000 }, { 5400, 0x0400 }, { 5401, 0x0480 }, { 5403, 0x1002 },
02705 { 5405, 0x803e }, { 5411, 0x0410 }, { 5413, 0x8000 }, { 5414, 0x0000 },
02706 { 5414, 0x4000 }, { 5415, 0x8002 }, { 5417, 0x4800 }, { 5419, 0x0000 },
02707 { 5419, 0x0200 }, { 5420, 0x0040 }, { 5421, 0x0110 }, { 5423, 0x0000 },
02708 /* 0x9c00 */
02709 { 5423, 0x2000 }, { 5424, 0x0025 }, { 5427, 0x0020 }, { 5428, 0x0804 },
02710 { 5430, 0x0280 }, { 5432, 0x0080 }, { 5433, 0x0000 }, { 5433, 0x0000 },
02711 { 5433, 0x0000 }, { 5433, 0x0000 }, { 5433, 0x0000 }, { 5433, 0x0000 },
02712 { 5433, 0x0000 }, { 5433, 0x0000 }, { 5433, 0x02a0 }, { 5436, 0x0058 },
02713 /* 0x9d00 */
02714 { 5439, 0x0200 }, { 5440, 0x0800 }, { 5441, 0x0140 }, { 5443, 0x0800 },
02715 { 5444, 0x0000 }, { 5444, 0x2002 }, { 5446, 0x1003 }, { 5449, 0x0004 },
02716 { 5450, 0x0000 }, { 5450, 0x0000 }, { 5450, 0x8200 }, { 5452, 0x0010 },
02717 { 5453, 0x0010 }, { 5454, 0x0080 }, { 5455, 0x0000 }, { 5455, 0x0704 },
02718 /* 0x9e00 */
02719 { 5459, 0x0000 }, { 5459, 0x4400 }, { 5461, 0x0000 }, { 5461, 0x0000 },
02720 { 5461, 0x0000 }, { 5461, 0x0000 }, { 5461, 0x0000 }, { 5461, 0xa220 },
02721 { 5465, 0x0000 }, { 5465, 0xa08c }, { 5470, 0x0020 }, { 5471, 0x4830 },
02722 { 5475, 0x6008 }, { 5478, 0x5912 }, { 5484, 0x0100 }, { 5485, 0x0010 },
02723 /* 0x9f00 */
02724 { 5486, 0x4180 }, { 5489, 0x0008 }, { 5490, 0x0001 }, { 5491, 0x0800 },
02725 { 5492, 0x4c00 }, { 5495, 0x8004 }, { 5497, 0x1482 }, { 5501, 0x0080 },
02726 { 5502, 0x2000 }, { 5503, 0x1021 },
02727 };
02728 static const Summary16 ksc5601_uni2indx_pageac[698] = {
```

```
02729  /* 0xac00 */
02730 { 5506, 0x0793 }, { 5513, 0x3eff }, { 5526, 0xb011 }, { 5531, 0x1303 },
02731 { 5536, 0x2801 }, { 5539, 0x1110 }, { 5542, 0x0000 }, { 5542, 0x0593 },
02732 { 5548, 0x1e7b }, { 5558, 0xb011 }, { 5563, 0x9703 }, { 5570, 0x3b01 },
02733 { 5576, 0x1112 }, { 5580, 0x00a0 }, { 5582, 0x9593 }, { 5590, 0x306b },
02734  /* 0xad00 */
02735 { 5597, 0xb051 }, { 5603, 0x1102 }, { 5606, 0x3201 }, { 5610, 0x1130 },
02736 { 5614, 0x02b0 }, { 5618, 0x0111 }, { 5621, 0x300a }, { 5625, 0xb879 },
02737 { 5634, 0x1306 }, { 5639, 0x3001 }, { 5642, 0x0010 }, { 5643, 0x0080 },
02738 { 5644, 0x0113 }, { 5648, 0x100b }, { 5652, 0x0011 }, { 5654, 0x9300 },
02739  /* 0xae00 */
02740 { 5658, 0x2b03 }, { 5664, 0x0010 }, { 5665, 0x0000 }, { 5665, 0x0593 },
02741 { 5671, 0x746b }, { 5680, 0xb051 }, { 5686, 0x1323 }, { 5692, 0x3b01 },
02742 { 5698, 0x1030 }, { 5701, 0x0000 }, { 5701, 0x0000 }, { 5701, 0x7000 },
02743 { 5704, 0xb011 }, { 5709, 0x1303 }, { 5714, 0x2900 }, { 5717, 0x1110 },
02744  /* 0xaf00 */
02745 { 5720, 0x2180 }, { 5723, 0x0001 }, { 5724, 0x3000 }, { 5726, 0xb015 },
02746 { 5732, 0x030e }, { 5737, 0x3001 }, { 5740, 0x0030 }, { 5742, 0x0200 },
02747 { 5743, 0x0111 }, { 5746, 0x1023 }, { 5750, 0x0000 }, { 5750, 0x1300 },
02748 { 5753, 0x6b81 }, { 5760, 0x1010 }, { 5762, 0x0300 }, { 5764, 0x0113 },
02749  /* 0xb000 */
02750 { 5768, 0x1013 }, { 5772, 0x3011 }, { 5776, 0x0100 }, { 5777, 0x0000 },
02751 { 5777, 0x5530 }, { 5783, 0x22b8 }, { 5789, 0x0000 }, { 5789, 0x3000 },
02752 { 5791, 0xb011 }, { 5796, 0x9702 }, { 5802, 0xfb07 }, { 5812, 0x113a },
02753 { 5818, 0x03b0 }, { 5823, 0x0113 }, { 5827, 0x0021 }, { 5829, 0x0000 },
02754  /* 0xb100 */
02755 { 5829, 0x1b00 }, { 5833, 0x3b0d }, { 5841, 0x1138 }, { 5846, 0x03b0 },
02756 { 5851, 0x0113 }, { 5855, 0x1133 }, { 5861, 0x0001 }, { 5862, 0x1300 },
02757 { 5865, 0x2b05 }, { 5871, 0x111c }, { 5876, 0x0100 }, { 5877, 0x0000 },
02758 { 5877, 0x1000 }, { 5878, 0xb011 }, { 5883, 0x1300 }, { 5886, 0x2a01 },
02759  /* 0xb200 */
02760 { 5890, 0x1930 }, { 5895, 0x02b0 }, { 5899, 0x0001 }, { 5900, 0x1010 },
02761 { 5902, 0x0000 }, { 5902, 0x1100 }, { 5904, 0x0301 }, { 5907, 0x1030 },
02762 { 5910, 0x0230 }, { 5913, 0x0713 }, { 5919, 0x146b }, { 5926, 0x0011 },
02763 { 5928, 0x1300 }, { 5931, 0x2b05 }, { 5937, 0xf974 }, { 5947, 0x8fb8 },
02764  /* 0xb300 */
02765 { 5956, 0x0113 }, { 5960, 0x103b }, { 5966, 0x0000 }, { 5966, 0x0000 },
02766 { 5966, 0x0000 }, { 5966, 0xd970 }, { 5974, 0x4ab0 }, { 5980, 0x0113 },
02767 { 5984, 0x103b }, { 5990, 0x0011 }, { 5992, 0x1103 }, { 5996, 0x0000 },
02768 { 5996, 0x5930 }, { 6002, 0x2ab1 }, { 6009, 0x0111 }, { 6012, 0x1000 },
02769  /* 0xb400 */
02770 { 6013, 0x0000 }, { 6013, 0x1101 }, { 6016, 0x0b01 }, { 6020, 0x0010 },
02771 { 6021, 0x0000 }, { 6021, 0x0113 }, { 6025, 0x102b }, { 6030, 0x0000 },
02772 { 6030, 0x0101 }, { 6032, 0x2000 }, { 6033, 0x1110 }, { 6036, 0x02a0 },
02773 { 6039, 0x0111 }, { 6042, 0x3021 }, { 6046, 0xb059 }, { 6053, 0x0102 },
02774  /* 0xb500 */
02775 { 6055, 0x0000 }, { 6055, 0x1930 }, { 6060, 0x07b0 }, { 6066, 0x0113 },
02776 { 6070, 0x383b }, { 6078, 0xb011 }, { 6083, 0x0003 }, { 6085, 0x0000 },
02777 { 6085, 0x0000 }, { 6085, 0x0000 }, { 6085, 0xd13 }, { 6091, 0x383b },
02778 { 6099, 0xb011 }, { 6104, 0x0103 }, { 6107, 0x1000 }, { 6108, 0x0000 },
02779  /* 0xb600 */
02780 { 6108, 0x0000 }, { 6108, 0x0113 }, { 6112, 0x1020 }, { 6114, 0x0010 },
02781 { 6115, 0x0100 }, { 6116, 0x0000 }, { 6116, 0x0110 }, { 6118, 0x0000 },
02782 { 6118, 0x0000 }, { 6118, 0x3000 }, { 6120, 0x1811 }, { 6124, 0x0002 },
02783 { 6125, 0x0000 }, { 6125, 0x0010 }, { 6126, 0x0000 }, { 6126, 0x0111 },
02784  /* 0xb700 */
02785 { 6129, 0x0023 }, { 6132, 0x0000 }, { 6132, 0x9300 }, { 6136, 0x0b01 },
02786 { 6140, 0x1110 }, { 6143, 0x0030 }, { 6145, 0x0111 }, { 6148, 0x302b },
02787 { 6154, 0xb011 }, { 6159, 0x13c7 }, { 6167, 0x3b01 }, { 6173, 0x0130 },
02788 { 6176, 0x0280 }, { 6178, 0x0000 }, { 6178, 0x3000 }, { 6180, 0xb011 },
02789  /* 0xb800 */
02790 { 6185, 0x1383 }, { 6191, 0x2b01 }, { 6196, 0x1130 }, { 6200, 0x03b0 },
02791 { 6205, 0x0011 }, { 6207, 0x300a }, { 6211, 0xb011 }, { 6216, 0x1102 },
02792 { 6219, 0x2000 }, { 6220, 0x0000 }, { 6220, 0x0100 }, { 6221, 0x0111 },
02793 { 6224, 0x102b }, { 6229, 0xa011 }, { 6233, 0x1302 }, { 6237, 0x2b01 },
02794  /* 0xb900 */
02795 { 6242, 0x0010 }, { 6243, 0x0100 }, { 6244, 0x0001 }, { 6245, 0x3000 },
02796 { 6247, 0x9011 }, { 6251, 0x1302 }, { 6255, 0x2b01 }, { 6260, 0x1130 },
02797 { 6264, 0x66b0 }, { 6271, 0x0000 }, { 6271, 0x3000 }, { 6273, 0xb011 },
02798 { 6278, 0xd302 }, { 6284, 0x6b07 }, { 6292, 0x113a }, { 6298, 0x07b0 },
02799  /* 0xba00 */
02800 { 6304, 0x0103 }, { 6307, 0x0020 }, { 6308, 0x0000 }, { 6308, 0x1300 },
02801 { 6311, 0x6b05 }, { 6318, 0x1138 }, { 6323, 0x03b0 }, { 6328, 0x0113 },
02802 { 6332, 0x10b8 }, { 6337, 0x0000 }, { 6337, 0x1b00 }, { 6341, 0x2b05 },
02803 { 6347, 0x0110 }, { 6349, 0x0300 }, { 6351, 0x0000 }, { 6351, 0x1000 },
02804  /* 0xbb00 */
02805 { 6352, 0xa011 }, { 6356, 0x1102 }, { 6359, 0xa0a1 }, { 6362, 0x7970 },
02806 { 6370, 0xa2b0 }, { 6376, 0x0111 }, { 6379, 0x100a }, { 6382, 0x0000 },
02807 { 6382, 0x1100 }, { 6384, 0x0001 }, { 6385, 0x1110 }, { 6388, 0x0090 },
02808 { 6390, 0x0111 }, { 6393, 0x0009 }, { 6395, 0x0000 }, { 6395, 0x9300 },
02809  /* 0xbc00 */
02810 { 6399, 0xbb05 }, { 6407, 0xf9f2 }, { 6418, 0x22b0 }, { 6423, 0x0113 },
02811 { 6427, 0x323b }, { 6435, 0x2001 }, { 6437, 0x0000 }, { 6437, 0x0000 },
02812 { 6437, 0x5930 }, { 6443, 0x06b0 }, { 6448, 0x0193 }, { 6453, 0x303b },
02813 { 6460, 0xa011 }, { 6464, 0x1123 }, { 6469, 0x0000 }, { 6469, 0x1170 },
02814  /* 0xbd00 */
02815 { 6474, 0x02b0 }, { 6478, 0x0011 }, { 6480, 0x1010 }, { 6482, 0x0000 },
```

```
02816 { 6482, 0x1301 }, { 6486, 0x0301 }, { 6489, 0x0110 }, { 6491, 0x0000 },
02817 { 6491, 0x0793 }, { 6498, 0x162b }, { 6505, 0x0010 }, { 6506, 0x0101 },
02818 { 6508, 0x0000 }, { 6508, 0x1130 }, { 6512, 0x0200 }, { 6513, 0x0111 },
02819 /* 0xbe00 */
02820 { 6516, 0x3029 }, { 6521, 0xb011 }, { 6526, 0x0000 }, { 6526, 0x0000 },
02821 { 6526, 0x5130 }, { 6531, 0x0eb0 }, { 6537, 0x0513 }, { 6542, 0x383b },
02822 { 6550, 0xb011 }, { 6555, 0x0303 }, { 6559, 0x0100 }, { 6560, 0x0000 },
02823 { 6560, 0x0000 }, { 6560, 0x0193 }, { 6565, 0x1039 }, { 6570, 0x0000 },
02824 /* 0xbf00 */
02825 { 6570, 0x0302 }, { 6573, 0x3b00 }, { 6578, 0x0000 }, { 6578, 0x0000 },
02826 { 6578, 0x0113 }, { 6582, 0x0023 }, { 6585, 0x0000 }, { 6585, 0x0000 },
02827 { 6585, 0x0000 }, { 6585, 0x0010 }, { 6586, 0x0000 }, { 6586, 0x0001 },
02828 { 6587, 0x3020 }, { 6590, 0x9011 }, { 6594, 0x0002 }, { 6595, 0x0000 },
02829 /* 0xc000 */
02830 { 6595, 0x0000 }, { 6595, 0x0000 }, { 6595, 0x0000 }, { 6595, 0x1000 },
02831 { 6596, 0x0000 }, { 6596, 0x1102 }, { 6599, 0x0301 }, { 6602, 0x0000 },
02832 { 6602, 0x0000 }, { 6602, 0x0113 }, { 6606, 0xb02b }, { 6613, 0xb079 },
02833 { 6621, 0x1323 }, { 6627, 0x3b01 }, { 6633, 0x1130 }, { 6637, 0x02b0 },
02834 /* 0xc100 */
02835 { 6641, 0x0111 }, { 6644, 0xf021 }, { 6650, 0xb0d9 }, { 6658, 0x1343 },
02836 { 6664, 0x3b01 }, { 6670, 0x1130 }, { 6674, 0x03b0 }, { 6679, 0x0111 },
02837 { 6682, 0x7020 }, { 6686, 0xb051 }, { 6692, 0x1322 }, { 6697, 0x2001 },
02838 { 6699, 0x1110 }, { 6702, 0x0190 }, { 6705, 0x0111 }, { 6708, 0x300b },
02839 /* 0xc200 */
02840 { 6713, 0xb011 }, { 6718, 0x9302 }, { 6723, 0xab01 }, { 6729, 0x0016 },
02841 { 6732, 0x0100 }, { 6733, 0x0113 }, { 6737, 0x3021 }, { 6741, 0xb011 },
02842 { 6746, 0x0302 }, { 6749, 0x2901 }, { 6753, 0x3130 }, { 6758, 0x02b0 },
02843 { 6762, 0x0000 }, { 6762, 0x3000 }, { 6764, 0xb819 }, { 6771, 0x1b42 },
02844 /* 0xc300 */
02845 { 6777, 0x3301 }, { 6782, 0x1138 }, { 6787, 0x0330 }, { 6791, 0x0000 },
02846 { 6791, 0x0020 }, { 6792, 0x0000 }, { 6792, 0x1300 }, { 6795, 0x3305 },
02847 { 6801, 0x1110 }, { 6804, 0x0000 }, { 6804, 0x0000 }, { 6804, 0x0000 },
02848 { 6804, 0x0001 }, { 6805, 0x9300 }, { 6809, 0x2305 }, { 6814, 0x0130 },
02849 /* 0xc400 */
02850 { 6817, 0x0100 }, { 6818, 0x0001 }, { 6819, 0x1010 }, { 6821, 0x3011 },
02851 { 6825, 0x0100 }, { 6826, 0x0000 }, { 6826, 0x1130 }, { 6830, 0x0230 },
02852 { 6833, 0x0001 }, { 6834, 0x1010 }, { 6836, 0x0000 }, { 6836, 0x1100 },
02853 { 6838, 0x0000 }, { 6838, 0x0000 }, { 6838, 0x0200 }, { 6839, 0x8513 },
02854 /* 0xc500 */
02855 { 6845, 0x1003 }, { 6848, 0x1011 }, { 6851, 0x1300 }, { 6854, 0x2b01 },
02856 { 6859, 0x7730 }, { 6867, 0x63b8 }, { 6875, 0x0113 }, { 6879, 0x303b },
02857 { 6886, 0xb091 }, { 6892, 0x11a2 }, { 6897, 0x0201 }, { 6899, 0x7b30 },
02858 { 6907, 0x57f0 }, { 6916, 0x0113 }, { 6920, 0x702b }, { 6927, 0xf0d1 },
02859 /* 0xc600 */
02860 { 6935, 0x11e3 }, { 6942, 0x1b01 }, { 6947, 0x7130 }, { 6953, 0x0ab9 },
02861 { 6960, 0x0113 }, { 6964, 0x303b }, { 6971, 0x9001 }, { 6974, 0x1302 },
02862 { 6978, 0x2b01 }, { 6983, 0x1130 }, { 6987, 0x02b0 }, { 6991, 0x0713 },
02863 { 6997, 0x302b }, { 7003, 0x3011 }, { 7007, 0x1303 }, { 7012, 0x2301 },
02864 /* 0xc700 */
02865 { 7016, 0x1130 }, { 7020, 0x02b0 }, { 7024, 0x0113 }, { 7028, 0x30ab },
02866 { 7035, 0xb411 }, { 7041, 0x11fe }, { 7050, 0x0901 }, { 7053, 0x7130 },
02867 { 7059, 0x47b8 }, { 7067, 0x05d3 }, { 7074, 0x307b }, { 7082, 0xb011 },
02868 { 7087, 0x5303 }, { 7093, 0x2101 }, { 7096, 0x1110 }, { 7099, 0x0000 },
02869 /* 0xc800 */
02870 { 7099, 0x0513 }, { 7104, 0x306b }, { 7111, 0xb011 }, { 7116, 0x1102 },
02871 { 7119, 0x3301 }, { 7124, 0x0010 }, { 7125, 0x0000 }, { 7125, 0x0513 },
02872 { 7130, 0x38eb }, { 7139, 0xa010 }, { 7142, 0x0102 }, { 7144, 0x3000 },
02873 { 7146, 0x1110 }, { 7149, 0x02b0 }, { 7153, 0x0013 }, { 7156, 0x3020 },
02874 /* 0xc900 */
02875 { 7159, 0xb071 }, { 7166, 0x0102 }, { 7168, 0x1000 }, { 7169, 0x0010 },
02876 { 7170, 0x0000 }, { 7170, 0x0113 }, { 7174, 0x100b }, { 7178, 0x1011 },
02877 { 7181, 0x1300 }, { 7184, 0x2b01 }, { 7189, 0x0000 }, { 7189, 0x0000 },
02878 { 7189, 0x0593 }, { 7195, 0x366b }, { 7204, 0xb095 }, { 7211, 0x1303 },
02879 /* 0xca00 */
02880 { 7216, 0x3b01 }, { 7222, 0x0110 }, { 7224, 0x0200 }, { 7225, 0x0000 },
02881 { 7225, 0x3000 }, { 7227, 0xb011 }, { 7232, 0x0103 }, { 7235, 0x2000 },
02882 { 7236, 0x0010 }, { 7237, 0x0100 }, { 7238, 0x0000 }, { 7238, 0x3000 },
02883 { 7240, 0xb011 }, { 7245, 0x030a }, { 7249, 0x1001 }, { 7251, 0x0010 },
02884 /* 0xcb00 */
02885 { 7252, 0x0100 }, { 7253, 0x0111 }, { 7256, 0x0003 }, { 7258, 0x0000 },
02886 { 7258, 0x1302 }, { 7262, 0x2301 }, { 7266, 0x0010 }, { 7267, 0x0300 },
02887 { 7269, 0x0000 }, { 7269, 0x1000 }, { 7270, 0x0000 }, { 7270, 0x0100 },
02888 { 7271, 0x0000 }, { 7271, 0x0010 }, { 7272, 0x0290 }, { 7275, 0x0000 },
02889 /* 0xcc00 */
02890 { 7275, 0x3000 }, { 7277, 0x3011 }, { 7281, 0x5386 }, { 7288, 0x7b01 },
02891 { 7295, 0x1130 }, { 7299, 0x03b0 }, { 7304, 0x0151 }, { 7308, 0x0021 },
02892 { 7310, 0x0000 }, { 7310, 0x1300 }, { 7313, 0x3b01 }, { 7319, 0x1130 },
02893 { 7323, 0x02b0 }, { 7327, 0x0011 }, { 7329, 0x1010 }, { 7331, 0x0001 },
02894 /* 0xcd00 */
02895 { 7332, 0x1302 }, { 7336, 0x2b01 }, { 7341, 0x1110 }, { 7344, 0x0200 },
02896 { 7345, 0x0000 }, { 7345, 0x1000 }, { 7346, 0xb011 }, { 7351, 0x0102 },
02897 { 7353, 0x0100 }, { 7354, 0x1130 }, { 7358, 0x02b0 }, { 7362, 0x0001 },
02898 { 7363, 0x1010 }, { 7365, 0x0001 }, { 7366, 0x1100 }, { 7368, 0x2b01 },
02899 /* 0xce00 */
02900 { 7373, 0x1110 }, { 7376, 0x0210 }, { 7378, 0x0113 }, { 7382, 0x002b },
02901 { 7386, 0x0000 }, { 7386, 0x9300 }, { 7390, 0x2b03 }, { 7396, 0x1130 },
02902 { 7400, 0x02b0 }, { 7404, 0x0113 }, { 7408, 0x303b }, { 7415, 0x0000 },
```

```

02903 { 7415, 0x0002 }, { 7416, 0x0000 }, { 7416, 0x1930 }, { 7421, 0x03b0 },
02904 /* 0xc0f00 */
02905 { 7426, 0x0113 }, { 7430, 0x102b }, { 7435, 0xb011 }, { 7440, 0x0103 },
02906 { 7443, 0x0000 }, { 7443, 0x1130 }, { 7447, 0x02b0 }, { 7451, 0x0113 },
02907 { 7455, 0x1021 }, { 7458, 0x0000 }, { 7458, 0x0102 }, { 7460, 0x0001 },
02908 { 7461, 0x0010 }, { 7462, 0x0000 }, { 7462, 0x0113 }, { 7466, 0x102b },
02909 /* 0xd000 */
02910 { 7471, 0x0011 }, { 7473, 0x0102 }, { 7475, 0x2000 }, { 7476, 0x1130 },
02911 { 7480, 0x02b0 }, { 7484, 0x0111 }, { 7487, 0x3001 }, { 7490, 0x3011 },
02912 { 7494, 0x0002 }, { 7495, 0x0000 }, { 7495, 0x1130 }, { 7499, 0x02b0 },
02913 { 7503, 0x0313 }, { 7508, 0x303b }, { 7515, 0xb011 }, { 7520, 0x0103 },
02914 /* 0xd100 */
02915 { 7523, 0x2000 }, { 7524, 0x0000 }, { 7524, 0x0000 }, { 7524, 0x0513 },
02916 { 7529, 0x303b }, { 7536, 0xb011 }, { 7541, 0x1102 }, { 7544, 0x1000 },
02917 { 7545, 0x0110 }, { 7547, 0x0000 }, { 7547, 0x0113 }, { 7551, 0x142b },
02918 { 7557, 0x0001 }, { 7558, 0x0100 }, { 7559, 0x0000 }, { 7559, 0x0110 },
02919 /* 0xd200 */
02920 { 7561, 0x0280 }, { 7563, 0x0001 }, { 7564, 0x3000 }, { 7566, 0xb011 },
02921 { 7571, 0x0102 }, { 7573, 0x1000 }, { 7574, 0x0010 }, { 7575, 0x0000 },
02922 { 7575, 0x0113 }, { 7579, 0x1023 }, { 7583, 0x1011 }, { 7586, 0x9302 },
02923 { 7591, 0x0b05 }, { 7596, 0x1110 }, { 7599, 0x0030 }, { 7601, 0x0113 },
02924 /* 0xd300 */
02925 { 7605, 0x702b }, { 7612, 0xb051 }, { 7618, 0x1323 }, { 7624, 0x3b01 },
02926 { 7630, 0x0030 }, { 7632, 0x0000 }, { 7632, 0x0000 }, { 7632, 0x3000 },
02927 { 7634, 0xb011 }, { 7639, 0x1303 }, { 7644, 0x2b01 }, { 7649, 0x1110 },
02928 { 7652, 0x0330 }, { 7656, 0x0101 }, { 7658, 0x300a }, { 7662, 0xb011 },
02929 /* 0xd400 */
02930 { 7667, 0x0102 }, { 7669, 0x2000 }, { 7670, 0x0000 }, { 7670, 0x0000 },
02931 { 7670, 0x0011 }, { 7672, 0x1000 }, { 7673, 0xa011 }, { 7677, 0x9300 },
02932 { 7681, 0x2b05 }, { 7687, 0x0010 }, { 7688, 0x0200 }, { 7689, 0x0000 },
02933 { 7689, 0x1000 }, { 7690, 0x9011 }, { 7694, 0x1100 }, { 7696, 0x2901 },
02934 /* 0xd500 */
02935 { 7700, 0x1110 }, { 7703, 0x00b0 }, { 7706, 0x0000 }, { 7706, 0x3000 },
02936 { 7708, 0xb011 }, { 7713, 0x1302 }, { 7717, 0x2b21 }, { 7723, 0x1130 },
02937 { 7727, 0x03b0 }, { 7732, 0x0001 }, { 7733, 0x0020 }, { 7734, 0x0000 },
02938 { 7734, 0x1300 }, { 7737, 0x2b05 }, { 7743, 0x1130 }, { 7747, 0x02b0 },
02939 /* 0xd600 */
02940 { 7751, 0x0113 }, { 7755, 0x103b }, { 7761, 0x2011 }, { 7764, 0x1300 },
02941 { 7767, 0x2b21 }, { 7773, 0x1132 }, { 7778, 0x0280 }, { 7780, 0x0013 },
02942 { 7783, 0x3028 }, { 7787, 0xa011 }, { 7791, 0x1102 }, { 7794, 0xa0a1 },
02943 { 7797, 0x1130 }, { 7801, 0x0292 }, { 7805, 0x0111 }, { 7808, 0x3021 },
02944 /* 0xd700 */
02945 { 7812, 0x0011 }, { 7814, 0x1302 }, { 7818, 0x2b01 }, { 7823, 0x1130 },
02946 { 7827, 0x0290 }, { 7830, 0x03d3 }, { 7837, 0x122b }, { 7843, 0x3011 },
02947 { 7847, 0x1302 }, { 7851, 0x2b01 },
02948 };
02949 static const Summary16 ksc5601_uni2indx_pagef9[17] = {
02950 /* 0xf900 */
02951 { 7856, 0xffff }, { 7872, 0xffff }, { 7888, 0xffff }, { 7904, 0xffff },
02952 { 7920, 0xffff }, { 7936, 0xffff }, { 7952, 0xffff }, { 7968, 0xffff },
02953 { 7984, 0xffff }, { 8000, 0xffff }, { 8016, 0xffff }, { 8032, 0xffff },
02954 { 8048, 0xffff }, { 8064, 0xffff }, { 8080, 0xffff }, { 8096, 0xffff },
02955 /* 0xfa00 */
02956 { 8112, 0x0fff },
02957 };
02958 static const Summary16 ksc5601_uni2indx_pageff[15] = {
02959 /* 0xff00 */
02960 { 8124, 0xfffe }, { 8139, 0xffff }, { 8155, 0xffff }, { 8171, 0xffff },
02961 { 8187, 0xffff }, { 8203, 0x7fff }, { 8218, 0x0000 }, { 8218, 0x0000 },
02962 { 8218, 0x0000 }, { 8218, 0x0000 }, { 8218, 0x0000 }, { 8218, 0x0000 },
02963 { 8218, 0x0000 }, { 8218, 0x0000 }, { 8218, 0x006f },
02964 };
02965
02966 static int
02967 ksc5601_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
02968 {
02969     (void)conv;
02970     if (n >= 2) {
02971         const Summary16 *summary = NULL;
02972         if (wc < 0x0460)
02973             summary = &ksc5601_uni2indx_page00[(wc>>4)];
02974         else if (wc >= 0x2000 && wc < 0x2670)
02975             summary = &ksc5601_uni2indx_page20[(wc>>4)-0x200];
02976         else if (wc >= 0x3000 && wc < 0x33e0)
02977             summary = &ksc5601_uni2indx_page30[(wc>>4)-0x300];
02978         else if (wc >= 0x4e00 && wc < 0x9fa0)
02979             summary = &ksc5601_uni2indx_page4e[(wc>>4)-0x4e0];
02980         else if (wc >= 0xac00 && wc < 0xd7a0)
02981             summary = &ksc5601_uni2indx_pageac[(wc>>4)-0xac0];
02982         else if (wc >= 0xf900 && wc < 0xfaf0)
02983             summary = &ksc5601_uni2indx_pagef9[(wc>>4)-0xf90];
02984         else if (wc >= 0xff00 && wc < 0xffff)
02985             summary = &ksc5601_uni2indx_pageff[(wc>>4)-0xff0];
02986         if (summary) {
02987             unsigned short used = summary->used;
02988             unsigned int i = wc & 0x0f;
02989             if (used & ((unsigned short) 1 << i)) {

```

```

02990         unsigned short c;
02991         /* Keep in `used' only the bits 0..i-1. */
02992         used &= ((unsigned short) 1 << i) - 1;
02993         /* Add `summary->indx' and the number of bits set in `used'. */
02994         used = (used & 0x5555) + ((used & 0xaaaa) >> 1);
02995         used = (used & 0x3333) + ((used & 0xcccc) >> 2);
02996         used = (used & 0x0f0f) + ((used & 0xf0f0) >> 4);
02997         used = (used & 0x00ff) + (used >> 8);
02998         c = ksc5601_2charset[summary->indx + used];
02999         r[0] = (c >> 8); r[1] = (c & 0xff);
03000         return 2;
03001     }
03002 }
03003 return RET_ILSEQ;
03004 }
03005 return RET_TOOSMALL;
03006 }
03007 #endif /* NEED_TOMB */

```

12.292 mulelao.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/mulelao.h,v 1.3 2000/11/29 17:40:35 dawes Exp $ */
00002
00003 /*
00004  * MULELao-1
00005  */
00006
00007 static const unsigned short mulelao_2uni[96] = {
00008     /* 0xa0 */
00009     0x00a0, 0x0e81, 0x0e82, 0xffff, 0x0e84, 0xffff, 0xffff, 0x0e87,
00010     0x0e88, 0xffff, 0x0e8a, 0xffff, 0xffff, 0x0e8d, 0xffff, 0xffff,
00011     /* 0xb0 */
00012     0xffff, 0xffff, 0xffff, 0xffff, 0x0e94, 0x0e95, 0x0e96, 0x0e97,
00013     0xffff, 0x0e99, 0x0e9a, 0x0e9b, 0x0e9c, 0x0e9d, 0x0e9e, 0x0e9f,
00014     /* 0xc0 */
00015     0xffff, 0x0ea1, 0x0ea2, 0x0ea3, 0xffff, 0x0ea5, 0xffff, 0x0ea7,
00016     0xffff, 0xffff, 0x0eaa, 0x0eab, 0xffff, 0x0ead, 0x0eae, 0x0eaf,
00017     /* 0xd0 */
00018     0x0eb0, 0x0eb1, 0x0eb2, 0x0eb3, 0x0eb4, 0x0eb5, 0x0eb6, 0x0eb7,
00019     0x0eb8, 0x0eb9, 0xffff, 0x0ebb, 0x0ebc, 0x0ebd, 0xffff, 0xffff,
00020     /* 0xe0 */
00021     0x0ec0, 0x0ec1, 0x0ec2, 0x0ec3, 0x0ec4, 0xffff, 0x0ec6, 0xffff,
00022     0x0ec8, 0x0ec9, 0x0eca, 0x0ecb, 0x0ecc, 0x0ecd, 0xffff, 0xffff,
00023     /* 0xf0 */
00024     0x0ed0, 0x0ed1, 0x0ed2, 0x0ed3, 0x0ed4, 0x0ed5, 0x0ed6, 0x0ed7,
00025     0x0ed8, 0x0ed9, 0xffff, 0xffff, 0x0edc, 0x0edd, 0xffff, 0xffff,
00026 };
00027
00028 static int
00029 mulelao_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00030 {
00031     unsigned char c = *s;
00032     if (c < 0xa0) {
00033         *pwc = (ucs4_t) c;
00034         return 1;
00035     }
00036     else {
00037         unsigned short wc = mulelao_2uni[c-0xa0];
00038         if (wc != 0xffff) {
00039             *pwc = (ucs4_t) wc;
00040             return 1;
00041         }
00042     }
00043     return RET_ILSEQ;
00044 }
00045
00046 static const unsigned char mulelao_page0e[96] = {
00047     0x00, 0xa1, 0xa2, 0x00, 0xa4, 0x00, 0x00, 0xa7, /* 0x80-0x87 */
00048     0xa8, 0x00, 0xaa, 0x00, 0x00, 0xad, 0x00, 0x00, /* 0x88-0x8f */
00049     0x00, 0x00, 0x00, 0x00, 0xb4, 0xb5, 0xb6, 0xb7, /* 0x90-0x97 */
00050     0x00, 0xb9, 0xba, 0xbb, 0xbc, 0xbd, 0xbe, 0xbf, /* 0x98-0x9f */
00051     0x00, 0xc1, 0xc2, 0xc3, 0x00, 0xc5, 0x00, 0xc7, /* 0xa0-0xa7 */
00052     0x00, 0x00, 0xca, 0xcb, 0x00, 0xcd, 0xce, 0xcf, /* 0xa8-0xaf */
00053     0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, /* 0xb0-0xb7 */
00054     0xd8, 0xd9, 0x00, 0xdb, 0xdc, 0xdd, 0x00, 0x00, /* 0xb8-0xbf */
00055     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0x00, 0xe6, 0x00, /* 0xc0-0xc7 */
00056     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0x00, 0x00, /* 0xc8-0xcf */
00057     0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0xd0-0xd7 */
00058     0xf8, 0xf9, 0x00, 0x00, 0xfc, 0xfd, 0x00, 0x00, /* 0xd8-0xdf */
00059 };
00060
00061 static int
00062 mulelao_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00063 {
00064     unsigned char c = 0;

```

```

00065     if (wc < 0x00a0) {
00066         *r = wc;
00067         return 1;
00068     }
00069     else if (wc == 0x00a0)
00070         c = 0xa0;
00071     else if (wc >= 0x0e80 && wc < 0x0ee0)
00072         c = mulelao_page0e[wc-0x0e80];
00073     if (c != 0) {
00074         *r = c;
00075         return 1;
00076     }
00077     return RET_ILSEQ;
00078 }

```

12.293 tatar_cyr.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/tatar_cyr.h,v 1.3 2000/12/04 18:49:42 dawes Exp $ */
00002
00003 /*
00004  * TATAR-CYR
00005  */
00006
00007 static const unsigned short tatar_cyr_2uni[128] = {
00008     /* 0x80 */
00009     0x04d8, 0x0403, 0x201a, 0x0453, 0x201e, 0x2026, 0x2020, 0x2021,
00010     0x20ac, 0x2030, 0x04e8, 0x2039, 0x04ae, 0x0496, 0x04a2, 0x04ba,
00011     /* 0x90 */
00012     0x04d9, 0x2018, 0x2019, 0x201c, 0x201d, 0x2022, 0x2013, 0x2014,
00013     0x98, 0x2122, 0x04e9, 0x203a, 0x04af, 0x0497, 0x04a3, 0x04bb,
00014     /* 0xa0 */
00015     0x00a0, 0x040e, 0x045e, 0x0408, 0x00a4, 0x0490, 0x00a6, 0x00a7,
00016     0x0401, 0x00a9, 0x0404, 0x00ab, 0x00ac, 0x00ad, 0x00ae, 0x0407,
00017     /* 0xb0 */
00018     0x00b0, 0x00b1, 0x0406, 0x0456, 0x0491, 0x00b5, 0x00b6, 0x00b7,
00019     0x0451, 0x2116, 0x0454, 0x00bb, 0x0458, 0x0405, 0x0455, 0x0457,
00020     /* 0xc0 */
00021     0x0410, 0x0411, 0x0412, 0x0413, 0x0414, 0x0415, 0x0416, 0x0417,
00022     0x0418, 0x0419, 0x041a, 0x041b, 0x041c, 0x041d, 0x041e, 0x041f,
00023     /* 0xd0 */
00024     0x0420, 0x0421, 0x0422, 0x0423, 0x0424, 0x0425, 0x0426, 0x0427,
00025     0x0428, 0x0429, 0x042a, 0x042b, 0x042c, 0x042d, 0x042e, 0x042f,
00026     /* 0xe0 */
00027     0x0430, 0x0431, 0x0432, 0x0433, 0x0434, 0x0435, 0x0436, 0x0437,
00028     0x0438, 0x0439, 0x043a, 0x043b, 0x043c, 0x043d, 0x043e, 0x043f,
00029     /* 0xf0 */
00030     0x0440, 0x0441, 0x0442, 0x0443, 0x0444, 0x0445, 0x0446, 0x0447,
00031     0x0448, 0x0449, 0x044a, 0x044b, 0x044c, 0x044d, 0x044e, 0x044f,
00032 };
00033
00034 static int
00035 tatar_cyr_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00036 {
00037     unsigned char c = *s;
00038     if (c < 0x80)
00039         *pwc = (ucs4_t) c;
00040     else
00041         *pwc = (ucs4_t) tatar_cyr_2uni[c-0x80];
00042     return 1;
00043 }
00044
00045 static const unsigned char tatar_cyr_page00[32] = {
00046     0xa0, 0x00, 0x00, 0x00, 0xa4, 0x00, 0xa6, 0xa7, /* 0xa0-0xa7 */
00047     0x00, 0xa9, 0x00, 0xab, 0xac, 0xad, 0xae, 0x00, /* 0xa8-0xaf */
00048     0xb0, 0xb1, 0x00, 0x00, 0x00, 0xb5, 0xb6, 0xb7, /* 0xb0-0xb7 */
00049     0x00, 0x00, 0x00, 0xbb, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00050 };
00051 static const unsigned char tatar_cyr_page04[240] = {
00052     0x00, 0xa8, 0x00, 0x81, 0xaa, 0xbd, 0xb2, 0xaf, /* 0x00-0x07 */
00053     0xa3, 0x00, 0x00, 0x00, 0x00, 0x00, 0xa1, 0x00, /* 0x08-0x0f */
00054     0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0x10-0x17 */
00055     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0x18-0x1f */
00056     0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, /* 0x20-0x27 */
00057     0xd8, 0xd9, 0xda, 0xdb, 0xdc, 0xdd, 0xde, 0xdf, /* 0x28-0x2f */
00058     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0x30-0x37 */
00059     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0x38-0x3f */
00060     0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0x40-0x47 */
00061     0xf8, 0xf9, 0xfa, 0xfb, 0xfc, 0xfd, 0xfe, 0xff, /* 0x48-0x4f */
00062     0x00, 0xb8, 0x00, 0x83, 0xba, 0xbe, 0xb3, 0xbf, /* 0x50-0x57 */
00063     0xbc, 0x00, 0x00, 0x00, 0x00, 0xa2, 0x00, /* 0x58-0x5f */
00064     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00065     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00066     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00067     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00068     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */

```

```

00069 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00070 0xa5, 0xb4, 0x00, 0x00, 0x00, 0x00, 0x8d, 0x9d, /* 0x90-0x97 */
00071 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x98-0x9f */
00072 0x00, 0x00, 0x8e, 0x9e, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00073 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x8c, 0x9c, /* 0xa8-0xaf */
00074 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb0-0xb7 */
00075 0x00, 0x00, 0x8f, 0x9f, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00076 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00077 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xc8-0xcf */
00078 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd0-0xd7 */
00079 0x80, 0x90, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xd8-0xdf */
00080 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe0-0xe7 */
00081 0x8a, 0x9a, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xe8-0xef */
00082 };
00083 static const unsigned char tatar_cyr_page20[48] = {
00084 0x00, 0x00, 0x00, 0x96, 0x97, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00085 0x91, 0x92, 0x82, 0x00, 0x93, 0x94, 0x84, 0x00, /* 0x18-0x1f */
00086 0x86, 0x87, 0x95, 0x00, 0x00, 0x00, 0x85, 0x00, /* 0x20-0x27 */
00087 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00088 0x89, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00089 0x00, 0x8b, 0x9b, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00090 };
00091 static const unsigned char tatar_cyr_page21[24] = {
00092 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xb9, 0x00, /* 0x10-0x17 */
00093 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00094 0x00, 0x00, 0x99, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00095 };
00096 static const unsigned char tatar_cyr_page22[1] = {
00097 0xb0, /* 0x16-0x16 */
00098 };
00099
00100 static int
00101 tatar_cyr_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00102 {
00103     unsigned char c = 0;
00104     if (wc < 0x0080) {
00105         *r = wc;
00106         return 1;
00107     }
00108     else if (wc >= 0x00a0 && wc < 0x00bc)
00109         c = tatar_cyr_page00[wc-0x00a0];
00110     else if (wc >= 0x0400 && wc < 0x04ef)
00111         c = tatar_cyr_page04[wc-0x0400];
00112     else if (wc >= 0x2010 && wc < 0x203b)
00113         c = tatar_cyr_page20[wc-0x2010];
00114     else if (wc == 0x20ac)
00115         c = 0x88;
00116     else if (wc >= 0x2110 && wc < 0x2123)
00117         c = tatar_cyr_page21[wc-0x2110];
00118     if (c != 0) {
00119         *r = c;
00120         return 1;
00121     }
00122     return RET_ILSEQ;
00123 }

```

12.294 tcvn.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/tcvn.h,v 1.3 2000/11/29 17:40:35 dawes Exp $ */
00002
00003 /*
00004  * TCVN-5712
00005  */
00006
00007 static const unsigned short tcvn_2uni_1[32] = {
00008     /* 0x00 */
00009     0x0000, 0x00da, 0x1ee4, 0x0003, 0x1eea, 0x1eec, 0x1eee, 0x0007,
00010     0x0008, 0x0009, 0x000a, 0x000b, 0x000c, 0x000d, 0x000e, 0x000f,
00011     /* 0x10 */
00012     0x0010, 0x1ee8, 0x1ef0, 0x1ef2, 0x1ef6, 0x1ef8, 0x00dd, 0x1ef4,
00013     0x0018, 0x0019, 0x001a, 0x001b, 0x001c, 0x001d, 0x001e, 0x001f,
00014 };
00015 static const unsigned short tcvn_2uni_2[128] = {
00016     /* 0x80 */
00017     0x00c0, 0x1ea2, 0x00c3, 0x00c1, 0x1ea0, 0x1eb6, 0x1eac, 0x00c8,
00018     0x1eba, 0x1ebc, 0x00c9, 0x1eb8, 0x1ec6, 0x00cc, 0x1ec8, 0x0128,
00019     /* 0x90 */
00020     0x00cd, 0x1eca, 0x00d2, 0x1ece, 0x00d5, 0x00d3, 0x1ecc, 0x1ed8,
00021     0x1edc, 0x1ede, 0x1ee0, 0x1eda, 0x1ee2, 0x00d9, 0x1ee6, 0x0168,
00022     /* 0xa0 */
00023     0x00a0, 0x0102, 0x00c2, 0x00ca, 0x00d4, 0x01a0, 0x01af, 0x0110,
00024     0x0103, 0x00e2, 0x00ea, 0x00f4, 0x01a1, 0x01b0, 0x0111, 0x1eb0,
00025     /* 0xb0 */
00026     0x0300, 0x0309, 0x0303, 0x0301, 0x0323, 0x00e0, 0x1ea3, 0x00e3,
00027     0x00e1, 0x1ea1, 0x1eb2, 0x1eb1, 0x1eb3, 0x1eb5, 0x1eaf, 0x1eb4,

```

```

00028  /* 0xc0 */
00029  0x1eae, 0x1ea6, 0x1ea8, 0x1eaa, 0x1ea4, 0x1ec0, 0x1eb7, 0x1ea7,
00030  0x1ea9, 0x1eab, 0x1ea5, 0x1ead, 0x00e8, 0x1ec2, 0x1ebb, 0x1ebd,
00031  /* 0xd0 */
00032  0x00e9, 0x1eb9, 0x1ec1, 0x1ec3, 0x1ec5, 0x1ebf, 0x1ec7, 0x00ec,
00033  0x1ec9, 0x1ec4, 0x1ebe, 0x1ed2, 0x0129, 0x00ed, 0x1ecb, 0x00f2,
00034  /* 0xe0 */
00035  0x1ed4, 0x1ecf, 0x00f5, 0x00f3, 0x1ecd, 0x1ed3, 0x1ed5, 0x1ed7,
00036  0x1ed1, 0x1ed9, 0x1edd, 0x1edf, 0x1ee1, 0x1edb, 0x1ee3, 0x00f9,
00037  /* 0xf0 */
00038  0x1ed6, 0x1ee7, 0x0169, 0x00fa, 0x1ee5, 0x1eeb, 0x1eed, 0x1eef,
00039  0x1ee9, 0x1ef1, 0x1ef3, 0x1ef7, 0x1ef9, 0x00fd, 0x1ef5, 0x1ed0,
00040  };
00041
00042  static int
00043  tcvn_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00044  {
00045      unsigned char c = *s;
00046      if (c < 0x20)
00047          *pwc = (ucs4_t) tcvn_2uni_1[c];
00048      else if (c < 0x80)
00049          *pwc = (ucs4_t) c;
00050      else
00051          *pwc = (ucs4_t) tcvn_2uni_2[c-0x80];
00052      return 1;
00053  }
00054
00055  static const unsigned char tcvn_page00[96+184] = {
00056      0xa0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00057      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa8-0xaf */
00058      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb0-0xb7 */
00059      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb8-0xbf */
00060      0x80, 0x83, 0xa2, 0x82, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00061      0x87, 0x8a, 0xa3, 0x00, 0x8d, 0x90, 0x00, 0x00, /* 0xc8-0xcf */
00062      0x00, 0x00, 0x92, 0x95, 0xa4, 0x94, 0x00, 0x00, /* 0xd0-0xd7 */
00063      0x00, 0x9d, 0x01, 0x00, 0x00, 0x16, 0x00, 0x00, /* 0xd8-0xdf */
00064      0xb5, 0xb8, 0xa9, 0xb7, 0x00, 0x00, 0x00, 0x00, /* 0xe0-0xe7 */
00065      0xcc, 0xd0, 0xaa, 0x00, 0xd7, 0xdd, 0x00, 0x00, /* 0xe8-0xef */
00066      0x00, 0x00, 0xdf, 0xe3, 0xab, 0xe2, 0x00, 0x00, /* 0xf0-0xf7 */
00067      0x00, 0xef, 0xf3, 0x00, 0x00, 0xfd, 0x00, 0x00, /* 0xf8-0xff */
00068      /* 0x0100 */
00069      0x00, 0x00, 0xa1, 0xa8, 0x00, 0x00, 0x00, 0x00, /* 0x00-0x07 */
00070      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00071      0xa7, 0xae, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00072      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00073      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00074      0x8f, 0xdc, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00075      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00076      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00077      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00078      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00079      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00080      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00081      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00082      0x9f, 0xf2, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00083      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00084      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00085      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00086      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00087      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */
00088      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x98-0x9f */
00089      0xa5, 0xac, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00090      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xa6, /* 0xa8-0xaf */
00091      0xad, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb0-0xb7 */
00092  };
00093  static const unsigned char tcvn_page03[40] = {
00094      0xb0, 0xb3, 0x00, 0xb2, 0x00, 0x00, 0x00, 0x00, /* 0x00-0x07 */
00095      0x00, 0xb1, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00096      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00097      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00098      0x00, 0x00, 0x00, 0xb4, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00099  };
00100  static const unsigned char tcvn_pagele[96] = {
00101      0x84, 0xb9, 0x81, 0xb6, 0xc4, 0xca, 0xc1, 0xc7, /* 0xa0-0xa7 */
00102      0xc2, 0xc8, 0xc3, 0xc9, 0x86, 0xcb, 0xc0, 0xbe, /* 0xa8-0xaf */
00103      0xaf, 0xbb, 0xba, 0xbc, 0xbf, 0xbd, 0x85, 0xc6, /* 0xb0-0xb7 */
00104      0x8b, 0xd1, 0x88, 0xce, 0x89, 0xcf, 0xda, 0xd5, /* 0xb8-0xbf */
00105      0xc5, 0xd2, 0xcd, 0xd3, 0xd9, 0xd4, 0x8c, 0xd6, /* 0xc0-0xc7 */
00106      0x8e, 0xd8, 0x91, 0xde, 0x96, 0xe4, 0x93, 0xe1, /* 0xc8-0xcf */
00107      0xff, 0xe8, 0xdb, 0xe5, 0xe0, 0xe6, 0xf0, 0xe7, /* 0xd0-0xd7 */
00108      0x97, 0xe9, 0x9b, 0xed, 0x98, 0xea, 0x99, 0xeb, /* 0xd8-0xdf */
00109      0x9a, 0xec, 0x9c, 0xee, 0x02, 0xf4, 0x9e, 0xf1, /* 0xe0-0xe7 */
00110      0x11, 0xf8, 0x04, 0xf5, 0x05, 0xf6, 0x06, 0xf7, /* 0xe8-0xef */
00111      0x12, 0xf9, 0x13, 0xfa, 0x17, 0xfe, 0x14, 0xfb, /* 0xf0-0xf7 */
00112      0x15, 0xfc, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xf8-0xff */
00113  };
00114

```



```

00115 static int
00116 tcvn_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00117 {
00118     unsigned char c = 0;
00119     if (wc < 0x0080 && (wc >= 0x0020 || (0x00fe0076 & (1 << wc)) == 0)) {
00120         *r = wc;
00121         return 1;
00122     }
00123     else if (wc >= 0x00a0 && wc < 0x01b8)
00124         c = tcvn_page00[wc-0x00a0];
00125     else if (wc >= 0x0300 && wc < 0x0328)
00126         c = tcvn_page03[wc-0x0300];
00127     else if (wc >= 0x1ea0 && wc < 0x1f00)
00128         c = tcvn_page1e[wc-0x1ea0];
00129     if (c != 0) {
00130         *r = c;
00131         return 1;
00132     }
00133     return RET_ILSEQ;
00134 }

```

12.295 tis620.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/tis620.h,v 1.4 2001/02/09 00:02:54 dawes Exp $ */
00002
00003 /*
00004  * TIS620-0
00005  */
00006
00007 static const unsigned short tis620_2uni[96] = {
00008     /* 0xa0 */
00009     0xfffd, 0x0e01, 0x0e02, 0x0e03, 0x0e04, 0x0e05, 0x0e06, 0x0e07,
00010     0x0e08, 0x0e09, 0x0e0a, 0x0e0b, 0x0e0c, 0x0e0d, 0x0e0e, 0x0e0f,
00011     /* 0xb0 */
00012     0x0e10, 0x0e11, 0x0e12, 0x0e13, 0x0e14, 0x0e15, 0x0e16, 0x0e17,
00013     0x0e18, 0x0e19, 0x0e1a, 0x0e1b, 0x0e1c, 0x0e1d, 0x0e1e, 0x0e1f,
00014     /* 0xc0 */
00015     0x0e20, 0x0e21, 0x0e22, 0x0e23, 0x0e24, 0x0e25, 0x0e26, 0x0e27,
00016     0x0e28, 0x0e29, 0x0e2a, 0x0e2b, 0x0e2c, 0x0e2d, 0x0e2e, 0x0e2f,
00017     /* 0xd0 */
00018     0x0e30, 0x0e31, 0x0e32, 0x0e33, 0x0e34, 0x0e35, 0x0e36, 0x0e37,
00019     0x0e38, 0x0e39, 0x0e3a, 0xfffd, 0xfffd, 0xfffd, 0xfffd, 0x0e3f,
00020     /* 0xe0 */
00021     0x0e40, 0x0e41, 0x0e42, 0x0e43, 0x0e44, 0x0e45, 0x0e46, 0x0e47,
00022     0x0e48, 0x0e49, 0x0e4a, 0x0e4b, 0x0e4c, 0x0e4d, 0x0e4e, 0x0e4f,
00023     /* 0xf0 */
00024     0x0e50, 0x0e51, 0x0e52, 0x0e53, 0x0e54, 0x0e55, 0x0e56, 0x0e57,
00025     0x0e58, 0x0e59, 0x0e5a, 0x0e5b, 0xfffd, 0xfffd, 0xfffd, 0xfffd,
00026 };
00027
00028 static int
00029 tis620_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00030 {
00031     unsigned char c = *s;
00032     if (c < 0x80) {
00033         *pwc = (ucs4_t) c;
00034         return 1;
00035     }
00036     else if (c < 0xa0) {
00037     }
00038     else {
00039         unsigned short wc = tis620_2uni[c-0xa0];
00040         if (wc != 0xfffd) {
00041             *pwc = (ucs4_t) wc;
00042             return 1;
00043         }
00044     }
00045     return RET_ILSEQ;
00046 }
00047
00048 static const unsigned char tis620_page0e[96] = {
00049     0x00, 0xa1, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, /* 0x00-0x07 */
00050     0xa8, 0xa9, 0xaa, 0xab, 0xac, 0xad, 0xae, 0xaf, /* 0x08-0x0f */
00051     0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5, 0xb6, 0xb7, /* 0x10-0x17 */
00052     0xb8, 0xb9, 0xba, 0xbb, 0xbc, 0xbd, 0xbe, 0xbf, /* 0x18-0x1f */
00053     0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, /* 0x20-0x27 */
00054     0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, /* 0x28-0x2f */
00055     0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, /* 0x30-0x37 */
00056     0xd8, 0xd9, 0xda, 0x00, 0x00, 0x00, 0x00, 0x00, 0xdf, /* 0x38-0x3f */
00057     0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, /* 0x40-0x47 */
00058     0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, /* 0x48-0x4f */
00059     0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, /* 0x50-0x57 */
00060     0xf8, 0xf9, 0xfa, 0xfb, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00061 };
00062

```

```

00063 static int
00064 tis620_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00065 {
00066     unsigned char c = 0;
00067     if (wc < 0x0080) {
00068         *r = wc;
00069         return 1;
00070     }
00071     else if (wc >= 0x0e00 && wc < 0x0e60)
00072         c = tis620_page0e[wc-0x0e00];
00073     if (c != 0) {
00074         *r = c;
00075         return 1;
00076     }
00077     return RET_ILSEQ;
00078 }

```

12.296 ucs2be.h

```

00001 /*
00002  * UCS-2BE = UCS-2 big endian
00003  */
00004 /* $XFree86: xc/lib/X11/lcUniConv/ucs2be.h,v 1.1 2000/11/28 17:25:09 dawes Exp $ */
00005
00006 static int
00007 ucs2be_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00008 {
00009     if (n >= 2) {
00010         if (s[0] >= 0xd8 && s[0] < 0xe0) {
00011             return RET_ILSEQ;
00012         } else {
00013             *pwc = (s[0] << 8) + s[1];
00014             return 2;
00015         }
00016     }
00017     return RET_TOOFEW(0);
00018 }
00019
00020 static int
00021 ucs2be_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00022 {
00023     if (wc < 0x10000 && !(wc >= 0xd800 && wc < 0xe000)) {
00024         if (n >= 2) {
00025             r[0] = (unsigned char) (wc >> 8);
00026             r[1] = (unsigned char) wc;
00027             return 2;
00028         } else
00029             return RET_TOOSMALL;
00030     }
00031     return RET_ILSEQ;
00032 }

```

12.297 utf8.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/utf8.h,v 1.3 2000/11/28 18:50:07 dawes Exp $ */
00002
00003 /*
00004  * UTF-8
00005  */
00006
00007 /* Specification: RFC 2279 */
00008
00009 static int
00010 utf8_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00011 {
00012     unsigned char c = s[0];
00013
00014     if (c < 0x80) {
00015         *pwc = c;
00016         return 1;
00017     } else if (c < 0xc2) {
00018         return RET_ILSEQ;
00019     } else if (c < 0xe0) {
00020         if (n < 2)
00021             return RET_TOOFEW(0);
00022         if (!(s[1] ^ 0x80) < 0x40)
00023             return RET_ILSEQ;
00024         *pwc = ((ucs4_t) (c & 0x1f) << 6)
00025             | (ucs4_t) (s[1] ^ 0x80);
00026         return 2;
00027     } else if (c < 0xf0) {
00028         if (n < 3)

```

```

00029     return RET_TOOFEW(0);
00030     if (!(s[1] ^ 0x80) < 0x40 && (s[2] ^ 0x80) < 0x40
00031         && (c >= 0xe1 || s[1] >= 0xa0)))
00032         return RET_ILSEQ;
00033     *pwc = ((ucs4_t) (c & 0x0f) << 12)
00034         | ((ucs4_t) (s[1] ^ 0x80) << 6)
00035         | (ucs4_t) (s[2] ^ 0x80);
00036     return 3;
00037 } else if (c < 0xf8) {
00038     if (n < 4)
00039         return RET_TOOFEW(0);
00040     if (!(s[1] ^ 0x80) < 0x40 && (s[2] ^ 0x80) < 0x40
00041         && (s[3] ^ 0x80) < 0x40
00042         && (c >= 0xf1 || s[1] >= 0x90)))
00043         return RET_ILSEQ;
00044     *pwc = ((ucs4_t) (c & 0x07) << 18)
00045         | ((ucs4_t) (s[1] ^ 0x80) << 12)
00046         | ((ucs4_t) (s[2] ^ 0x80) << 6)
00047         | (ucs4_t) (s[3] ^ 0x80);
00048     return 4;
00049 } else if (c < 0xfc) {
00050     if (n < 5)
00051         return RET_TOOFEW(0);
00052     if (!(s[1] ^ 0x80) < 0x40 && (s[2] ^ 0x80) < 0x40
00053         && (s[3] ^ 0x80) < 0x40 && (s[4] ^ 0x80) < 0x40
00054         && (c >= 0xf9 || s[1] >= 0x88)))
00055         return RET_ILSEQ;
00056     *pwc = ((ucs4_t) (c & 0x03) << 24)
00057         | ((ucs4_t) (s[1] ^ 0x80) << 18)
00058         | ((ucs4_t) (s[2] ^ 0x80) << 12)
00059         | ((ucs4_t) (s[3] ^ 0x80) << 6)
00060         | (ucs4_t) (s[4] ^ 0x80);
00061     return 5;
00062 } else if (c < 0xfe) {
00063     if (n < 6)
00064         return RET_TOOFEW(0);
00065     if (!(s[1] ^ 0x80) < 0x40 && (s[2] ^ 0x80) < 0x40
00066         && (s[3] ^ 0x80) < 0x40 && (s[4] ^ 0x80) < 0x40
00067         && (s[5] ^ 0x80) < 0x40
00068         && (c >= 0xfd || s[1] >= 0x84)))
00069         return RET_ILSEQ;
00070     *pwc = ((ucs4_t) (c & 0x01) << 30)
00071         | ((ucs4_t) (s[1] ^ 0x80) << 24)
00072         | ((ucs4_t) (s[2] ^ 0x80) << 18)
00073         | ((ucs4_t) (s[3] ^ 0x80) << 12)
00074         | ((ucs4_t) (s[4] ^ 0x80) << 6)
00075         | (ucs4_t) (s[5] ^ 0x80);
00076     return 6;
00077 } else
00078     return RET_ILSEQ;
00079 }
00080
00081 static int
00082 utf8_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n) /* n == 0 is acceptable */
00083 {
00084     int count;
00085     if (wc < 0x80)
00086         count = 1;
00087     else if (wc < 0x800)
00088         count = 2;
00089     else if (wc < 0x10000)
00090         count = 3;
00091     else if (wc < 0x200000)
00092         count = 4;
00093     else if (wc < 0x4000000)
00094         count = 5;
00095     else if (wc <= 0x7fffffff)
00096         count = 6;
00097     else
00098         return RET_ILSEQ;
00099     if (n < count)
00100         return RET_TOOSMALL;
00101     switch (count) { /* note: code falls through cases! */
00102     case 6: r[5] = 0x80 | (wc & 0x3f); wc = wc >> 6; wc |= 0x4000000;
00103     case 5: r[4] = 0x80 | (wc & 0x3f); wc = wc >> 6; wc |= 0x200000;
00104     case 4: r[3] = 0x80 | (wc & 0x3f); wc = wc >> 6; wc |= 0x10000;
00105     case 3: r[2] = 0x80 | (wc & 0x3f); wc = wc >> 6; wc |= 0x800;
00106     case 2: r[1] = 0x80 | (wc & 0x3f); wc = wc >> 6; wc |= 0xc0;
00107     case 1: r[0] = wc;
00108     }
00109     return count;
00110 }

```

12.298 viscii.h

```

00001 /* $XFree86: xc/lib/X11/lcUniConv/viscii.h,v 1.3 2000/11/29 17:40:35 dawes Exp $ */
00002
00003 /*
00004  * VISCI11.1-1
00005  */
00006
00007 /* Specification: RFC 1456 */
00008
00009 static const unsigned short viscii_2uni_1[32] = {
00010     /* 0x00 */
00011     0x0000, 0x0001, 0x1eb2, 0x0003, 0x0004, 0x1eb4, 0x1eaa, 0x0007,
00012     0x0008, 0x0009, 0x000a, 0x000b, 0x000c, 0x000d, 0x000e, 0x000f,
00013     /* 0x10 */
00014     0x0010, 0x0011, 0x0012, 0x0013, 0x1ef6, 0x0015, 0x0016, 0x0017,
00015     0x0018, 0x1ef8, 0x001a, 0x001b, 0x001c, 0x001d, 0x1ef4, 0x001f,
00016 };
00017 static const unsigned short viscii_2uni_2[128] = {
00018     /* 0x80 */
00019     0x1ea0, 0x1eae, 0x1eb0, 0x1eb6, 0x1ea4, 0x1ea6, 0x1ea8, 0x1eac,
00020     0x1ebc, 0x1eb8, 0x1ebe, 0x1ec0, 0x1ec2, 0x1ec4, 0x1ec6, 0x1ed0,
00021     /* 0x90 */
00022     0x1ed2, 0x1ed4, 0x1ed6, 0x1ed8, 0x1ee2, 0x1eda, 0x1edc, 0x1ede,
00023     0x1eca, 0x1ece, 0x1ecc, 0x1ec8, 0x1ee6, 0x0168, 0x1ee4, 0x1ef2,
00024     /* 0xa0 */
00025     0x00d5, 0x1eaf, 0x1eb1, 0x1eb7, 0x1ea5, 0x1ea7, 0x1ea9, 0x1ead,
00026     0x1ebd, 0x1eb9, 0x1ebf, 0x1ec1, 0x1ec3, 0x1ec5, 0x1ec7, 0x1ed1,
00027     /* 0xb0 */
00028     0x1ed3, 0x1ed5, 0x1ed7, 0x1ee0, 0x01a0, 0x1ed9, 0x1edd, 0x1edf,
00029     0x1ecb, 0x1ef0, 0x1ee8, 0x1eea, 0x1eec, 0x01a1, 0x1edb, 0x01af,
00030     /* 0xc0 */
00031     0x00c0, 0x00c1, 0x00c2, 0x00c3, 0x1ea2, 0x0102, 0x1eb3, 0x1eb5,
00032     0x00c8, 0x00c9, 0x00ca, 0x1eba, 0x00cc, 0x00cd, 0x0128, 0x1ef3,
00033     /* 0xd0 */
00034     0x0110, 0x1ee9, 0x00d2, 0x00d3, 0x00d4, 0x1ea1, 0x1ef7, 0x1eeb,
00035     0x1eed, 0x00d9, 0x00da, 0x1ef9, 0x1ef5, 0x00dd, 0x1ee1, 0x01b0,
00036     /* 0xe0 */
00037     0x00e0, 0x00e1, 0x00e2, 0x00e3, 0x1ea3, 0x0103, 0x1eef, 0x1eab,
00038     0x00e8, 0x00e9, 0x00ea, 0x1ebb, 0x00ec, 0x00ed, 0x0129, 0x1ec9,
00039     /* 0xf0 */
00040     0x0111, 0x1ef1, 0x00f2, 0x00f3, 0x00f4, 0x00f5, 0x1ecf, 0x1ecd,
00041     0x1ee5, 0x00f9, 0x00fa, 0x0169, 0x1ee7, 0x00fd, 0x1ee3, 0x1eee,
00042 };
00043
00044 static int
00045 viscii_mbtowc (conv_t conv, ucs4_t *pwc, const unsigned char *s, int n)
00046 {
00047     unsigned char c = *s;
00048     if (c < 0x20)
00049         *pwc = (ucs4_t) viscii_2uni_1[c];
00050     else if (c < 0x80)
00051         *pwc = (ucs4_t) c;
00052     else
00053         *pwc = (ucs4_t) viscii_2uni_2[c-0x80];
00054     return 1;
00055 }
00056
00057 static const unsigned char viscii_page00[64+184] = {
00058     0xc0, 0xc1, 0xc2, 0xc3, 0x00, 0x00, 0x00, 0x00, /* 0xc0-0xc7 */
00059     0xc8, 0xc9, 0xca, 0x00, 0xcc, 0xcd, 0x00, 0x00, /* 0xc8-0xcf */
00060     0x00, 0x00, 0xd2, 0xd3, 0xd4, 0xa0, 0x00, 0x00, /* 0xd0-0xd7 */
00061     0x00, 0xd9, 0xda, 0x00, 0x00, 0xdd, 0x00, 0x00, /* 0xd8-0xdf */
00062     0xe0, 0xe1, 0xe2, 0xe3, 0x00, 0x00, 0x00, 0x00, /* 0xe0-0xe7 */
00063     0xe8, 0xe9, 0xea, 0x00, 0xec, 0xed, 0x00, 0x00, /* 0xe8-0xef */
00064     0x00, 0x00, 0xf2, 0xf3, 0xf4, 0xf5, 0x00, 0x00, /* 0xf0-0xf7 */
00065     0x00, 0xf9, 0xfa, 0x00, 0x00, 0xfd, 0x00, 0x00, /* 0xf8-0xff */
00066     /* 0x0100 */
00067     0x00, 0x00, 0xc5, 0xe5, 0x00, 0x00, 0x00, 0x00, /* 0x00-0x07 */
00068     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x08-0x0f */
00069     0xd0, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x10-0x17 */
00070     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x18-0x1f */
00071     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x20-0x27 */
00072     0xce, 0xee, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x28-0x2f */
00073     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x30-0x37 */
00074     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x38-0x3f */
00075     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x40-0x47 */
00076     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x48-0x4f */
00077     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x50-0x57 */
00078     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x58-0x5f */
00079     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x60-0x67 */
00080     0x9d, 0xfb, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x68-0x6f */
00081     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x70-0x77 */
00082     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x78-0x7f */
00083     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x80-0x87 */
00084     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x88-0x8f */
00085     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x90-0x97 */

```

```

00086 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0x98-0x9f */
00087 0xb4, 0xbd, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xa0-0xa7 */
00088 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xbf, /* 0xa8-0xaf */
00089 0xdf, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xb0-0xb7 */
00090 };
00091 static const unsigned char viscii_pagele[96] = {
00092 0x80, 0xd5, 0xc4, 0xe4, 0x84, 0xa4, 0x85, 0xa5, /* 0xa0-0xa7 */
00093 0x86, 0xa6, 0x06, 0xe7, 0x87, 0xa7, 0x81, 0xa1, /* 0xa8-0xaf */
00094 0x82, 0xa2, 0x02, 0xc6, 0x05, 0xc7, 0x83, 0xa3, /* 0xb0-0xb7 */
00095 0x89, 0xa9, 0xcb, 0xeb, 0x88, 0xa8, 0x8a, 0xaa, /* 0xb8-0xbf */
00096 0x8b, 0xab, 0x8c, 0xac, 0x8d, 0xad, 0x8e, 0xae, /* 0xc0-0xc7 */
00097 0x9b, 0xef, 0x98, 0xb8, 0x9a, 0xf7, 0x99, 0xf6, /* 0xc8-0xcf */
00098 0x8f, 0xaf, 0x90, 0xb0, 0x91, 0xb1, 0x92, 0xb2, /* 0xd0-0xd7 */
00099 0x93, 0xb5, 0x95, 0xbe, 0x96, 0xb6, 0x97, 0xb7, /* 0xd8-0xdf */
00100 0xb3, 0xde, 0x94, 0xfe, 0x9e, 0xf8, 0x9c, 0xfc, /* 0xe0-0xe7 */
00101 0xba, 0xd1, 0xbb, 0xd7, 0xbc, 0xd8, 0xff, 0xe6, /* 0xe8-0xef */
00102 0xb9, 0xf1, 0x9f, 0xcf, 0x1e, 0xdc, 0x14, 0xd6, /* 0xf0-0xf7 */
00103 0x19, 0xdb, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, /* 0xf8-0xff */
00104 };
00105
00106 static int
00107 viscii_wctomb (conv_t conv, unsigned char *r, ucs4_t wc, int n)
00108 {
00109     unsigned char c = 0;
00110     if (wc < 0x0080 && (wc >= 0x0020 || (0x42100064 & (1 << wc)) == 0)) {
00111         *r = wc;
00112         return 1;
00113     }
00114     else if (wc >= 0x00c0 && wc < 0x01b8)
00115         c = viscii_page00[wc-0x00c0];
00116     else if (wc >= 0x1ea0 && wc < 0x1f00)
00117         c = viscii_pagele[wc-0x1ea0];
00118     if (c != 0) {
00119         *r = c;
00120         return 1;
00121     }
00122     return RET_ILSEQ;
00123 }

```

12.299 mk_wcwidth.c

```

00001 /*
00002  * FLTK: Important!
00003  * This file should remain as close to Markus Kuhn's original source
00004  * as possible for easy checking for changes later, however unlikely.
00005  * All customisations to work with FLTK shall be annotated!
00006  */
00007
00008 /*
00009  * This is an implementation of wcwidth() and wcswidth() (defined in
00010  * IEEE Std 1002.1-2001) for Unicode.
00011  *
00012  * http://www.opengroup.org/onlinepubs/007904975/functions/wcwidth.html
00013  * http://www.opengroup.org/onlinepubs/007904975/functions/wcswidth.html
00014  *
00015  * In fixed-width output devices, Latin characters all occupy a single
00016  * "cell" position of equal width, whereas ideographic CJK characters
00017  * occupy two such cells. Interoperability between terminal-line
00018  * applications and (teletype-style) character terminals using the
00019  * UTF-8 encoding requires agreement on which character should advance
00020  * the cursor by how many cell positions. No established formal
00021  * standards exist at present on which Unicode character shall occupy
00022  * how many cell positions on character terminals. These routines are
00023  * a first attempt of defining such behavior based on simple rules
00024  * applied to data provided by the Unicode Consortium.
00025  *
00026  * For some graphical characters, the Unicode standard explicitly
00027  * defines a character-cell width via the definition of the East Asian
00028  * FullWidth (F), Wide (W), Half-width (H), and Narrow (Na) classes.
00029  * In all these cases, there is no ambiguity about which width a
00030  * terminal shall use. For characters in the East Asian Ambiguous (A)
00031  * class, the width choice depends purely on a preference of backward
00032  * compatibility with either historic CJK or Western practice.
00033  * Choosing single-width for these characters is easy to justify as
00034  * the appropriate long-term solution, as the CJK practice of
00035  * displaying these characters as double-width comes from historic
00036  * implementation simplicity (8-bit encoded characters were displayed
00037  * single-width and 16-bit ones double-width, even for Greek,
00038  * Cyrillic, etc.) and not any typographic considerations.
00039  *
00040  * Much less clear is the choice of width for the Not East Asian
00041  * (Neutral) class. Existing practice does not dictate a width for any
00042  * of these characters. It would nevertheless make sense
00043  * typographically to allocate two character cells to characters such
00044  * as for instance EM SPACE or VOLUME INTEGRAL, which cannot be

```

```

00045 * represented adequately with a single-width glyph. The following
00046 * routines at present merely assign a single-cell width to all
00047 * neutral characters, in the interest of simplicity. This is not
00048 * entirely satisfactory and should be reconsidered before
00049 * establishing a formal standard in this area. At the moment, the
00050 * decision which Not East Asian (Neutral) characters should be
00051 * represented by double-width glyphs cannot yet be answered by
00052 * applying a simple rule from the Unicode database content. Setting
00053 * up a proper standard for the behavior of UTF-8 character terminals
00054 * will require a careful analysis not only of each Unicode character,
00055 * but also of each presentation form, something the author of these
00056 * routines has avoided to do so far.
00057 *
00058 * http://www.unicode.org/unicode/reports/tr11/
00059 *
00060 * Markus Kuhn -- 2007-05-26 (Unicode 5.0)
00061 *
00062 * Permission to use, copy, modify, and distribute this software
00063 * for any purpose and without fee is hereby granted. The author
00064 * disclaims all warranties with regard to this software.
00065 *
00066 * Latest version: http://www.cl.cam.ac.uk/~mgk25/ucs/wcwidth.c
00067 */
00068
00069 /*
00070 * FLTK - avoid possible problems on systems with 32-bit wchar_t.
00071 *      Don't include wchar.h, and change wchar_t to unsigned int.
00072 *      Can we guarantee sizeof(unsigned int) >= 4 ?
00073 */
00074 #if 0
00075 #include <wchar.h>
00076 #endif
00077
00078 struct interval {
00079     unsigned int first;
00080     unsigned int last;
00081 };
00082
00083 /* auxiliary function for binary search in interval table */
00084 /*
00085 * FLTK: was
00086 static int bisearch(wchar_t ucs, const struct interval *table, int max) {
00087 */
00088 static int bisearch(unsigned int ucs, const struct interval *table, int max) {
00089     int min = 0;
00090     int mid;
00091
00092     if (ucs < table[0].first || ucs > table[max].last)
00093         return 0;
00094     while (max >= min) {
00095         mid = (min + max) / 2;
00096         if (ucs > table[mid].last)
00097             min = mid + 1;
00098         else if (ucs < table[mid].first)
00099             max = mid - 1;
00100         else
00101             return 1;
00102     }
00103     return 0;
00104 }
00105 }
00106
00107
00108 /* The following two functions define the column width of an ISO 10646
00109 * character as follows:
00110 *
00111 *     - The null character (U+0000) has a column width of 0.
00112 *
00113 *     - Other C0/C1 control characters and DEL will lead to a return
00114 *       value of -1.
00115 *
00116 *     - Non-spacing and enclosing combining characters (general
00117 *       category code Mn or Me in the Unicode database) have a
00118 *       column width of 0.
00119 *
00120 *     - SOFT HYPHEN (U+00AD) has a column width of 1.
00121 *
00122 *     - Other format characters (general category code Cf in the Unicode
00123 *       database) and ZERO WIDTH SPACE (U+200B) have a column width of 0.
00124 *
00125 *     - Hangul Jamo medial vowels and final consonants (U+1160-U+11FF)
00126 *       have a column width of 0.
00127 *
00128 *     - Spacing characters in the East Asian Wide (W) or East Asian
00129 *       Full-width (F) category as defined in Unicode Technical
00130 *       Report #11 have a column width of 2.
00131 */

```

```

00132 *      - All remaining characters (including all printable
00133 *      ISO 8859-1 and WGL4 characters, Unicode control characters,
00134 *      etc.) have a column width of 1.
00135 *
00136 * This implementation assumes that wchar_t characters are encoded
00137 * in ISO 10646.
00138 */
00139
00140 /*
00141  * FLTK: was
00142  int mk_wcwidth(wchar_t ucs)
00143  */
00144 int mk_wcwidth(unsigned int ucs)
00145 {
00146     /* sorted list of non-overlapping intervals of non-spacing characters */
00147     /* generated by "uniset +cat=Me +cat=Mn +cat=Cf -00AD +1160-11FF +200B c" */
00148     static const struct interval combining[] = {
00149         { 0x0300, 0x036F }, { 0x0483, 0x0486 }, { 0x0488, 0x0489 },
00150         { 0x0591, 0x05BD }, { 0x05BF, 0x05BF }, { 0x05C1, 0x05C2 },
00151         { 0x05C4, 0x05C5 }, { 0x05C7, 0x05C7 }, { 0x0600, 0x0603 },
00152         { 0x0610, 0x0615 }, { 0x064B, 0x065E }, { 0x0670, 0x0670 },
00153         { 0x06D6, 0x06E4 }, { 0x06E7, 0x06E8 }, { 0x06EA, 0x06ED },
00154         { 0x070F, 0x070F }, { 0x0711, 0x0711 }, { 0x0730, 0x074A },
00155         { 0x07A6, 0x07B0 }, { 0x07EB, 0x07F3 }, { 0x0901, 0x0902 },
00156         { 0x093C, 0x093C }, { 0x0941, 0x0948 }, { 0x094D, 0x094D },
00157         { 0x0951, 0x0954 }, { 0x0962, 0x0963 }, { 0x0981, 0x0981 },
00158         { 0x09BC, 0x09BC }, { 0x09C1, 0x09C4 }, { 0x09CD, 0x09CD },
00159         { 0x09E2, 0x09E3 }, { 0x0A01, 0x0A02 }, { 0x0A3C, 0x0A3C },
00160         { 0x0A41, 0x0A42 }, { 0x0A47, 0x0A48 }, { 0x0A4B, 0x0A4D },
00161         { 0x0A70, 0x0A71 }, { 0x0A81, 0x0A82 }, { 0x0ABC, 0x0ABC },
00162         { 0x0AC1, 0x0AC5 }, { 0x0AC7, 0x0AC8 }, { 0x0ACD, 0x0ACD },
00163         { 0x0AE2, 0x0AE3 }, { 0x0B01, 0x0B01 }, { 0x0B3C, 0x0B3C },
00164         { 0x0B3F, 0x0B3F }, { 0x0B41, 0x0B43 }, { 0x0B4D, 0x0B4D },
00165         { 0x0B56, 0x0B56 }, { 0x0B82, 0x0B82 }, { 0x0BC0, 0x0BC0 },
00166         { 0x0BCD, 0x0BCD }, { 0x0C3E, 0x0C40 }, { 0x0C46, 0x0C48 },
00167         { 0x0C4A, 0x0C4D }, { 0x0C55, 0x0C56 }, { 0x0CBC, 0x0CBC },
00168         { 0x0CBF, 0x0CBF }, { 0x0CC6, 0x0CC6 }, { 0x0CCC, 0x0CCD },
00169         { 0x0CE2, 0x0CE3 }, { 0x0D41, 0x0D43 }, { 0x0D4D, 0x0D4D },
00170         { 0x0DCA, 0x0DCA }, { 0x0DD2, 0x0DD4 }, { 0x0DD6, 0x0DD6 },
00171         { 0x0E31, 0x0E31 }, { 0x0E34, 0x0E3A }, { 0x0E47, 0x0E4E },
00172         { 0x0EB1, 0x0EB1 }, { 0x0EB4, 0x0EB9 }, { 0x0EBB, 0x0EBB },
00173         { 0x0EC8, 0x0ECD }, { 0x0F18, 0x0F19 }, { 0x0F35, 0x0F35 },
00174         { 0x0F37, 0x0F37 }, { 0x0F39, 0x0F39 }, { 0x0F71, 0x0F7E },
00175         { 0x0F80, 0x0F84 }, { 0x0F86, 0x0F87 }, { 0x0F90, 0x0F97 },
00176         { 0x0F99, 0x0FBC }, { 0x0FC6, 0x0FC6 }, { 0x102D, 0x1030 },
00177         { 0x1032, 0x1032 }, { 0x1036, 0x1037 }, { 0x1039, 0x1039 },
00178         { 0x1058, 0x1059 }, { 0x1160, 0x11FF }, { 0x135F, 0x135F },
00179         { 0x1712, 0x1714 }, { 0x1732, 0x1734 }, { 0x1752, 0x1753 },
00180         { 0x1772, 0x1773 }, { 0x17B4, 0x17B5 }, { 0x17B7, 0x17BD },
00181         { 0x17C6, 0x17C6 }, { 0x17C9, 0x17D3 }, { 0x17DD, 0x17DD },
00182         { 0x180B, 0x180D }, { 0x18A9, 0x18A9 }, { 0x1920, 0x1922 },
00183         { 0x1927, 0x1928 }, { 0x1932, 0x1932 }, { 0x1939, 0x193B },
00184         { 0x1A17, 0x1A18 }, { 0x1B00, 0x1B03 }, { 0x1B34, 0x1B34 },
00185         { 0x1B36, 0x1B3A }, { 0x1B3C, 0x1B3C }, { 0x1B42, 0x1B42 },
00186         { 0x1B6B, 0x1B73 }, { 0x1DC0, 0x1DCA }, { 0x1DFE, 0x1DFE },
00187         { 0x200B, 0x200F }, { 0x202A, 0x202E }, { 0x2060, 0x2063 },
00188         { 0x206A, 0x206F }, { 0x20D0, 0x20EF }, { 0x302A, 0x302F },
00189         { 0x3099, 0x309A }, { 0xA806, 0xA806 }, { 0xA80B, 0xA80B },
00190         { 0xA825, 0xA826 }, { 0xFB1E, 0xFB1E }, { 0xFE00, 0xFE0F },
00191         { 0xFE20, 0xFE23 }, { 0xFEFF, 0xFEFF }, { 0xFFFF, 0xFFFF },
00192         { 0x10A01, 0x10A03 }, { 0x10A05, 0x10A06 }, { 0x10A0C, 0x10A0F },
00193         { 0x10A38, 0x10A3A }, { 0x10A3F, 0x10A3F }, { 0x1D167, 0x1D169 },
00194         { 0x1D173, 0x1D182 }, { 0x1D185, 0x1D18B }, { 0x1D1AA, 0x1D1AD },
00195         { 0x1D242, 0x1D244 }, { 0xE0001, 0xE0001 }, { 0xE0020, 0xE007F },
00196         { 0xE0100, 0xE01EF }
00197     };
00198
00199     /* test for 8-bit control characters */
00200     if (ucs == 0)
00201         return 0;
00202     if (ucs < 32 || (ucs >= 0x7f && ucs < 0xa0))
00203         return -1;
00204
00205     /* binary search in table of non-spacing characters */
00206     if (bsearch(ucs, combining,
00207                 sizeof(combining) / sizeof(struct interval) - 1))
00208         return 0;
00209
00210     /* if we arrive here, ucs is not a combining or C0/C1 control character */
00211
00212     return 1 +
00213         (ucs >= 0x1100 &&
00214          (ucs <= 0x115f || /* Hangul Jamo init. consonants */
00215           ucs == 0x2329 || ucs == 0x232a ||
00216           (ucs >= 0x2e80 && ucs <= 0xa4cf &&
00217            ucs != 0x303f) || /* CJK ... Yi */
00218            (ucs >= 0xac00 && ucs <= 0xd7a3) || /* Hangul Syllables */

```

```

00219     (ucs >= 0xf900 && ucs <= 0xfaff) || /* CJK Compatibility Ideographs */
00220     (ucs >= 0xfe10 && ucs <= 0xfe19) || /* Vertical forms */
00221     (ucs >= 0xfe30 && ucs <= 0xfe6f) || /* CJK Compatibility Forms */
00222     (ucs >= 0xff00 && ucs <= 0xff60) || /* Fullwidth Forms */
00223     (ucs >= 0xffe0 && ucs <= 0xffe6) ||
00224     (ucs >= 0x20000 && ucs <= 0x2ffffd) ||
00225     (ucs >= 0x30000 && ucs <= 0x3ffffd));
00226 }
00227
00228
00229 /*
00230  * FLTK: comment out the remaining functions, as we don't need them.
00231  */
00232 #if 0
00233
00234 /*
00235  * FLTK: was
00236  int mk_wcswidth(const wchar_t *pwcs, size_t n)
00237  */
00238  int mk_wcswidth(const unsigned int *pwcs, size_t n)
00239  {
00240      int w, width = 0;
00241
00242      for (; *pwcs && n-- > 0; pwcs++)
00243          if ((w = mk_wcwidth(*pwcs)) < 0)
00244              return -1;
00245          else
00246              width += w;
00247
00248      return width;
00249  }
00250
00251
00252 /*
00253  * The following functions are the same as mk_wcwidth() and
00254  * mk_wcswidth(), except that spacing characters in the East Asian
00255  * Ambiguous (A) category as defined in Unicode Technical Report #11
00256  * have a column width of 2. This variant might be useful for users of
00257  * CJK legacy encodings who want to migrate to UCS without changing
00258  * the traditional terminal character-width behaviour. It is not
00259  * otherwise recommended for general use.
00260  */
00261 /*
00262  * FLTK: was
00263  int mk_wcwidth_cjk(wchar_t ucs)
00264  */
00265  int mk_wcwidth_cjk(unsigned int ucs)
00266  {
00267      /* sorted list of non-overlapping intervals of East Asian Ambiguous
00268      * characters, generated by "uniset +WIDTH-A -cat=Me -cat=Mn -cat=Cf c" */
00269      static const struct interval ambiguous[] = {
00270          { 0x00A1, 0x00A1 }, { 0x00A4, 0x00A4 }, { 0x00A7, 0x00A8 },
00271          { 0x00AA, 0x00AA }, { 0x00AE, 0x00AE }, { 0x00B0, 0x00B4 },
00272          { 0x00B6, 0x00BA }, { 0x00BC, 0x00BF }, { 0x00C6, 0x00C6 },
00273          { 0x00D0, 0x00D0 }, { 0x00D7, 0x00D8 }, { 0x00DE, 0x00E1 },
00274          { 0x00E6, 0x00E6 }, { 0x00E8, 0x00EA }, { 0x00EC, 0x00ED },
00275          { 0x00F0, 0x00F0 }, { 0x00F2, 0x00F3 }, { 0x00F7, 0x00FA },
00276          { 0x00FC, 0x00FC }, { 0x00FE, 0x00FE }, { 0x0101, 0x0101 },
00277          { 0x0111, 0x0111 }, { 0x0113, 0x0113 }, { 0x011B, 0x011B },
00278          { 0x0126, 0x0127 }, { 0x012B, 0x012B }, { 0x0131, 0x0133 },
00279          { 0x0138, 0x0138 }, { 0x013F, 0x0142 }, { 0x0144, 0x0144 },
00280          { 0x0148, 0x014B }, { 0x014D, 0x014D }, { 0x0152, 0x0153 },
00281          { 0x0166, 0x0167 }, { 0x016B, 0x016B }, { 0x01CE, 0x01CE },
00282          { 0x01D0, 0x01D0 }, { 0x01D2, 0x01D2 }, { 0x01D4, 0x01D4 },
00283          { 0x01D6, 0x01D6 }, { 0x01D8, 0x01D8 }, { 0x01DA, 0x01DA },
00284          { 0x01DC, 0x01DC }, { 0x0251, 0x0251 }, { 0x0261, 0x0261 },
00285          { 0x02C4, 0x02C4 }, { 0x02C7, 0x02C7 }, { 0x02C9, 0x02CB },
00286          { 0x02CD, 0x02CD }, { 0x02D0, 0x02D0 }, { 0x02D8, 0x02DB },
00287          { 0x02DD, 0x02DD }, { 0x02DF, 0x02DF }, { 0x0391, 0x03A1 },
00288          { 0x03A3, 0x03A9 }, { 0x03B1, 0x03C1 }, { 0x03C3, 0x03C9 },
00289          { 0x0401, 0x0401 }, { 0x0410, 0x044F }, { 0x0451, 0x0451 },
00290          { 0x2010, 0x2010 }, { 0x2013, 0x2016 }, { 0x2018, 0x2019 },
00291          { 0x201C, 0x201D }, { 0x2020, 0x2022 }, { 0x2024, 0x2027 },
00292          { 0x2030, 0x2030 }, { 0x2032, 0x2033 }, { 0x2035, 0x2035 },
00293          { 0x203B, 0x203B }, { 0x203E, 0x203E }, { 0x2074, 0x2074 },
00294          { 0x207F, 0x207F }, { 0x2081, 0x2084 }, { 0x20AC, 0x20AC },
00295          { 0x2103, 0x2103 }, { 0x2105, 0x2105 }, { 0x2109, 0x2109 },
00296          { 0x2113, 0x2113 }, { 0x2116, 0x2116 }, { 0x2121, 0x2122 },
00297          { 0x2126, 0x2126 }, { 0x212B, 0x212B }, { 0x2153, 0x2154 },
00298          { 0x215B, 0x215E }, { 0x2160, 0x216B }, { 0x2170, 0x2179 },
00299          { 0x2190, 0x2199 }, { 0x21B8, 0x21B9 }, { 0x21D2, 0x21D2 },
00300          { 0x21D4, 0x21D4 }, { 0x21E7, 0x21E7 }, { 0x2200, 0x2200 },
00301          { 0x2202, 0x2203 }, { 0x2207, 0x2208 }, { 0x220B, 0x220B },
00302          { 0x220F, 0x220F }, { 0x2211, 0x2211 }, { 0x2215, 0x2215 },
00303          { 0x221A, 0x221A }, { 0x221D, 0x2220 }, { 0x2223, 0x2223 },
00304          { 0x2225, 0x2225 }, { 0x2227, 0x222C }, { 0x222E, 0x222E },
00305          { 0x2234, 0x2237 }, { 0x223C, 0x223D }, { 0x2248, 0x2248 },

```



```

00306     { 0x224C, 0x224C }, { 0x2252, 0x2252 }, { 0x2260, 0x2261 },
00307     { 0x2264, 0x2267 }, { 0x226A, 0x226B }, { 0x226E, 0x226F },
00308     { 0x2282, 0x2283 }, { 0x2286, 0x2287 }, { 0x2295, 0x2295 },
00309     { 0x2299, 0x2299 }, { 0x22A5, 0x22A5 }, { 0x22BF, 0x22BF },
00310     { 0x2312, 0x2312 }, { 0x2460, 0x24E9 }, { 0x24EB, 0x254B },
00311     { 0x2550, 0x2573 }, { 0x2580, 0x258F }, { 0x2592, 0x2595 },
00312     { 0x25A0, 0x25A1 }, { 0x25A3, 0x25A9 }, { 0x25B2, 0x25B3 },
00313     { 0x25B6, 0x25B7 }, { 0x25BC, 0x25BD }, { 0x25C0, 0x25C1 },
00314     { 0x25C6, 0x25C8 }, { 0x25CB, 0x25CB }, { 0x25CE, 0x25D1 },
00315     { 0x25E2, 0x25E5 }, { 0x25EF, 0x25EF }, { 0x2605, 0x2606 },
00316     { 0x2609, 0x2609 }, { 0x260E, 0x260F }, { 0x2614, 0x2615 },
00317     { 0x261C, 0x261C }, { 0x261E, 0x261E }, { 0x2640, 0x2640 },
00318     { 0x2642, 0x2642 }, { 0x2660, 0x2661 }, { 0x2663, 0x2665 },
00319     { 0x2667, 0x266A }, { 0x266C, 0x266D }, { 0x266F, 0x266F },
00320     { 0x273D, 0x273D }, { 0x2776, 0x277F }, { 0xE000, 0xF8FF },
00321     { 0xFFFF, 0xFFFF }, { 0xF000, 0xFFFF }, { 0x10000, 0x10FFFF }
00322 };
00323
00324 /* binary search in table of non-spacing characters */
00325 if (bisearch(ucs, ambiguous,
00326             sizeof(ambiguous) / sizeof(struct interval) - 1))
00327     return 2;
00328
00329 return mk_wcwidth(ucs);
00330 }
00331
00332
00333 /*
00334  * FLTK: was
00335  int mk_wcswidth_cjk(const wchar_t *pwcs, size_t n)
00336  */
00337 int mk_wcswidth_cjk(const unsigned int *pwcs, size_t n)
00338 {
00339     int w, width = 0;
00340
00341     for (; *pwcs && n-- > 0; pwcs++)
00342         if ((w = mk_wcwidth_cjk(*pwcs)) < 0)
00343             return -1;
00344         else
00345             width += w;
00346
00347     return width;
00348 }
00349
00350 /*
00351  * FLTK: end of commented out functions
00352  */
00353 #endif

```

12.300 ucs2fontmap.c

```

00001 /*
00002  * Author: Jean-Marc Lienher ( http://oksid.ch )
00003  * Copyright 2000-2003 by O'ksi'D.
00004  *
00005  * This library is free software. Distribution and use rights are outlined in
00006  * the file "COPYING" which should have been included with this file. If this
00007  * file is missing or damaged, see the license at:
00008  *
00009  *     https://www.fltk.org/COPYING.php
00010  *
00011  * Please see the following page on how to report bugs and issues:
00012  *
00013  *     https://www.fltk.org/bugs.php
00014  */
00015
00016 #include <stdlib.h>
00017 #include <string.h>
00018
00019 #define RET_ILSEQ -1
00020 #define RET_TOOFEW(x) (-10 - x)
00021 #define RET_TOOSMALL -2
00022 #define conv_t void*
00023 #define ucs4_t unsigned int
00024 typedef struct {
00025     unsigned short indx;
00026     unsigned short used;
00027 } Summary16;
00028
00029 #define NEED_TOMB /* indicates what part of these include files is needed here (avoid compilation
00030                  warnings) */
00030 #include "lcUniConv/cp936ext.h"
00031 #include "lcUniConv/big5.h"
00032 #include "lcUniConv/gb2312.h"
00033 #include "lcUniConv/iso8859_10.h"

```

```
00034 #include "lcUniConv/iso8859_11.h"
00035 #include "lcUniConv/iso8859_13.h"
00036 #include "lcUniConv/iso8859_14.h"
00037 #include "lcUniConv/iso8859_15.h"
00038 #include "lcUniConv/iso8859_2.h"
00039 #include "lcUniConv/iso8859_3.h"
00040 #include "lcUniConv/iso8859_4.h"
00041 #include "lcUniConv/iso8859_5.h"
00042 #include "lcUniConv/iso8859_6.h"
00043 #include "lcUniConv/iso8859_7.h"
00044 #include "lcUniConv/iso8859_8.h"
00045 #include "lcUniConv/iso8859_9.h"
00046 #include "lcUniConv/jisx0201.h"
00047 #include "lcUniConv/jisx0208.h"
00048 #include "lcUniConv/jisx0212.h"
00049 #include "lcUniConv/koi8_r.h"
00050 #include "lcUniConv/koi8_u.h"
00051 #include "lcUniConv/ksc5601.h"
00052 #include "lcUniConv/cp1251.h"
00053 #include "headers/symbol_.h"
00054 #include "headers/dingbats_.h"
00055
00056 /***** conv_gen.c *****/
00057
00058 /*const*/
00059 static int ucs2fontmap(char *s, unsigned int ucs, int enc) {
00060     switch(enc) {
00061         case 0: /* iso10646-1 */
00062             s[0] = (char) ((ucs & 0xFF00) >> 8);
00063             s[1] = (char) (ucs & 0xFF);
00064             return 0;
00065         case 1: /* iso8859-1 */
00066             if (ucs <= 0x00FF) {
00067                 if (ucs >= 0x0001) {
00068                     s[0] = 0;
00069                     s[1] = (char) (ucs & 0xFF);
00070                     return 1;
00071                 }
00072             }
00073             break;
00074         case 2: /* iso8859-2 */
00075             if (ucs <= 0x00a0) {
00076                 s[0] = 0;
00077                 s[1] = (char) ucs;
00078                 return 2;
00079             } else if (ucs < 0x0180) {
00080                 if (ucs >= 0x00a0) {
00081                     s[0] = 0;
00082                     s[1] = (char) iso8859_2_page00[ucs-0x00a0];
00083                     if (s[1]) return 2;
00084                 }
00085             } else if (ucs < 0x02e0) {
00086                 if (ucs >= 0x02c0) {
00087                     s[0] = 0;
00088                     s[1] = (char) iso8859_2_page02[ucs-0x02c0];
00089                     if (s[1]) return 2;
00090                 }
00091             }
00092             break;
00093         case 3: /* iso8859-3 */
00094             if (iso8859_3_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00095                 return 3;
00096             }
00097             break;
00098         case 4: /* iso8859-4 */
00099             if (iso8859_4_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00100                 return 4;
00101             }
00102             break;
00103         case 5: /* iso8859-5 */
00104             if (iso8859_5_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00105                 return 5;
00106             }
00107             break;
00108         case 6: /* iso8859-6 */
00109             if (iso8859_6_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00110                 return 6;
00111             }
00112             break;
00113         case 7: /* iso8859-7 */
00114             if (iso8859_7_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00115                 return 7;
00116             }
00117             break;
00118         case 8: /* iso8859-8 */
00119             if (iso8859_8_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00120                 return 8;
00121             }
00122     }
```

```
00121     }
00122     break;
00123 case 9:      /* iso8859-9 */
00124     if (iso8859_9_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00125         return 9;
00126     }
00127     break;
00128 case 10:     /* iso8859-10 */
00129     if (iso8859_10_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00130         return 10;
00131     }
00132     break;
00133 case 25:     /* iso8859-11 */
00134     if (iso8859_11_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00135         return 25;
00136     }
00137     break;
00138 case 11:     /* iso8859-13 */
00139     if (iso8859_13_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00140         return 11;
00141     }
00142     break;
00143 case 12:     /* iso8859-14 */
00144     if (iso8859_14_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00145         return 12;
00146     }
00147     break;
00148 case 13:     /* iso8859-15 */
00149     if (iso8859_15_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00150         return 13;
00151     }
00152     break;
00153 case 14:     /* koi8-r */
00154     if (koi8_r_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00155         return 14;
00156     }
00157     break;
00158 case 15:     /* big5 */
00159     if (big5_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00160         return 15;
00161     }
00162     break;
00163 case 16:     /* ksc5601.1987-0 */
00164     if (ksc5601_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00165         return 16;
00166     }
00167     break;
00168 case 17:     /* gb2312.1980-0 */
00169     if (gb2312_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00170         return 17;
00171     }
00172     break;
00173 case 18:     /* jisx0201.1976-0 */
00174     if (jisx0201_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00175         return 18;
00176     }
00177     break;
00178 case 19:     /* jisx0208.1983-0 */
00179     if (jisx0208_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00180         return 19;
00181     }
00182     break;
00183 case 20:     /* jisx0212.1990-0 */
00184     if (jisx0212_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00185         return 20;
00186     }
00187     break;
00188 case 21:     /* symbol */
00189     if (ucs <= 0x00F7) {
00190         if (ucs >= 0x0020) {
00191             s[0] = 0;
00192             s[1] = unicode_to_symbol_lb_0020[ucs - 0x0020];
00193             if (s[1]) return 21;
00194         }
00195     } else if (ucs <= 0x0192) {
00196         if (ucs >= 0x0192) {
00197             s[0] = 0;
00198             s[1] = unicode_to_symbol_lb_0192[ucs - 0x0192];
00199             if (s[1]) return 21;
00200         }
00201     } else if (ucs <= 0x03D6) {
00202         if (ucs >= 0x0391) {
00203             s[0] = 0;
00204             s[1] = unicode_to_symbol_lb_0391[ucs - 0x0391];
00205             if (s[1]) return 21;
00206         }
00207     } else if (ucs <= 0x232A) {
```

```

00208     if (ucs >= 0x2022) {
00209         s[0] = 0;
00210         s[1] = unicode_to_symbol_1b_2022[ucs - 0x2022];
00211         if (s[1]) return 21;
00212     }
00213 } else if (ucs <= 0x25CA) {
00214     if (ucs >= 0x25CA) {
00215         s[0] = 0;
00216         s[1] = unicode_to_symbol_1b_25CA[ucs - 0x25CA];
00217         if (s[1]) return 21;
00218     }
00219 } else if (ucs <= 0x2666) {
00220     if (ucs >= 0x2660) {
00221         s[0] = 0;
00222         s[1] = unicode_to_symbol_1b_2660[ucs - 0x2660];
00223         if (s[1]) return 21;
00224     }
00225 } else if (ucs <= 0xF6DB) {
00226     if (ucs >= 0xF6D9) {
00227         s[0] = 0;
00228         s[1] = unicode_to_symbol_1b_F6D9[ucs - 0xF6D9];
00229         if (s[1]) return 21;
00230     }
00231 } else if (ucs <= 0xF8FE) {
00232     if (ucs >= 0xF8E5) {
00233         s[0] = 0;
00234         s[1] = unicode_to_symbol_1b_F8E5[ucs - 0xF8E5];
00235         if (s[1]) return 21;
00236     }
00237 }
00238 break;
00239 case 22: /* dingbats */
00240     if (ucs <= 0x00A0) {
00241         if (ucs >= 0x0020) {
00242             s[0] = 0;
00243             s[1] = unicode_to_dingbats_1b_0020[ucs - 0x0020];
00244             if (s[1]) return 22;
00245         }
00246     } else if (ucs <= 0x2195) {
00247         if (ucs >= 0x2192) {
00248             s[0] = 0;
00249             s[1] = unicode_to_dingbats_1b_2192[ucs - 0x2192];
00250             if (s[1]) return 22;
00251         }
00252     } else if (ucs <= 0x2469) {
00253         if (ucs >= 0x2460) {
00254             s[0] = 0;
00255             s[1] = unicode_to_dingbats_1b_2460[ucs - 0x2460];
00256             if (s[1]) return 22;
00257         }
00258     } else if (ucs <= 0x2666) {
00259         if (ucs >= 0x25A0) {
00260             s[0] = 0;
00261             s[1] = unicode_to_dingbats_1b_25A0[ucs - 0x25A0];
00262             if (s[1]) return 22;
00263         }
00264     } else if (ucs <= 0x27BE) {
00265         if (ucs >= 0x2701) {
00266             s[0] = 0;
00267             s[1] = unicode_to_dingbats_1b_2701[ucs - 0x2701];
00268             if (s[1]) return 22;
00269         }
00270     } else if (ucs <= 0xF8E4) {
00271         if (ucs >= 0xF8D7) {
00272             s[0] = 0;
00273             s[1] = unicode_to_dingbats_1b_F8D7[ucs - 0xF8D7];
00274             if (s[1]) return 22;
00275         }
00276     }
00277     break;
00278 case 23: /* koi8-u */
00279     if (koi8_u_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00280         return 23;
00281     }
00282     break;
00283 case 24: /* microsoft-cp1251 */
00284     if (cp1251_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00285         return 24;
00286     }
00287     break;
00288 case 26: /* gbkc/936ext */
00289     if (cp936ext_wctomb(NULL, (unsigned char*)s, ucs, 2) > 0) {
00290         return 26;
00291     }
00292     break;
00293 default:
00294     break;

```

```

00295     };
00296     return -1;
00297 }
00298
00299 /*const*/
00300 static int encoding_number(const char *enc) {
00301     if (!enc || !strcmp(enc, "iso10646-1", 10)) {
00302         return 0;
00303     } else if (!strcmp(enc, "iso8859-1")) {
00304         return 1;
00305     } else if (!strcmp(enc, "iso8859-2")) {
00306         return 2;
00307     } else if (!strcmp(enc, "iso8859-3")) {
00308         return 3;
00309     } else if (!strcmp(enc, "iso8859-4")) {
00310         return 4;
00311     } else if (!strcmp(enc, "iso8859-5")) {
00312         return 5;
00313     } else if (!strcmp(enc, "iso8859-6")) {
00314         return 6;
00315     } else if (!strcmp(enc, "iso8859-7")) {
00316         return 7;
00317     } else if (!strcmp(enc, "iso8859-8")) {
00318         return 8;
00319     } else if (!strcmp(enc, "iso8859-9")) {
00320         return 9;
00321     } else if (!strcmp(enc, "iso8859-10")) {
00322         return 10;
00323     } else if (!strcmp(enc, "iso8859-13")) {
00324         return 11;
00325     } else if (!strcmp(enc, "iso8859-14")) {
00326         return 12;
00327     } else if (!strcmp(enc, "iso8859-15")) {
00328         return 13;
00329     } else if (!strcmp(enc, "koi8-r")) {
00330         return 14;
00331     } else if (!strcmp(enc, "big5-0") || !strcmp(enc, "big5.eten-0") ||
00332         !strcmp(enc, "big5p-0"))
00333     {
00334         return 15;
00335     } else if (!strcmp(enc, "ksc5601.1987-0")) {
00336         return 16;
00337     } else if (!strcmp(enc, "gb2312.1980-0") || !strcmp(enc, "gb2312.80-0") ||
00338         !strcmp(enc, "gb2312.80&gb8565.88") || !strcmp(enc, "gb2312.80-0"))
00339     {
00340         return 17;
00341     } else if (!strcmp(enc, "jisx0201.1976-0")) {
00342         return 18;
00343     } else if (!strcmp(enc, "jisx0208.1983-0") || !strcmp(enc, "jisx0208.1990-0")
00344         || !strcmp(enc, "jisx0208.1978-0"))
00345     {
00346         return 19;
00347     } else if (!strcmp(enc, "jisx0212.1990-0")) {
00348         return 20;
00349     } else if (!strcmp(enc, "symbol")) {
00350         return 21;
00351     } else if (!strcmp(enc, "dingbats") || !strcmp(enc, "zapfdingbats") ||
00352         !strcmp(enc, "zapf dingbats") || !strcmp(enc, "itc zapf dingbats"))
00353     {
00354         return 22;
00355     } else if (!strcmp(enc, "koi8-u")) {
00356         return 23;
00357     } else if (!strcmp(enc, "microsoft-cp1251")) {
00358         return 24;
00359     } else if (!strcmp(enc, "iso8859-11")) {
00360         return 25;
00361     } else if (!strcmp(enc, "gbk-0") || !strcmp(enc, "cp936") || !strcmp(enc, "gbk")) {
00362         return 26;
00363     };
00364     return -1;
00365 }

```

12.301 utf8Utils.c

```

00001 /*
00002  * Author: Jean-Marc Lienher ( http://oksid.ch )
00003  * Copyright 2000-2003 by O'ksi'D.
00004  *
00005  * This library is free software. Distribution and use rights are outlined in
00006  * the file "COPYING" which should have been included with this file. If this
00007  * file is missing or damaged, see the license at:
00008  *
00009  * https://www.fltk.org/COPYING.php
00010  *
00011  * Please see the following page on how to report bugs and issues:

```

```

00012  *
00013  *   https://www.fltk.org/bugs.php
00014  */
00015
00016 /*
00017  * Unicode to UTF-8 conversion functions.
00018  *
00019  * This file is compiled and linked only for X11 w/o Xft.
00020  */
00021
00022 #include "../Xutf8.h"
00023
00024 /** NOTE : all functions are LIMITED to 24 bits Unicode values !!! */
00025
00026 /*
00027  * Converts the first char of the UTF-8 string to an Unicode value
00028  * Returns the byte length of the converted UTF-8 char
00029  * Returns -1 if the UTF-8 string is not valid
00030  */
00031 int
00032 XConvertUtf8ToUcs(const unsigned char *buf,
00033                  int len,
00034                  unsigned int *ucs) {
00035
00036     if (buf[0] & 0x80) {
00037         if (buf[0] & 0x40) {
00038             if (buf[0] & 0x20) {
00039                 if (buf[0] & 0x10) {
00040                     if (buf[0] & 0x08) {
00041                         if (buf[0] & 0x04) {
00042                             if (buf[0] & 0x02) {
00043                                 /* bad UTF-8 string */
00044                             } else {
00045                                 /* 0x04000000 - 0x7FFFFFFF */
00046                             }
00047                         } else if (len > 4
00048                                && (buf[1] & 0xC0) == 0x80
00049                                && (buf[2] & 0xC0) == 0x80
00050                                && (buf[3] & 0xC0) == 0x80
00051                                && (buf[4] & 0xC0) == 0x80) {
00052                             /* 0x00200000 - 0x03FFFFFF */
00053                             *ucs = ((buf[0] & ~0xF8) << 24) +
00054                                   ((buf[1] & ~0x80) << 18) +
00055                                   ((buf[2] & ~0x80) << 12) +
00056                                   ((buf[3] & ~0x80) << 6) +
00057                                   (buf[4] & ~0x80);
00058                             if (*ucs > 0x001FFFFF && *ucs < 0x01000000) return 5;
00059                         }
00060                     } else if (len > 3
00061                            && (buf[1] & 0xC0) == 0x80
00062                            && (buf[2] & 0xC0) == 0x80
00063                            && (buf[3] & 0xC0) == 0x80) {
00064                         /* 0x00010000 - 0x001FFFFF */
00065                         *ucs = ((buf[0] & ~0xF0) << 18) +
00066                               ((buf[1] & ~0x80) << 12) +
00067                               ((buf[2] & ~0x80) << 6) +
00068                               (buf[3] & ~0x80);
00069                         if (*ucs > 0x0000FFFF) return 4;
00070                     }
00071                 } else if (len > 2
00072                        && (buf[1] & 0xC0) == 0x80
00073                        && (buf[2] & 0xC0) == 0x80) {
00074                     /* 0x00000800 - 0x0000FFFF */
00075                     *ucs = ((buf[0] & ~0xE0) << 12) +
00076                           ((buf[1] & ~0x80) << 6) +
00077                           (buf[2] & ~0x80);
00078                     if (*ucs > 0x000007FF) return 3;
00079                 }
00080             } else if (len > 1 && (buf[1] & 0xC0) == 0x80) {
00081                 /* 0x00000080 - 0x000007FF */
00082                 *ucs = ((buf[0] & ~0xC0) << 6) +
00083                       (buf[1] & ~0x80);
00084                 if (*ucs > 0x0000007F) return 2;
00085             }
00086         }
00087     } else if (len > 0) {
00088         /* 0x00000000 - 0x0000007F */
00089         *ucs = buf[0];
00090         return 1;
00091     }
00092
00093     *ucs = (unsigned int) '?'; /* bad UTF-8 string */
00094     return -1;
00095 }
00096
00097 /*
00098  * Converts an Unicode value to an UTF-8 string

```

```

00099  * NOTE : the buffer (buf) must be at least 5 bytes long !!!
00100  */
00101  int
00102  XConvertUcsToUtf8(unsigned int  ucs,
00103                  char          *buf) {
00104
00105      if (ucs < 0x000080) {
00106          buf[0] = ucs;
00107          return 1;
00108      } else if (ucs < 0x000800) {
00109          buf[0] = 0xC0 | (ucs » 6);
00110          buf[1] = 0x80 | (ucs & 0x3F);
00111          return 2;
00112      } else if (ucs < 0x010000) {
00113          buf[0] = 0xE0 | (ucs » 12);
00114          buf[1] = 0x80 | ((ucs » 6) & 0x3F);
00115          buf[2] = 0x80 | (ucs & 0x3F);
00116          return 3;
00117      } else if (ucs < 0x00200000) {
00118          buf[0] = 0xF0 | (ucs » 18);
00119          buf[1] = 0x80 | ((ucs » 12) & 0x3F);
00120          buf[2] = 0x80 | ((ucs » 6) & 0x3F);
00121          buf[3] = 0x80 | (ucs & 0x3F);
00122          return 4;
00123      } else if (ucs < 0x01000000) {
00124          buf[0] = 0xF8 | (ucs » 24);
00125          buf[1] = 0x80 | ((ucs » 18) & 0x3F);
00126          buf[2] = 0x80 | ((ucs » 12) & 0x3F);
00127          buf[3] = 0x80 | ((ucs » 6) & 0x3F);
00128          buf[4] = 0x80 | (ucs & 0x3F);
00129          return 5;
00130      }
00131      buf[0] = '?';
00132      return -1;
00133  }
00134
00135  /*
00136  * returns the byte length of the first UTF-8 char
00137  * (returns -1 if not valid)
00138  */
00139  int
00140  XUtf8CharByteLen(const unsigned char  *buf,
00141                  int                  len) {
00142      unsigned int ucs;
00143      return XConvertUtf8ToUcs(buf, len, &ucs);
00144  }
00145
00146  /*
00147  * returns the quantity of Unicode chars in the UTF-8 string
00148  */
00149  int
00150  XCountUtf8Char(const unsigned char  *buf,
00151                  int                  len) {
00152
00153      int i = 0;
00154      int nbc = 0;
00155      while (i < len) {
00156          int cl = XUtf8CharByteLen(buf + i, len - i);
00157          if (cl < 1) cl = 1;
00158          nbc++;
00159          i += cl;
00160      }
00161      return nbc;
00162  }
00163
00164  /*
00165  * Same as XConvertUtf8ToUcs but no sanity check is done.
00166  */
00167  int
00168  XFastConvertUtf8ToUcs(const unsigned char  *buf,
00169                       int                  len,
00170                       unsigned int         *ucs) {
00171
00172      if (buf[0] & 0x80) {
00173          if (buf[0] & 0x40) {
00174              if (buf[0] & 0x20) {
00175                  if (buf[0] & 0x10) {
00176                      if (buf[0] & 0x08) {
00177                          if (buf[0] & 0x04) {
00178                              if (buf[0] & 0x02) {
00179                                  /* bad UTF-8 string */
00180                              } else {
00181                                  /* 0x04000000 - 0x7FFFFFFF */
00182                              }
00183                          } else if (len > 4) {
00184                              /* 0x00200000 - 0x03FFFFFF */
00185                              *ucs = ((buf[0] & ~0xF8) « 24) +

```

```

00186             ((buf[1] & ~0x80) << 18) +
00187             ((buf[2] & ~0x80) << 12) +
00188             ((buf[3] & ~0x80) << 6) +
00189             (buf[4] & ~0x80);
00190         return 5;
00191     }
00192     } else if (len > 3) {
00193         /* 0x00010000 - 0x001FFFFF */
00194         *ucs = ((buf[0] & ~0xF0) << 18) +
00195             ((buf[1] & ~0x80) << 12) +
00196             ((buf[2] & ~0x80) << 6) +
00197             (buf[3] & ~0x80);
00198         return 4;
00199     }
00200     } else if (len > 2) {
00201         /* 0x00000800 - 0x0000FFFF */
00202         *ucs = ((buf[0] & ~0xE0) << 12) +
00203             ((buf[1] & ~0x80) << 6) +
00204             (buf[2] & ~0x80);
00205         return 3;
00206     }
00207     } else if (len > 1) {
00208         /* 0x00000080 - 0x000007FF */
00209         *ucs = ((buf[0] & ~0xC0) << 6) +
00210             (buf[1] & ~0x80);
00211         return 2;
00212     }
00213     }
00214     } else if (len > 0) {
00215         /* 0x00000000 - 0x0000007F */
00216         *ucs = buf[0];
00217         return 1;
00218     }
00219     }
00220     *ucs = (unsigned int) '?'; /* bad UTF-8 string */
00221     return -1;
00222 }

```

12.302 Ximint.h

```
00001
```

12.303 Xlibint.h

```
00001
```


Index

[_FL_DIAMOND_DOWN_BOX](#)
Enumerations.H, [2005](#)

[_FL_DIAMOND_UP_BOX](#)
Enumerations.H, [2005](#)

[_FL_EMBOSSED_LABEL](#)
Enumerations.H, [2011](#)

[_FL_ENGRAVED_LABEL](#)
Enumerations.H, [2011](#)

[_FL_GLEAM_DOWN_BOX](#)
Enumerations.H, [2005](#)

[_FL_GLEAM_DOWN_FRAME](#)
Enumerations.H, [2005](#)

[_FL_GLEAM_ROUND_DOWN_BOX](#)
Enumerations.H, [2006](#)

[_FL_GLEAM_ROUND_UP_BOX](#)
Enumerations.H, [2006](#)

[_FL_GLEAM_THIN_DOWN_BOX](#)
Enumerations.H, [2006](#)

[_FL_GLEAM_THIN_UP_BOX](#)
Enumerations.H, [2005](#)

[_FL_GLEAM_UP_BOX](#)
Enumerations.H, [2005](#)

[_FL_GLEAM_UP_FRAME](#)
Enumerations.H, [2005](#)

[_FL_GTK_DOWN_BOX](#)
Enumerations.H, [2005](#)

[_FL_GTK_DOWN_FRAME](#)
Enumerations.H, [2005](#)

[_FL_GTK_ROUND_DOWN_BOX](#)
Enumerations.H, [2005](#)

[_FL_GTK_ROUND_UP_BOX](#)
Enumerations.H, [2005](#)

[_FL_GTK_THIN_DOWN_BOX](#)
Enumerations.H, [2005](#)

[_FL_GTK_THIN_DOWN_FRAME](#)
Enumerations.H, [2005](#)

[_FL_GTK_THIN_UP_BOX](#)
Enumerations.H, [2005](#)

[_FL_GTK_THIN_UP_FRAME](#)
Enumerations.H, [2005](#)

[_FL_GTK_UP_BOX](#)
Enumerations.H, [2005](#)

[_FL_GTK_UP_FRAME](#)
Enumerations.H, [2005](#)

[_FL_ICON_LABEL](#)
Enumerations.H, [2011](#)

[_FL_IMAGE_LABEL](#)
Enumerations.H, [2011](#)

[_FL_MULTI_LABEL](#)
Enumerations.H, [2011](#)

[_FL_OFLAT_BOX](#)
Enumerations.H, [2005](#)

[_FL_OSHADOW_BOX](#)
Enumerations.H, [2005](#)

[_FL_OVAL_BOX](#)
Enumerations.H, [2005](#)

[_FL_OVAL_FRAME](#)
Enumerations.H, [2005](#)

[_FL_OXY_BUTTON_DOWN_BOX](#)
Enumerations.H, [2006](#)

[_FL_OXY_BUTTON_UP_BOX](#)
Enumerations.H, [2006](#)

[_FL_OXY_DOWN_BOX](#)
Enumerations.H, [2006](#)

[_FL_OXY_DOWN_FRAME](#)
Enumerations.H, [2006](#)

[_FL_OXY_ROUND_DOWN_BOX](#)
Enumerations.H, [2006](#)

[_FL_OXY_ROUND_UP_BOX](#)
Enumerations.H, [2006](#)

[_FL_OXY_THIN_DOWN_BOX](#)
Enumerations.H, [2006](#)

[_FL_OXY_THIN_DOWN_FRAME](#)
Enumerations.H, [2006](#)

[_FL_OXY_THIN_UP_BOX](#)
Enumerations.H, [2006](#)

[_FL_OXY_THIN_UP_FRAME](#)
Enumerations.H, [2006](#)

[_FL_OXY_UP_BOX](#)
Enumerations.H, [2006](#)

[_FL_OXY_UP_FRAME](#)
Enumerations.H, [2006](#)

[_FL_PLASTIC_DOWN_BOX](#)
Enumerations.H, [2005](#)

[_FL_PLASTIC_DOWN_FRAME](#)
Enumerations.H, [2005](#)

[_FL_PLASTIC_ROUND_DOWN_BOX](#)
Enumerations.H, [2005](#)

[_FL_PLASTIC_ROUND_UP_BOX](#)
Enumerations.H, [2005](#)

[_FL_PLASTIC_THIN_DOWN_BOX](#)
Enumerations.H, [2005](#)

[_FL_PLASTIC_THIN_UP_BOX](#)
Enumerations.H, [2005](#)

[_FL_PLASTIC_UP_BOX](#)
Enumerations.H, [2005](#)

[_FL_PLASTIC_UP_FRAME](#)
Enumerations.H, [2005](#)

- [_FL_RFLAT_BOX](#)
 - Enumerations.H, [2005](#)
- [_FL_ROUNDED_BOX](#)
 - Enumerations.H, [2005](#)
- [_FL_ROUNDED_FRAME](#)
 - Enumerations.H, [2005](#)
- [_FL_ROUND_DOWN_BOX](#)
 - Enumerations.H, [2005](#)
- [_FL_ROUND_UP_BOX](#)
 - Enumerations.H, [2005](#)
- [_FL_RSHADOW_BOX](#)
 - Enumerations.H, [2005](#)
- [_FL_SHADOW_BOX](#)
 - Enumerations.H, [2004](#)
- [_FL_SHADOW_FRAME](#)
 - Enumerations.H, [2005](#)
- [_FL_SHADOW_LABEL](#)
 - Enumerations.H, [2011](#)
- [_RESERVED_1](#)
 - FI_Terminal, [1598](#)
- [_RESERVED_2](#)
 - FI_Terminal, [1598](#)
- [__fl_attr](#)
 - fl_attr.h, [2040](#)
- [_remove](#)
 - FI_Browser, [455](#)
- [~Cell](#)
 - FI_Grid::Cell, [363](#)
- [~FI_Anim_GIF_Image](#)
 - FI_Anim_GIF_Image, [415](#)
- [~FI_Double_Window](#)
 - FI_Double_Window, [643](#)
- [~FI_EPS_File_Surface](#)
 - FI_EPS_File_Surface, [647](#)
- [~FI_Group](#)
 - FI_Group, [837](#)
- [~FI_Help_View](#)
 - FI_Help_View, [863](#)
- [~FI_Input_](#)
 - FI_Input_, [958](#)
- [~FI_Native_File_Chooser](#)
 - FI_Native_File_Chooser, [1135](#)
- [~FI_Plugin_Manager](#)
 - FI_Plugin_Manager, [1205](#)
- [~FI_Preferences](#)
 - FI_Preferences, [1238](#)
- [~FI_SVG_File_Surface](#)
 - FI_SVG_File_Surface, [1488](#)
- [~FI_Scroll](#)
 - FI_Scroll, [1368](#)
- [~FI_Shared_Image](#)
 - FI_Shared_Image, [1413](#)
- [~FI_Table](#)
 - FI_Table, [1528](#)
- [~FI_Table_Row](#)
 - FI_Table_Row, [1552](#)
- [~FI_Terminal](#)
 - FI_Terminal, [1600](#)
- [~FI_Text_Display](#)
 - FI_Text_Display, [1655](#)
- [~FI_Widget](#)
 - FI_Widget, [1892](#)
- [~FI_Window](#)
 - FI_Window, [1945](#)
- [A0](#)
 - FI_Paged_Device, [1184](#)
- [A1](#)
 - FI_Paged_Device, [1184](#)
- [A2](#)
 - FI_Paged_Device, [1184](#)
- [A3](#)
 - FI_Paged_Device, [1184](#)
- [A4](#)
 - FI_Paged_Device, [1185](#)
- [A5](#)
 - FI_Paged_Device, [1185](#)
- [A6](#)
 - FI_Paged_Device, [1185](#)
- [A7](#)
 - FI_Paged_Device, [1185](#)
- [A8](#)
 - FI_Paged_Device, [1185](#)
- [A9](#)
 - FI_Paged_Device, [1185](#)
- [abi_check](#)
 - FI, [377](#)
- [abi_version](#)
 - FI, [377](#)
- [about](#)
 - FI_Sys_Menu_Bar, [1508](#)
- [absolute_top_line_number](#)
 - FI_Text_Display, [1655](#)
- [activate](#)
 - FI_Tree_Item, [1816](#)
 - FI_Widget, [1892](#)
- [active](#)
 - FI_Widget, [1892](#)
- [active_r](#)
 - FI_Widget, [1892](#)
- [add](#)
 - FI_Browser, [455](#)
 - FI_Chart, [530](#)
 - FI_Check_Browser, [548](#)
 - FI_File_Icon, [673](#)
 - FI_Input_Choice, [986](#)
 - FI_Menu_, [1032](#)
 - FI_Menu_Item, [1073](#)
 - FI_Shared_Image, [1413](#)
 - FI_Sys_Menu_Bar, [1508](#), [1509](#)
 - FI_Tree, [1780](#), [1781](#)
 - FI_Tree_Item, [1816](#), [1817](#)
 - FI_Tree_Item_Array, [1833](#)
- [add_check](#)
 - FI, [377](#)
- [add_clipboard_notify](#)
 - Selection & Clipboard functions, [251](#)

- add_color
 - FI_File_Icon, [673](#)
- add_extra
 - FI_File_Chooser, [669](#)
- add_fd
 - FI, [377](#), [378](#)
- add_handler
 - Events handling functions, [237](#)
 - FI_Shared_Image, [1413](#)
- add_idle
 - FI, [378](#)
- add_key_binding
 - FI_Text_Editor, [1703](#)
- add_modify_callback
 - FI_Text_Buffer, [1625](#)
- add_scheme_name
 - FI_Scheme, [1346](#)
- add_system_handler
 - Events handling functions, [237](#)
- add_timeout
 - FI, [378](#)
 - FI_Timeout, [1735](#)
- add_vertex
 - FI_File_Icon, [673](#), [674](#)
- Adding and Extending Widgets, [82](#)
- addPlugin
 - FI_Plugin_Manager, [1205](#)
- address
 - FI_Text_Buffer, [1626](#)
- Advanced FLTK, [99](#)
- align
 - FI_Widget, [1893](#)
- alloc_size
 - FI_Flex, [718](#)
- allow_expand_outside_parent
 - FI_Window, [1945](#)
- ALWAYS_ON
 - FI_Browser_, [483](#)
- angle1
 - FI_Dial, [629](#)
- animate
 - FI_GIF_Image, [768](#)
- ansi
 - FI_Terminal, [1600](#)
- api_version
 - FI, [379](#)
- append
 - FI_Input_, [958](#)
 - FI_Terminal, [1601](#)
 - FI_Text_Buffer, [1626](#)
- append_ascii
 - FI_Terminal, [1601](#)
- append_utf8
 - FI_Terminal, [1601](#)
- appendfile
 - FI_Text_Buffer, [1626](#)
- apply_undo
 - FI_Input_, [958](#)
- arg
 - FI, [379](#)
- args
 - FI, [380](#)
- args_to_utf8
 - FI, [381](#)
- argument
 - FI_Menu_Item, [1074](#)
 - FI_Widget, [1893](#)
- armSCII_8.h, [2387](#)
- array
 - FI_Group, [838](#)
 - FI_RGB_Image, [1320](#)
 - FI_Table, [1528](#)
- arrow_widths
 - FI_Counter, [617](#)
- as_double_window
 - FI_Double_Window, [643](#)
 - FI_Window, [1945](#)
- as_gl_window
 - FI_GL_Window, [783](#)
 - FI_Widget, [1894](#)
- as_group
 - FI_Group, [838](#)
 - FI_Widget, [1894](#)
- as_overlay_window
 - FI_Overlay_Window, [1170](#)
 - FI_Window, [1945](#)
- as_shared_image
 - FI_Image, [921](#)
 - FI_Shared_Image, [1413](#)
- as_svg_image
 - FI_RGB_Image, [1317](#)
 - FI_SVG_Image, [1495](#)
- as_window
 - FI_Widget, [1894](#)
 - FI_Window, [1945](#)
- ascii.h, [2388](#)
- atclose
 - Windows handling functions, [234](#)
- ATTR_BGCOLOR
 - FI_Text_Display, [1654](#)
- ATTR_BGCOLOR_EXT
 - FI_Text_Display, [1654](#)
- ATTR_BGCOLOR_EXT_
 - FI_Text_Display, [1654](#)
- ATTR_GRAMMAR
 - FI_Text_Display, [1654](#)
- ATTR_LINES_MASK
 - FI_Text_Display, [1655](#)
- ATTR_SPELLING
 - FI_Text_Display, [1655](#)
- ATTR_STRIKE_THROUGH
 - FI_Text_Display, [1655](#)
- ATTR_UNDERLINE
 - FI_Text_Display, [1654](#)
- Attrib
 - FI_Terminal, [1598](#)

- AUTO_DELETE_USER_DATA
 - FI_Widget, [1891](#)
- autosize
 - FI_Chart, [530](#)
- awake
 - Multithreading support functions, [312](#)
- b
 - FI_Color_Chooser, [601](#)
 - FI_Rect, [1294](#)
- B0
 - FI_Paged_Device, [1185](#)
- B1
 - FI_Paged_Device, [1185](#)
- B10
 - FI_Paged_Device, [1185](#)
- B2
 - FI_Paged_Device, [1185](#)
- B3
 - FI_Paged_Device, [1185](#)
- B4
 - FI_Paged_Device, [1185](#)
- B5
 - FI_Paged_Device, [1185](#)
- B6
 - FI_Paged_Device, [1185](#)
- B7
 - FI_Paged_Device, [1185](#)
- B8
 - FI_Paged_Device, [1185](#)
- B9
 - FI_Paged_Device, [1185](#)
- background
 - FI, [382](#)
- background2
 - FI, [382](#)
- bbox
 - FI_Browser_, [483](#)
 - FI_Scroll, [1368](#)
- begin
 - FI_Group, [838](#)
- begin_document
 - FI_PDF_File_Surface, [1191](#)
- begin_job
 - FI_Paged_Device, [1185](#)
 - FI_PDF_File_Surface, [1191](#)
 - FI_PostScript_File_Device, [1225](#), [1226](#)
 - FI_Printer, [1257](#)
- begin_page
 - FI_Paged_Device, [1186](#)
 - FI_PDF_File_Surface, [1192](#)
 - FI_PostScript_File_Device, [1226](#)
 - FI_Printer, [1257](#)
- belowmouse
 - Events handling functions, [238](#)
- BG_XTERM
 - FI_Terminal, [1598](#)
- big5.h, [2388](#)
- big5_emacs.h, [2436](#)
- bind_deimage
 - FI_Widget, [1895](#)
- bind_image
 - FI_Widget, [1895](#), [1897](#)
- BLOCK_CURSOR
 - FI_Text_Display, [1654](#)
- BOLD
 - FI_Terminal, [1598](#)
- border
 - FI_Window, [1946](#)
- BOTH
 - FI_Browser_, [483](#)
- BOTH_ALWAYS
 - FI_Browser_, [483](#)
- bottomline
 - FI_Browser, [456](#)
- bounds
 - FI_Chart, [532](#)
 - FI_Group, [838](#)
 - FI_Slider, [1459](#)
- box
 - FI_Terminal, [1602](#)
 - FI_Widget, [1897](#)
- box_border_radius_max
 - FI, [382](#)
- box_color
 - FI, [383](#)
- box_dh
 - FI, [383](#)
- box_dw
 - FI, [383](#)
- box_dx
 - FI, [383](#)
- box_dy
 - FI, [383](#)
- box_shadow_width
 - FI, [383](#), [384](#)
- BROWSE_DIRECTORY
 - FI_Native_File_Chooser, [1135](#)
- BROWSE_FILE
 - FI_Native_File_Chooser, [1135](#)
- BROWSE_MULTI_DIRECTORY
 - FI_Native_File_Chooser, [1135](#)
- BROWSE_MULTI_FILE
 - FI_Native_File_Chooser, [1135](#)
- BROWSE_SAVE_DIRECTORY
 - FI_Native_File_Chooser, [1135](#)
- BROWSE_SAVE_FILE
 - FI_Native_File_Chooser, [1135](#)
- buffer
 - FI_Text_Display, [1655](#), [1656](#)
- buffer_modified_cb
 - FI_Text_Display, [1656](#)
- buffer_predelete_cb
 - FI_Text_Display, [1656](#)
- byte_at
 - FI_Text_Buffer, [1626](#)
- C_LOCALE

- FI_Preferences, [1235](#)
- Cairo Support Functions and Classes, [315](#)
 - cairo_autolink_context, [316](#)
 - cairo_cc, [316](#)
 - cairo_flush, [317](#)
 - cairo_make_current, [317](#)
- cairo_autolink_context
 - Cairo Support Functions and Classes, [316](#)
- cairo_cc
 - Cairo Support Functions and Classes, [316](#)
- cairo_flush
 - Cairo Support Functions and Classes, [317](#)
- cairo_make_current
 - Cairo Support Functions and Classes, [317](#)
- calc_dimensions
 - FI_Tree, [1781](#)
- calc_item_height
 - FI_Tree_Item, [1817](#)
- calc_last_char
 - FI_Text_Display, [1657](#)
- calc_line_starts
 - FI_Text_Display, [1657](#)
- calc_tree
 - FI_Tree, [1781](#)
- callback
 - FI_Menu_Item, [1074](#)
 - FI_Table, [1528](#)
 - FI_Widget, [1897](#), [1898](#)
- Callback Function Typedefs, [231](#)
 - FI_Event_Dispatch, [232](#)
 - FI_Timeout_Handler, [232](#)
- callback_col
 - FI_Table, [1529](#)
- callback_context
 - FI_Table, [1529](#)
- callback_item
 - FI_Tree, [1782](#)
- callback_reason
 - Events handling functions, [238](#)
 - FI_Tree, [1782](#)
- callback_row
 - FI_Table, [1529](#)
- can_do
 - FI_Gl_Window, [784](#)
- can_do_overlay
 - FI_Gl_Window, [784](#)
- can_redo
 - FI_Input_, [958](#)
 - FI_Text_Buffer, [1627](#)
- can_undo
 - FI_Input_, [959](#)
 - FI_Text_Buffer, [1627](#)
- canUndo
 - FI_Text_Buffer, [1627](#)
- canvas
 - FI_Anim_GIF_Image, [415](#)
- canvas_h
 - FI_Anim_GIF_Image, [415](#)
- canvas_w
 - FI_Anim_GIF_Image, [415](#)
- CARET_CURSOR
 - FI_Text_Display, [1654](#)
- case.h, [2320](#)
- cc
 - FI_Cairo_State, [507](#)
- cell
 - FI_Grid, [818](#), [819](#)
- cgdebug.h, [2273](#)
- CHANGED
 - FI_Widget, [1891](#)
- changed
 - FI_Widget, [1899](#)
- char_at
 - FI_Text_Buffer, [1627](#)
- CharFlags
 - FI_Terminal, [1598](#)
- check
 - FI, [384](#)
 - FI_Menu_Item, [1075](#)
- checkbox
 - FI_Menu_Item, [1075](#)
- checked
 - FI_Menu_Item, [1075](#)
- child
 - FI_Group, [839](#)
 - FI_Table, [1529](#)
 - FI_Tree_Item, [1818](#)
- children
 - FI_Table, [1530](#)
- CLEAR
 - FI_Preferences, [1235](#)
- clear
 - FI_Browser, [456](#)
 - FI_Button, [503](#)
 - FI_Group, [839](#)
 - FI_Menu_, [1035](#)
 - FI_Pack, [1181](#)
 - FI_Sys_Menu_Bar, [1509](#)
 - FI_Table, [1530](#)
 - FI_Table_Row, [1553](#)
 - FI_Terminal, [1602](#)
 - FI_Tree, [1783](#)
 - FI_Tree_Item_Array, [1833](#)
- clear_active
 - FI_Widget, [1899](#)
- clear_border
 - FI_Window, [1946](#)
- clear_changed
 - FI_Widget, [1899](#)
- clear_children
 - FI_Tree, [1783](#)
- clear_damage
 - FI_Widget, [1899](#)
- clear_layout
 - FI_Grid, [819](#)
- clear_modal_states

- FI_Window, 1946
- clear_output
 - FI_Widget, 1900
- clear_rect
 - FI_Text_Display, 1657
- clear_screen
 - FI_Terminal, 1602
- clear_screen_home
 - FI_Terminal, 1602
- clear_submenu
 - FI_Menu_, 1035
 - FI_Sys_Menu_Bar, 1509
- clear_tab_positions
 - FI_Tabs, 1568
- clear_visible
 - FI_Widget, 1900
- clear_visible_focus
 - FI_Widget, 1900
- clear_widget_pointer
 - Safe widget deletion support functions, 314
- client_area
 - FI_Tabs, 1568
- CLIP_CHILDREN
 - FI_Widget, 1891
- clip_children
 - FI_Group, 839, 840
- clipboard_contains
 - Selection & Clipboard functions, 251
- close
 - FI_EPS_File_Surface, 648
 - FI_SVG_File_Surface, 1488
 - FI_Tree, 1783
- closedeicon
 - FI_Tree_Prefs, 1838
- closeicon
 - FI_Tree, 1784
 - FI_Tree_Prefs, 1838
- col_gap
 - FI_Grid, 819
- col_header
 - FI_Table, 1530
- col_resize
 - FI_Table, 1530
- col_resize_min
 - FI_Table, 1530
- col_to_x
 - FI_Text_Display, 1657
- col_weight
 - FI_Grid, 820
- col_width
 - FI_Grid, 821
 - FI_Table, 1530
- col_width_all
 - FI_Table, 1530
- color
 - FI_Terminal, 1603
 - FI_Tooltip, 1758
 - FI_Widget, 1900, 1901
- Color & Font functions, 261
 - fl_color, 263
 - fl_color_average, 264
 - fl_contrast, 264
 - fl_contrast_function, 265
 - fl_contrast_level, 266
 - fl_contrast_mode, 267
 - fl_font, 268
 - fl_height, 268
 - fl_latin1_to_local, 269
 - fl_lightness, 269
 - fl_local_to_latin1, 269
 - fl_local_to_mac_roman, 270
 - fl_luminance, 270
 - fl_mac_roman_to_local, 270
 - fl_show_colormap, 271
 - fl_size, 272
 - fl_text_extents, 272
 - fl_width, 272
 - free_color, 272
 - get_color, 273
 - get_font, 273
 - get_font_name, 273
 - get_font_sizes, 274
 - set_color, 274
 - set_font, 274
 - set_fonts, 275
- color2
 - FI_Widget, 1901
- color_average
 - FI_Anim_GIF_Image, 416
 - FI_Image, 921
 - FI_Pixmap, 1198
 - FI_RGB_Image, 1317
 - FI_Shared_Image, 1413
 - FI_SVG_Image, 1495
 - FI_Tiled_Image, 1732
- COLUMN
 - FI_Flex, 716
- column_char
 - FI_Browser, 456
- column_widths
 - FI_Browser, 456, 457
- Common Dialog Classes and Functions, 336
 - error, 349
 - fatal, 349
 - fl_alert, 338
 - fl_ask, 338
 - fl_beep, 338
 - fl_choice, 338
 - fl_choice_n, 340
 - fl_color_chooser, 340, 341
 - fl_dir_chooser, 342
 - fl_file_chooser, 342
 - fl_file_chooser_callback, 343
 - fl_file_chooser_ok_label, 344
 - fl_input, 344
 - fl_message, 345

- [fl_message_hotspot](#), [345](#)
 - [fl_message_icon](#), [345](#)
 - [fl_message_icon_label](#), [346](#)
 - [fl_message_position](#), [346](#), [347](#)
 - [fl_message_title](#), [348](#)
 - [fl_message_title_default](#), [348](#)
 - [fl_password](#), [348](#), [349](#)
 - [warning](#), [350](#)
- Common Widgets and Attributes, [16](#)
- compact
 - [Fl_Button](#), [503](#), [504](#)
- compare
 - [Fl_Shared_Image](#), [1414](#)
- compose
 - Events handling functions, [239](#)
- compose_reset
 - Events handling functions, [239](#)
- connectorstyle
 - [Fl_Tree](#), [1784](#)
- Constants and Enumerations, [112](#)
- contains
 - [Fl_Widget](#), [1901](#)
- context
 - [Fl_Gl_Window](#), [784](#)
- CONTEXT_CELL
 - [Fl_Table](#), [1528](#)
- CONTEXT_COL_HEADER
 - [Fl_Table](#), [1528](#)
- CONTEXT_ENDPAGE
 - [Fl_Table](#), [1528](#)
- CONTEXT_NONE
 - [Fl_Table](#), [1528](#)
- CONTEXT_RC_RESIZE
 - [Fl_Table](#), [1528](#)
- CONTEXT_ROW_HEADER
 - [Fl_Table](#), [1528](#)
- CONTEXT_STARTPAGE
 - [Fl_Table](#), [1528](#)
- CONTEXT_TABLE
 - [Fl_Table](#), [1528](#)
- context_valid
 - [Fl_Gl_Window](#), [784](#)
- Coordinates and Layout Widgets, [28](#)
- COPIED_LABEL
 - [Fl_Widget](#), [1891](#)
- COPIED_TOOLTIP
 - [Fl_Widget](#), [1891](#)
- copy
 - [Fl_Anim_GIF_Image](#), [416](#)
 - [Fl_Bitmap](#), [427](#)
 - [Fl_Help_View](#), [863](#)
 - [Fl_Image](#), [922](#)
 - [Fl_Input_](#), [959](#)
 - [Fl_Menu_](#), [1036](#)
 - [Fl_Pixmap](#), [1198](#)
 - [Fl_RGB_Image](#), [1318](#)
 - [Fl_Shared_Image](#), [1414](#), [1415](#)
 - [Fl_SVG_Image](#), [1495](#)
 - [Fl_Text_Buffer](#), [1627](#)
 - [Fl_Tiled_Image](#), [1732](#)
 - Selection & Clipboard functions, [251](#)
- copy_
 - [Fl_Shared_Image](#), [1415](#)
- copy_cuts
 - [Fl_Input_](#), [959](#)
- copy_label
 - [Fl_Widget](#), [1902](#)
- copy_tooltip
 - [Fl_Widget](#), [1902](#)
- CORE
 - [Fl_Preferences](#), [1235](#)
- CORE_READ_OK
 - [Fl_Preferences](#), [1252](#)
- CORE_SYSTEM
 - [Fl_Preferences](#), [1235](#)
- CORE_SYSTEM_L
 - [Fl_Preferences](#), [1235](#)
- CORE_USER
 - [Fl_Preferences](#), [1235](#)
- CORE_USER_L
 - [Fl_Preferences](#), [1235](#)
- CORE_WRITE_OK
 - [Fl_Preferences](#), [1252](#)
- count
 - [Fl_Image](#), [923](#)
 - [Fl_Native_File_Chooser](#), [1135](#)
- count_displayed_characters
 - [Fl_Text_Buffer](#), [1628](#)
- count_lines
 - [Fl_Text_Buffer](#), [1628](#)
 - [Fl_Text_Display](#), [1658](#)
- cp1133.h, [2438](#)
- cp1251.h, [2439](#)
- cp1255.h, [2440](#)
- cp1256.h, [2442](#)
- cp936ext.h, [2443](#)
- CR_TO_LF
 - [Fl_Terminal](#), [1599](#)
- CREATE
 - [Fl_File_Chooser](#), [669](#)
- create_window_menu
 - [Fl_Sys_Menu_Bar](#), [1510](#)
- current
 - [Fl_Group](#), [840](#)
 - [Fl_Timeout](#), [1736](#)
 - [Fl_Tooltip](#), [1759](#)
 - [Fl_Window](#), [1946](#)
- current_
 - [Fl_Window](#), [1963](#)
- current_timeout
 - [Fl_Timeout](#), [1740](#)
- cursor
 - [Fl_Tile](#), [1723](#)
 - [Fl_Window](#), [1947](#)
- cursor2rowcol
 - [Fl_Table](#), [1531](#)

- cursor_col
 - FI_Terminal, [1603](#)
- cursor_color
 - FI_Input_, [959](#)
 - FI_Text_Display, [1658](#)
 - FI_Value_Input, [1859](#), [1860](#)
- cursor_cr
 - FI_Terminal, [1603](#)
- cursor_down
 - FI_Terminal, [1603](#)
- cursor_right
 - FI_Terminal, [1603](#)
- cursor_row
 - FI_Terminal, [1604](#)
- cursor_style
 - FI_Text_Display, [1658](#)
- cursor_up
 - FI_Terminal, [1604](#)
- custom_application_menu_items
 - FI_Mac_App_Menu, [1022](#)
- cut
 - FI_Input_, [960](#)
- d
 - FI_Image, [923](#)
- damage
 - FI_Widget, [1902](#), [1904](#)
- damage_zone
 - FI_Table, [1531](#)
- data
 - FI_Browser, [457](#)
 - FI_Image, [923](#)
- deactivate
 - FI_Menu_Item, [1075](#)
 - FI_Tree_Item, [1818](#)
 - FI_Widget, [1904](#)
- debug
 - FI_Grid, [821](#)
- DEBUG_FLAG
 - FI_Anim_GIF_Image, [414](#)
- decorated_h
 - FI_Window, [1947](#)
- decorated_w
 - FI_Window, [1948](#)
- default_atclose
 - Windows handling functions, [233](#)
- default_callback
 - FI_Widget, [1904](#)
- default_cursor
 - FI_Window, [1948](#)
- default_icon
 - FI_Window, [1948](#)
- default_icons
 - FI_Window, [1948](#), [1949](#)
- default_size_range
 - FI_Window, [1949](#)
- default_xclass
 - FI_Window, [1950](#)
- deimage
 - FI_Widget, [1905](#)
- DEIMAGE_BOUND
 - FI_Widget, [1891](#)
- deimage_bound
 - FI_Widget, [1905](#)
- delay
 - FI_Anim_GIF_Image, [416](#)
 - FI_Tooltip, [1759](#)
- delete_child
 - FI_Group, [840](#)
 - FI_Scroll, [1368](#)
- delete_entry
 - FI_Preferences, [1239](#)
- delete_group
 - FI_Preferences, [1239](#)
- delete_rows
 - FI_Terminal, [1604](#)
- delete_widget
 - Safe widget deletion support functions, [314](#)
- deleted
 - FI_Widget_Tracker, [1931](#)
- deleting
 - FI_Browser_, [483](#)
- deparent
 - FI_Tree_Item, [1818](#)
 - FI_Tree_Item_Array, [1833](#)
- Deprecated List, [207](#)
- depth
 - FI_Tree_Item, [1818](#)
- desaturate
 - FI_Anim_GIF_Image, [417](#)
 - FI_Image, [924](#)
 - FI_Pixmap, [1199](#)
 - FI_RGB_Image, [1318](#)
 - FI_Shared_Image, [1415](#)
 - FI_SVG_Image, [1496](#)
 - FI_Tiled_Image, [1733](#)
- deselect
 - FI_Browser_, [483](#)
 - FI_Tree, [1784](#), [1785](#)
- deselect_all
 - FI_Tree, [1785](#)
 - FI_Tree_Item, [1818](#)
- Designing a Simple Text Editor, [38](#)
- Developer info for bundled libs, [194](#)
- Developer Information, [198](#)
- Development of the FLTK library, [175](#)
- DIM
 - FI_Terminal, [1598](#)
- DIM_CURSOR
 - FI_Text_Display, [1654](#)
- dingbats_.h, [2341](#)
- direction
 - FI_Timer, [1747](#), [1748](#)
- DIRECTORY
 - FI_File_Chooser, [669](#)
- directory
 - FI_Native_File_Chooser, [1135](#)

- dirty
 - FI_Preferences, 1239
- disable
 - FI_Tooltip, 1759
- disable_im
 - Events handling functions, 239
- display
 - FI, 384
 - FI_Browser, 457
 - FI_Browser_, 484
 - FI_Tree, 1786
- display_columns
 - FI_Terminal, 1604
- display_insert
 - FI_Text_Display, 1659
- display_needs_recalc
 - FI_Text_Display, 1659
- display_rows
 - FI_Terminal, 1604
- displayed
 - FI_Browser, 458
 - FI_Browser_, 484
 - FI_Tree, 1786
- dnd
 - Selection & Clipboard functions, 252
- dnd_text_ops
 - FI, 384
- do_callback
 - FI_Menu_Item, 1075
 - FI_Table, 1531
 - FI_Widget, 1906
- do_widget_deletion
 - Safe widget deletion support functions, 314
- DONT_RESIZE_CANVAS
 - FI_Anim_GIF_Image, 414
- DONT_SET_AS_IMAGE
 - FI_Anim_GIF_Image, 414
- DONT_START
 - FI_Anim_GIF_Image, 413
- down_box
 - FI_Button, 504, 505
 - FI_File_Input, 687
 - FI_Menu_, 1036
- drag_intersection
 - FI_Tile, 1723
- draw
 - FI_Adjuster, 407
 - FI_Anim_GIF_Image, 417
 - FI_Bitmap, 427
 - FI_Box, 440
 - FI_Browser_, 484
 - FI_Button, 505
 - FI_Cairo_Window, 521
 - FI_Chart, 532
 - FI_Choice, 570
 - FI_Clock_Output, 589
 - FI_Counter, 617
 - FI_Dial, 629
 - FI_File_Icon, 674
 - FI_File_Input, 687
 - FI_Flex, 718
 - FI_FormsBitmap, 740
 - FI_FormsPixmap, 747
 - FI_FormsText, 755
 - FI_Free, 763
 - FI_Gl_Window, 784
 - FI_Glut_Window, 805
 - FI_Grid, 822
 - FI_Group, 841
 - FI_Help_View, 864
 - FI_Image, 924
 - FI_Input, 947
 - FI_Input_Choice, 986
 - FI_Label, 1003
 - FI_Light_Button, 1012
 - FI_Menu_Bar, 1056
 - FI_Menu_Button, 1067
 - FI_Pack, 1181
 - FI_Pixmap, 1199
 - FI_Positioner, 1220
 - FI_Progress, 1267
 - FI_Return_Button, 1312
 - FI_RGB_Image, 1318
 - FI_Roller, 1328
 - FI_Scroll, 1369
 - FI_Scrollbar, 1382
 - FI_Shared_Image, 1416
 - FI_Shortcut_Button, 1427
 - FI_Slider, 1459
 - FI_Spinner, 1470
 - FI_SVG_Image, 1496
 - FI_Sys_Menu_Bar, 1510
 - FI_Table, 1531
 - FI_Tabs, 1569
 - FI_Terminal, 1604
 - FI_Text_Display, 1659
 - FI_Tiled_Image, 1733
 - FI_Timer, 1748
 - FI_Tree, 1786
 - FI_Tree_Item, 1818
 - FI_Value_Input, 1860
 - FI_Value_Output, 1871
 - FI_Value_Slider, 1882
 - FI_Widget, 1907
 - FI_Widget_Surface, 1928
 - FI_Window, 1951
 - FI_Wizard, 1972
- draw_barchart
 - FI_Chart, 532
- draw_begin
 - FI_Gl_Window, 785
- draw_box_active
 - FI, 384
- draw_buff
 - FI_Terminal, 1605
- draw_cell

- FI_Table, 1531
- draw_child
 - FI_Group, 841
- draw_children
 - FI_Group, 841
- draw_cursor
 - FI_Text_Display, 1659
- draw_decorated_window
 - FI_Widget_Surface, 1929
- draw_empty
 - FI_Image, 924
- draw_end
 - FI_GL_Window, 785
- draw_focus
 - FI_Widget, 1907, 1908
- draw_GL_text_with_textures
 - FI, 385
- draw_grid
 - FI_Grid, 822
- draw_horbarchart
 - FI_Chart, 533
- draw_horizontal_connector
 - FI_Tree_Item, 1819
- draw_item_content
 - FI_Tree_Item, 1819
- draw_label
 - FI_Widget, 1908
- draw_line_numbers
 - FI_Text_Display, 1659
- draw_linechart
 - FI_Chart, 533
- draw_overlay
 - FI_Glut_Window, 806
 - FI_Overlay_Window, 1170
- draw_piechart
 - FI_Chart, 534
- draw_range
 - FI_Text_Display, 1661
- draw_row
 - FI_Terminal, 1605
- draw_row_bg
 - FI_Terminal, 1605
- draw_scaled
 - FI_Image, 924
- draw_string
 - FI_Text_Display, 1661
- draw_tab
 - FI_Tabs, 1569
- draw_text
 - FI_Text_Display, 1661
- draw_vertical_connector
 - FI_Tree_Item, 1820
- draw_vline
 - FI_Text_Display, 1662
- drawbgcolor
 - FI_Tree_Item, 1820
- drawfgcolor
 - FI_Tree_Item, 1821

- Drawing functions, 275
 - fl_add_symbol, 281
 - fl_antialias, 281
 - fl_arc, 281, 283
 - fl_begin_complex_polygon, 284
 - fl_begin_offscreen, 284
 - fl_begin_points, 284
 - fl_can_do_alpha_blending, 284
 - FL_CAP_FLAT, 281
 - FL_CAP_ROUND, 281
 - FL_CAP_SQUARE, 281
 - fl_capture_window, 284
 - fl_circle, 285
 - fl_clip, 285
 - fl_clip_box, 285
 - fl_clip_region, 286
 - fl_copy_offscreen, 287
 - fl_create_offscreen, 287
 - fl_cursor, 287
 - fl_curve, 288
 - FL_DASH, 281
 - FL_DASHDOT, 281
 - FL_DASHDOTDOT, 281
 - fl_delete_offscreen, 288
 - FL_DOT, 281
 - fl_draw, 288–290
 - fl_draw_arrow, 290
 - fl_draw_box, 290
 - fl_draw_box_focus, 291
 - fl_draw_check, 291
 - fl_draw_circle, 291
 - fl_draw_image, 292
 - fl_draw_image_mono, 293
 - fl_draw_pixmap, 294
 - fl_draw_radio, 294
 - fl_draw_symbol, 295
 - fl_expand_text, 295
 - fl_focus_rect, 296
 - fl_frame, 296
 - fl_frame2, 296
 - fl_gap, 297
 - FL_JOIN_BEVEL, 281
 - FL_JOIN_MITER, 281
 - FL_JOIN_ROUND, 281
 - fl_line_style, 297
 - fl_load_matrix, 297
 - fl_measure, 298
 - fl_measure_pixmap, 298, 299
 - fl_mult_matrix, 299
 - fl_not_clipped, 299
 - fl_old_shortcut, 299
 - fl_overlay_clear, 300
 - fl_overlay_rect, 300
 - fl_override_scale, 301
 - fl_pie, 301
 - fl_polygon, 302
 - fl_pop_clip, 302
 - fl_push_clip, 302

- fl_push_matrix, 303
- fl_read_image, 303
- fl_rect, 303, 304
- fl_rectf, 304, 305
- fl_rescale_offscreen, 305
- fl_reset_spot, 305
- fl_restore_scale, 305
- fl_rotate, 305
- fl_rounded_rect, 306
- fl_rounded_rectf, 306
- fl_scale, 306
- fl_scroll, 306
- fl_set_spot, 307
- fl_set_status, 307
- fl_shortcut_label, 307, 308
- FL_SOLID, 281
- fl_transform_dx, 309
- fl_transform_dy, 309
- fl_transform_x, 309
- fl_transform_y, 309
- fl_transformed_vertex, 309
- fl_translate, 311
- fl_vertex, 311
- Drawing Things in FLTK, 52
- drawtext
 - Fl_Input_, 960, 961
- dvalue
 - Fl_Input_, 961
- elapsed_timeouts
 - Fl_Timeout, 1736
- empty_vlines
 - Fl_Text_Display, 1662
- enable
 - Fl_Tooltip, 1759
- enable_im
 - Events handling functions, 239
- enabled
 - Fl_Tooltip, 1759
- end
 - Fl_Flex, 718
 - Fl_Group, 841
 - Fl_Text_Selection, 1710
- end_current
 - Fl_PostScript_File_Device, 1226
 - Fl_Surface_Device, 1484
- end_job
 - Fl_Paged_Device, 1186
 - Fl_PDF_File_Surface, 1192
 - Fl_PostScript_File_Device, 1226
 - Fl_Printer, 1257
- end_page
 - Fl_Paged_Device, 1186
 - Fl_PDF_File_Surface, 1192
 - Fl_PostScript_File_Device, 1227
 - Fl_Printer, 1257
- enter_area
 - Fl_Tooltip, 1759
- entries
 - Fl_Preferences, 1239
- entry
 - Fl_Preferences, 1239
- entry_exists
 - Fl_Preferences, 1241
- Enumerations.H, 1989, 2014
 - _FL_DIAMOND_DOWN_BOX, 2005
 - _FL_DIAMOND_UP_BOX, 2005
 - _FL_EMBOSSED_LABEL, 2011
 - _FL_ENGRAVED_LABEL, 2011
 - _FL_GLEAM_DOWN_BOX, 2005
 - _FL_GLEAM_DOWN_FRAME, 2005
 - _FL_GLEAM_ROUND_DOWN_BOX, 2006
 - _FL_GLEAM_ROUND_UP_BOX, 2006
 - _FL_GLEAM_THIN_DOWN_BOX, 2006
 - _FL_GLEAM_THIN_UP_BOX, 2005
 - _FL_GLEAM_UP_BOX, 2005
 - _FL_GLEAM_UP_FRAME, 2005
 - _FL_GTK_DOWN_BOX, 2005
 - _FL_GTK_DOWN_FRAME, 2005
 - _FL_GTK_ROUND_DOWN_BOX, 2005
 - _FL_GTK_ROUND_UP_BOX, 2005
 - _FL_GTK_THIN_DOWN_BOX, 2005
 - _FL_GTK_THIN_DOWN_FRAME, 2005
 - _FL_GTK_THIN_UP_BOX, 2005
 - _FL_GTK_THIN_UP_FRAME, 2005
 - _FL_GTK_UP_BOX, 2005
 - _FL_GTK_UP_FRAME, 2005
 - _FL_ICON_LABEL, 2011
 - _FL_IMAGE_LABEL, 2011
 - _FL_MULTI_LABEL, 2011
 - _FL_OFLAT_BOX, 2005
 - _FL_OSHADOW_BOX, 2005
 - _FL_OVAL_BOX, 2005
 - _FL_OVAL_FRAME, 2005
 - _FL_OXY_BUTTON_DOWN_BOX, 2006
 - _FL_OXY_BUTTON_UP_BOX, 2006
 - _FL_OXY_DOWN_BOX, 2006
 - _FL_OXY_DOWN_FRAME, 2006
 - _FL_OXY_ROUND_DOWN_BOX, 2006
 - _FL_OXY_ROUND_UP_BOX, 2006
 - _FL_OXY_THIN_DOWN_BOX, 2006
 - _FL_OXY_THIN_DOWN_FRAME, 2006
 - _FL_OXY_THIN_UP_BOX, 2006
 - _FL_OXY_THIN_UP_FRAME, 2006
 - _FL_OXY_UP_BOX, 2006
 - _FL_OXY_UP_FRAME, 2006
 - _FL_PLASTIC_DOWN_BOX, 2005
 - _FL_PLASTIC_DOWN_FRAME, 2005
 - _FL_PLASTIC_ROUND_DOWN_BOX, 2005
 - _FL_PLASTIC_ROUND_UP_BOX, 2005
 - _FL_PLASTIC_THIN_DOWN_BOX, 2005
 - _FL_PLASTIC_THIN_UP_BOX, 2005
 - _FL_PLASTIC_UP_BOX, 2005
 - _FL_PLASTIC_UP_FRAME, 2005
 - _FL_RFLAT_BOX, 2005
 - _FL_ROUNDED_BOX, 2005
 - _FL_ROUNDED_FRAME, 2005

[_FL_ROUND_DOWN_BOX, 2005](#)
[_FL_ROUND_UP_BOX, 2005](#)
[_FL_RSHADOW_BOX, 2005](#)
[_FL_SHADOW_BOX, 2004](#)
[_FL_SHADOW_FRAME, 2005](#)
[_FL_SHADOW_LABEL, 2011](#)
[FL_ABI_VERSION, 2001](#)
[FL_ACTIVATE, 2010](#)
[FL_ALIGN_LEFT, 2014](#)
[FL_ALIGN_TOP, 2014](#)
[FL_API_VERSION, 2001](#)
[FL_ARROW_CHOICE, 2004](#)
[FL_ARROW_DOUBLE, 2004](#)
[FL_ARROW_RETURN, 2004](#)
[FL_ARROW_SINGLE, 2004](#)
[Fl_Arrow_Type, 2003](#)
[FL_BORDER_BOX, 2004](#)
[FL_BORDER_FRAME, 2004](#)
[fl_box, 2013](#)
[Fl_Boxtype, 2004](#)
[FL_BUTTON, 2002](#)
[Fl_Callback_Reason, 2006](#)
[FL_CLOSE, 2009](#)
[fl_color_cube, 2013](#)
[FL_CONTRAST_CIELAB, 2007](#)
[FL_CONTRAST_CUSTOM, 2007](#)
[Fl_Contrast_Function, 2003](#)
[FL_CONTRAST_LAST, 2007](#)
[FL_CONTRAST_LEGACY, 2007](#)
[Fl_Contrast_Mode, 2007](#)
[FL_CONTRAST_NONE, 2007](#)
[Fl_Cursor, 2007](#)
[FL_CURSOR_ARROW, 2007](#)
[FL_CURSOR_CROSS, 2007](#)
[FL_CURSOR_DEFAULT, 2007](#)
[FL_CURSOR_E, 2007](#)
[FL_CURSOR_HAND, 2007](#)
[FL_CURSOR_HELP, 2007](#)
[FL_CURSOR_INSERT, 2007](#)
[FL_CURSOR_MOVE, 2007](#)
[FL_CURSOR_N, 2007](#)
[FL_CURSOR_NE, 2007](#)
[FL_CURSOR_NESW, 2007](#)
[FL_CURSOR_NONE, 2007](#)
[FL_CURSOR_NS, 2007](#)
[FL_CURSOR_NW, 2007](#)
[FL_CURSOR_NWSE, 2007](#)
[FL_CURSOR_S, 2007](#)
[FL_CURSOR_SE, 2007](#)
[FL_CURSOR_SW, 2007](#)
[FL_CURSOR_W, 2007](#)
[FL_CURSOR_WAIT, 2007](#)
[FL_CURSOR_WE, 2007](#)
[Fl_Damage, 2007](#)
[FL_DAMAGE_ALL, 2008](#)
[FL_DAMAGE_CHILD, 2008](#)
[FL_DAMAGE_EXPOSE, 2008](#)
[FL_DAMAGE_OVERLAY, 2008](#)
[FL_DAMAGE_SCROLL, 2008](#)
[FL_DAMAGE_USER1, 2008](#)
[FL_DAMAGE_USER2, 2008](#)
[FL_DEACTIVATE, 2010](#)
[fl_define_FL_EMBOSSED_LABEL, 2013](#)
[fl_define_FL_ENGRAVED_LABEL, 2013](#)
[fl_define_FL_ICON_LABEL, 2013](#)
[fl_define_FL_IMAGE_LABEL, 2013](#)
[fl_define_FL_MULTI_LABEL, 2013](#)
[fl_define_FL_SHADOW_LABEL, 2013](#)
[FL_DND_DRAG, 2010](#)
[FL_DND_ENTER, 2010](#)
[FL_DND_LEAVE, 2010](#)
[FL_DND_RELEASE, 2010](#)
[fl_down, 2013](#)
[FL_DOWN_BOX, 2004](#)
[FL_DOWN_FRAME, 2004](#)
[FL_DRAG, 2009](#)
[FL_EMBOSSED_BOX, 2004](#)
[FL_EMBOSSED_FRAME, 2004](#)
[FL_ENGRAVED_BOX, 2004](#)
[FL_ENGRAVED_FRAME, 2004](#)
[FL_ENTER, 2008](#)
[Fl_Event, 2008](#)
[FL_EXCEPT, 2003](#)
[FL_FLAT_BOX, 2004](#)
[FL_FOCUS, 2009](#)
[Fl_Fontsize, 2003](#)
[fl_frame, 2014](#)
[FL_FREE_BOXTYPE, 2006](#)
[FL_FREE_LABELTYPE, 2011](#)
[FL_FULLSCREEN, 2011](#)
[fl_gray_ramp, 2014](#)
[FL_HIDE, 2010](#)
[FL_IMAGE_LABEL, 2002](#)
[FL_KEYBOARD, 2009](#)
[FL_KEYDOWN, 2009](#)
[FL_KEYUP, 2009](#)
[Fl_Labeltype, 2011](#)
[FL_LEAVE, 2009](#)
[FL_MAJOR_VERSION, 2002](#)
[FL_MAX_BOXTYPE, 2006](#)
[FL_MINOR_VERSION, 2002](#)
[FL_MOUSEWHEEL, 2010](#)
[FL_MOVE, 2010](#)
[FL_MULTI_LABEL, 2002](#)
[FL_NO_BOX, 2004](#)
[FL_NO_EVENT, 2008](#)
[FL_NO_LABEL, 2011](#)
[FL_NORMAL_LABEL, 2011](#)
[FL_NORMAL_SIZE, 2014](#)
[FL_ORIENT_DOWN, 2012](#)
[FL_ORIENT_LEFT, 2012](#)
[FL_ORIENT_NE, 2012](#)
[FL_ORIENT_NONE, 2012](#)
[FL_ORIENT_NW, 2012](#)
[FL_ORIENT_RIGHT, 2012](#)
[FL_ORIENT_SE, 2012](#)

- FL_ORIENT_SW, [2012](#)
- FL_ORIENT_UP, [2012](#)
- FI_Orientation, [2011](#)
- FL_PASTE, [2010](#)
- FL_PATCH_VERSION, [2002](#)
- FL_PUSH, [2008](#)
- FL_READ, [2003](#)
- FL_REASON_CANCELLED, [2006](#)
- FL_REASON_CHANGED, [2006](#)
- FL_REASON_CLOSED, [2006](#)
- FL_REASON_DESELECTED, [2006](#)
- FL_REASON_DRAGGED, [2006](#)
- FL_REASON_ENTER_KEY, [2007](#)
- FL_REASON_GOT_FOCUS, [2006](#)
- FL_REASON_LOST_FOCUS, [2006](#)
- FL_REASON_OPENED, [2006](#)
- FL_REASON_RELEASED, [2006](#)
- FL_REASON_RESELECTED, [2006](#)
- FL_REASON_SELECTED, [2006](#)
- FL_REASON_UNKNOWN, [2006](#)
- FL_REASON_USER, [2007](#)
- FL_RELEASE, [2008](#)
- FL_SCREEN_CONFIGURATION_CHANGED, [2011](#)
- FL_SELECTIONCLEAR, [2010](#)
- FL_SHORTCUT, [2010](#)
- FL_SHOW, [2010](#)
- FL_SYMBOL_LABEL, [2002](#)
- FL_THIN_DOWN_BOX, [2004](#)
- FL_THIN_DOWN_FRAME, [2004](#)
- FL_THIN_UP_BOX, [2004](#)
- FL_THIN_UP_FRAME, [2004](#)
- FL_UNFOCUS, [2009](#)
- FL_UP_BOX, [2004](#)
- FL_UP_FRAME, [2004](#)
- FL_VERSION, [2002](#)
- FI_When, [2012](#)
- FL_WHEN_CHANGED, [2012](#)
- FL_WHEN_CLOSED, [2012](#)
- FL_WHEN_ENTER_KEY, [2012](#)
- FL_WHEN_ENTER_KEY_ALWAYS, [2012](#)
- FL_WHEN_ENTER_KEY_CHANGED, [2012](#)
- FL_WHEN_NEVER, [2012](#)
- FL_WHEN_NOT_CHANGED, [2012](#)
- FL_WHEN_RELEASE, [2012](#)
- FL_WHEN_RELEASE_ALWAYS, [2012](#)
- FL_WRITE, [2003](#)
- FL_ZOOM_EVENT, [2011](#)
- FL_ZOOM_GESTURE, [2011](#)
- EOL
 - FI_Terminal, [1598](#)
- errmsg
 - FI_File_Browser, [663](#)
 - FI_Native_File_Chooser, [1135](#)
- error
 - Common Dialog Classes and Functions, [349](#)
- error_char
 - FI_Terminal, [1605](#), [1606](#)
- errorcolor
 - FI_File_Input, [687](#)
- ERRORS_TO_CP1252
 - Unicode and UTF-8 functions, [320](#)
- ERRORS_TO_ISO8859_1
 - Unicode and UTF-8 functions, [320](#)
- estimate_lines
 - FI_Text_Buffer, [1628](#)
- event
 - Events handling functions, [240](#)
- event_button
 - Events handling functions, [240](#)
- event_button1
 - Events handling functions, [240](#)
- event_button2
 - Events handling functions, [240](#)
- event_button3
 - Events handling functions, [240](#)
- event_button4
 - Events handling functions, [240](#)
- event_button5
 - Events handling functions, [240](#)
- event_buttons
 - Events handling functions, [241](#)
- event_clicks
 - Events handling functions, [241](#)
- event_clipboard
 - Events handling functions, [241](#)
- event_clipboard_type
 - Events handling functions, [241](#)
- event_dispatch
 - Events handling functions, [241](#)
- event_dx
 - Events handling functions, [242](#)
- event_dy
 - Events handling functions, [242](#)
- event_inside
 - Events handling functions, [242](#), [243](#)
- event_is_click
 - Events handling functions, [243](#)
- event_key
 - Events handling functions, [243](#), [244](#)
- event_length
 - Events handling functions, [244](#)
- event_original_key
 - Events handling functions, [244](#)
- event_state
 - Events handling functions, [244](#), [245](#)
- event_text
 - Events handling functions, [245](#)
- event_x_root
 - Events handling functions, [246](#)
- event_y_root
 - Events handling functions, [246](#)
- Events handling functions, [234](#)
 - add_handler, [237](#)
 - add_system_handler, [237](#)
 - belowmouse, [238](#)

- callback_reason, 238
- compose, 239
- compose_reset, 239
- disable_im, 239
- enable_im, 239
- event, 240
- event_button, 240
- event_button1, 240
- event_button2, 240
- event_button3, 240
- event_button4, 240
- event_button5, 240
- event_buttons, 241
- event_clicks, 241
- event_clipboard, 241
- event_clipboard_type, 241
- event_dispatch, 241
- event_dx, 242
- event_dy, 242
- event_inside, 242, 243
- event_is_click, 243
- event_key, 243, 244
- event_length, 244
- event_original_key, 244
- event_state, 244, 245
- event_text, 245
- event_x_root, 246
- event_y_root, 246
- fl_callback_reason_names, 249
- fl_eventnames, 249
- fl_fontnames, 249
- focus, 246
- get_key, 246
- get_mouse, 247
- handle, 247
- handle_, 247
- last_handler, 248
- pushed, 248
- remove_handler, 248
- remove_system_handler, 248
- test_shortcut, 248
- Example Source Code, 154
- EXECUTIVE
 - FI_Paged_Device, 1185
- exists
 - FI_Widget_Tracker, 1931
- extend_range_for_styles
 - FI_Text_Display, 1662
- extend_selection
 - FI_Tree, 1786
- extend_selection_dir
 - FI_Tree, 1787
- fail
 - FI_Image, 926
- FAQ (Frequently Asked Questions), 164
- fastarrow.h, 2275
- fatal
 - Common Dialog Classes and Functions, 349
- FG_XTERM
 - FI_Terminal, 1598
- File names and URI utility functions, 350
 - fl_decode_uri, 352
 - FI_File_Sort_F, 351
 - fl_filename_absolute, 352
 - fl_filename_absolute_str, 352, 353
 - fl_filename_expand, 353
 - fl_filename_expand_str, 354
 - fl_filename_ext, 354
 - fl_filename_ext_str, 354
 - fl_filename_free_list, 354
 - fl_filename_isdir, 355
 - fl_filename_list, 355
 - fl_filename_match, 356
 - fl_filename_name, 357
 - fl_filename_name_str, 357
 - fl_filename_path_str, 357
 - fl_filename_relative, 357, 358
 - fl_filename_relative_str, 359
 - fl_filename_setext, 359
 - fl_filename_setext_str, 360
 - fl_getcwd_str, 360
 - fl_open_uri, 360
- file_access
 - FI_Preferences, 1241
- file_encoding_warning_message
 - FI_Text_Buffer, 1637
- filename
 - FI_Native_File_Chooser, 1136
 - FI_Preferences, 1241, 1243
- filename.H, 2023, 2024
- filetype
 - FI_File_Browser, 663
- filter
 - FI_File_Browser, 663
 - FI_File_Chooser, 670
 - FI_Native_File_Chooser, 1136
- filter_value
 - FI_Native_File_Chooser, 1136, 1137
- find
 - FI_File_Icon, 674
 - FI_Group, 841
 - FI_Help_View, 864
 - FI_Shared_Image, 1416
- find_cell
 - FI_Table, 1533
- find_child
 - FI_Tree_Item, 1821
- find_child_item
 - FI_Tree_Item, 1821
- find_clicked
 - FI_Tree, 1787
 - FI_Tree_Item, 1822
- find_index
 - FI_Menu_, 1036, 1037
- find_item
 - FI_Browser_, 484

- FI_Menu_, 1037, 1038
- FI_Tree, 1788
- FI_Tree_Item, 1822
- find_item_with_argument
 - FI_Menu_, 1038
- find_item_with_user_data
 - FI_Menu_, 1038
- find_line
 - FI_Browser, 458
- find_line_end
 - FI_Text_Display, 1662
- find_shortcut
 - FI_Menu_Item, 1076
- find_wrap_range
 - FI_Text_Display, 1663
- find_x
 - FI_Text_Display, 1663
- findchar_backward
 - FI_Text_Buffer, 1628
- findchar_forward
 - FI_Text_Buffer, 1628
- first
 - FI_Tree, 1788
- first_selected_item
 - FI_Tree, 1789
- first_timeout
 - FI_Timeout, 1740
- first_visible
 - FI_Tree, 1789
- first_visible_item
 - FI_Tree, 1789
- first_window
 - Windows handling functions, 233
- fix_scrollbar_order
 - FI_Scroll, 1369
- fixed
 - FI_Flex, 718, 719
- Fl, 366
 - abi_check, 377
 - abi_version, 377
 - add_check, 377
 - add_fd, 377, 378
 - add_idle, 378
 - add_timeout, 378
 - api_version, 379
 - arg, 379
 - args, 380
 - args_to_utf8, 381
 - background, 382
 - background2, 382
 - box_border_radius_max, 382
 - box_color, 383
 - box_dh, 383
 - box_dw, 383
 - box_dx, 383
 - box_dy, 383
 - box_shadow_width, 383, 384
 - check, 384
 - display, 384
 - dnd_text_ops, 384
 - draw_box_active, 384
 - draw_GL_text_with_textures, 385
 - FI_Option, 376
 - flush, 385
 - get_system_colors, 385
 - gl_visual, 385
 - has_timeout, 386
 - help, 397
 - hide_all_windows, 386
 - idle, 397
 - is_scheme, 386
 - menu_linespacing, 387
 - now, 387
 - option, 388, 389
 - OPTION_ARROW_FOCUS, 376
 - OPTION_DND_TEXT, 376
 - OPTION_FNFC_USES_GTK, 376
 - OPTION_FNFC_USES_KDIALOG, 376
 - OPTION_FNFC_USES_ZENITY, 376
 - OPTION_LAST, 377
 - OPTION_PRINTER_USES_GTK, 376
 - OPTION_SHOW_SCALING, 376
 - OPTION_SHOW_TOOLTIPS, 376
 - OPTION_SIMPLE_ZOOM_SHORTCUT, 377
 - OPTION_VISIBLE_FOCUS, 376
 - own_colormap, 389
 - program_should_quit, 389
 - readqueue, 390
 - ready, 390
 - release, 390
 - reload_scheme, 390
 - remove_check, 390
 - remove_next_timeout, 391
 - remove_timeout, 391
 - repeat_timeout, 392
 - run, 392
 - scheme, 392
 - scrollbar_size, 393
 - seconds_between, 393
 - seconds_since, 394
 - set_box_color, 394
 - set_boxtype, 394
 - set_idle, 395
 - ticks_between, 395
 - ticks_since, 395
 - use_high_res_GL, 396
 - version, 396
 - visible_focus, 396
 - visual, 396
 - wait, 397
- Fl.cxx, 2276
 - fl_close_display, 2277
 - fl_disable_wayland, 2277
 - fl_find, 2277
 - fl_open_display, 2277
- Fl.H, 2026, 2027

- FL_ABI_VERSION
 - Enumerations.H, [2001](#)
- fl_access
 - Unicode and UTF-8 functions, [320](#)
- FL_ACTIVATE
 - Enumerations.H, [2010](#)
- fl_add_symbol
 - Drawing functions, [281](#)
- Fl_Adjuster, [398](#)
 - draw, [407](#)
 - Fl_Adjuster, [407](#)
 - handle, [407](#)
 - soft, [407](#), [408](#)
 - value_damage, [408](#)
- Fl_Adjuster.H, [2034](#)
- fl_alert
 - Common Dialog Classes and Functions, [338](#)
- FL_ALIGN_LEFT
 - Enumerations.H, [2014](#)
- FL_ALIGN_TOP
 - Enumerations.H, [2014](#)
- Fl_Anim_GIF_Image, [408](#)
 - ~Fl_Anim_GIF_Image, [415](#)
 - canvas, [415](#)
 - canvas_h, [415](#)
 - canvas_w, [415](#)
 - color_average, [416](#)
 - copy, [416](#)
 - DEBUG_FLAG, [414](#)
 - delay, [416](#)
 - desaturate, [417](#)
 - DONT_RESIZE_CANVAS, [414](#)
 - DONT_SET_AS_IMAGE, [414](#)
 - DONT_START, [413](#)
 - draw, [417](#)
 - Fl_Anim_GIF_Image, [414](#)
 - Flags, [413](#)
 - frame, [417](#)
 - frame_count, [417](#)
 - frame_h, [418](#)
 - frame_uncache, [418](#)
 - frame_w, [419](#)
 - frame_x, [419](#)
 - frame_y, [419](#)
 - frames, [419](#)
 - image, [419](#), [420](#)
 - is_animated, [420](#)
 - load, [420](#)
 - LOG_FLAG, [414](#)
 - loop, [423](#)
 - min_delay, [423](#)
 - name, [420](#)
 - next, [421](#)
 - on_extension_data, [421](#)
 - on_frame_data, [421](#)
 - OPTIMIZE_MEMORY, [414](#)
 - playing, [421](#)
 - resize, [421](#)
 - speed, [422](#)
 - start, [422](#)
 - stop, [422](#)
 - uncache, [422](#)
 - valid, [422](#)
- Fl_Anim_GIF_Image.H, [2034](#)
- fl_antialias
 - Drawing functions, [281](#)
- FL_API_VERSION
 - Enumerations.H, [2001](#)
- fl_arc
 - Drawing functions, [281](#), [283](#)
- fl_arc.cxx, [2278](#)
- FL_ARROW_CHOICE
 - Enumerations.H, [2004](#)
- FL_ARROW_DOUBLE
 - Enumerations.H, [2004](#)
- FL_ARROW_RETURN
 - Enumerations.H, [2004](#)
- FL_ARROW_SINGLE
 - Enumerations.H, [2004](#)
- Fl_Arrow_Type
 - Enumerations.H, [2003](#)
- fl_ask
 - Common Dialog Classes and Functions, [338](#)
- fl_ask.cxx, [2278](#)
- fl_ask.H, [2036](#), [2038](#)
 - Fl_BEEP, [2037](#)
 - FL_BEEP_DEFAULT, [2038](#)
 - FL_BEEP_ERROR, [2038](#)
 - FL_BEEP_MESSAGE, [2038](#)
 - FL_BEEP_NOTIFICATION, [2038](#)
 - FL_BEEP_PASSWORD, [2038](#)
 - FL_BEEP_QUESTION, [2038](#)
 - fl_message_position, [2038](#)
- fl_attr.h, [2039](#), [2040](#)
 - __fl_attr, [2040](#)
 - FL_DEPRECATED, [2040](#)
 - FL_OVERRIDE, [2040](#)
- Fl_BEEP
 - fl_ask.H, [2037](#)
- fl_bEEP
 - Common Dialog Classes and Functions, [338](#)
- FL_BEEP_DEFAULT
 - fl_ask.H, [2038](#)
- FL_BEEP_ERROR
 - fl_ask.H, [2038](#)
- FL_BEEP_MESSAGE
 - fl_ask.H, [2038](#)
- FL_BEEP_NOTIFICATION
 - fl_ask.H, [2038](#)
- FL_BEEP_PASSWORD
 - fl_ask.H, [2038](#)
- FL_BEEP_QUESTION
 - fl_ask.H, [2038](#)
- fl_begin_complex_polygon
 - Drawing functions, [284](#)
- fl_begin_offscreen

- Drawing functions, 284
- fl_begin_points
 - Drawing functions, 284
- Fl_Bitmap, 423
 - copy, 427
 - draw, 427
 - Fl_Bitmap, 426
 - label, 428
 - uncache, 428
- Fl_Bitmap.H, 2042
- Fl_BMP_Image, 428
 - Fl_BMP_Image, 432
- Fl_BMP_Image.H, 2043
- FL_BORDER_BOX
 - Enumerations.H, 2004
- FL_BORDER_FRAME
 - Enumerations.H, 2004
- Fl_Box, 433
 - draw, 440
 - Fl_Box, 439, 440
 - handle, 440
- fl_box
 - Enumerations.H, 2013
- Fl_Box.H, 2044
- Fl_Boxtype
 - Enumerations.H, 2004
- fl_boxtype.cxx, 2280
 - fl_internal_boxtype, 2281
 - fl_rectbound, 2281
- Fl_Browser, 441
 - _remove, 455
 - add, 455
 - bottomline, 456
 - clear, 456
 - column_char, 456
 - column_widths, 456, 457
 - data, 457
 - display, 457
 - displayed, 458
 - find_line, 458
 - Fl_Browser, 455
 - format_char, 458, 459
 - full_height, 459
 - hide, 460
 - icon, 460
 - incr_height, 460
 - insert, 461
 - item_at, 461
 - item_draw, 462
 - item_first, 462
 - item_height, 462
 - item_last, 463
 - item_next, 463
 - item_prev, 463
 - item_select, 463
 - item_selected, 464
 - item_swap, 464
 - item_text, 464
 - item_width, 465
 - lineno, 465
 - lineposition, 465
 - load, 466
 - make_visible, 466
 - middleline, 466
 - move, 466
 - remove, 467
 - remove_icon, 467
 - select, 467
 - selected, 468
 - show, 468
 - size, 468
 - swap, 468, 469
 - text, 469
 - textsize, 469
 - topline, 470
 - value, 470
 - visible, 470
- Fl_Browser.H, 2044
- Fl_Browser_, 471
 - ALWAYS_ON, 483
 - bbox, 483
 - BOTH, 483
 - BOTH_ALWAYS, 483
 - deleting, 483
 - deselect, 483
 - display, 484
 - displayed, 484
 - draw, 484
 - find_item, 484
 - Fl_Browser_, 483
 - full_height, 485
 - full_width, 485
 - handle, 485
 - has_scrollbar, 485
 - HORIZONTAL, 483
 - HORIZONTAL_ALWAYS, 483
 - hposition, 486
 - hscrollbar, 495
 - incr_height, 486
 - inserting, 486
 - item_at, 487
 - item_draw, 487
 - item_first, 487
 - item_height, 487
 - item_last, 487
 - item_next, 488
 - item_prev, 488
 - item_quick_height, 488
 - item_select, 488
 - item_selected, 489
 - item_swap, 489
 - item_text, 489
 - item_width, 489
 - leftedge, 490
 - linespacing, 490
 - new_list, 490

- position, [490](#)
- redraw_line, [491](#)
- redraw_lines, [491](#)
- replacing, [491](#)
- resize, [491](#)
- scrollbar, [495](#)
- scrollbar_left, [491](#)
- scrollbar_right, [492](#)
- scrollbar_size, [492](#)
- scrollbar_width, [492](#)
- select, [493](#)
- select_only, [493](#)
- selection, [493](#)
- sort, [493](#)
- swapping, [494](#)
- textfont, [494](#)
- VERTICAL, [483](#)
- VERTICAL_ALWAYS, [483](#)
- vposition, [494](#)
- Fl_Browser.H, [2046](#)
- FL_BUTTON
 - Enumerations.H, [2002](#)
- Fl_Button, [495](#)
 - clear, [503](#)
 - compact, [503](#), [504](#)
 - down_box, [504](#), [505](#)
 - draw, [505](#)
 - Fl_Button, [503](#)
 - handle, [505](#)
 - set, [506](#)
 - shortcut, [506](#)
 - value, [506](#)
- Fl_Button.H, [2048](#)
- Fl_Cairo.H, [2049](#)
- Fl_Cairo_State, [507](#)
 - cc, [507](#)
- Fl_Cairo_Window, [507](#)
 - draw, [521](#)
 - set_draw_cb, [521](#)
- Fl_Cairo_Window.H, [2050](#), [2051](#)
- fl_callback_macros.H, [2051](#), [2054](#)
 - FL_FUNCTION_CALLBACK_3, [2052](#)
 - FL_INLINE_CALLBACK_2, [2052](#)
 - FL_METHOD_CALLBACK_1, [2053](#)
- Fl_Callback_Reason
 - Enumerations.H, [2006](#)
- fl_callback_reason_names
 - Events handling functions, [249](#)
- Fl_Callback_User_Data, [521](#)
- fl_can_do_alpha_blending
 - Drawing functions, [284](#)
- FL_CAP_FLAT
 - Drawing functions, [281](#)
- FL_CAP_ROUND
 - Drawing functions, [281](#)
- FL_CAP_SQUARE
 - Drawing functions, [281](#)
- fl_capture_window
 - Drawing functions, [284](#)
- fl_casenumERICsort
 - numericSort.c, [2314](#)
- fl_casts.H, [2060](#)
- Fl_Chart, [522](#)
 - add, [530](#)
 - autosize, [530](#)
 - bounds, [532](#)
 - draw, [532](#)
 - draw_barchart, [532](#)
 - draw_horbarchart, [533](#)
 - draw_linechart, [533](#)
 - draw_piechart, [534](#)
 - Fl_Chart, [530](#)
 - insert, [534](#)
 - maxsize, [534](#)
 - replace, [535](#)
 - size, [535](#)
- Fl_Chart.H, [2060](#), [2061](#)
- FL_CHART_ENTRY, [535](#)
- fl_chdir
 - Unicode and UTF-8 functions, [321](#)
- Fl_Check_Browser, [536](#)
 - add, [548](#)
 - handle, [548](#)
 - item_at, [548](#)
 - item_draw, [548](#)
 - item_first, [548](#)
 - item_height, [549](#)
 - item_next, [549](#)
 - item_prev, [549](#)
 - item_select, [549](#)
 - item_selected, [550](#)
 - item_swap, [550](#)
 - item_text, [550](#)
 - item_width, [550](#)
 - nitems, [551](#)
 - remove, [551](#)
- Fl_Check_Browser.H, [2062](#)
- Fl_Check_Button, [551](#)
 - Fl_Check_Button, [559](#)
- Fl_Check_Button.H, [2063](#)
- fl_chmod
 - Unicode and UTF-8 functions, [321](#)
- Fl_Choice, [559](#)
 - draw, [570](#)
 - Fl_Choice, [570](#)
 - handle, [570](#)
 - value, [571](#)
- fl_choice
 - Common Dialog Classes and Functions, [338](#)
- Fl_Choice.H, [2064](#)
- fl_choice_n
 - Common Dialog Classes and Functions, [340](#)
- fl_circle
 - Drawing functions, [285](#)
- fl_clip
 - Drawing functions, [285](#)

- fl_clip_box
 - Drawing functions, [285](#)
- fl_clip_region
 - Drawing functions, [286](#)
- Fl_Clock, [571](#)
 - Fl_Clock, [579](#), [580](#)
 - handle, [580](#)
- Fl_Clock.H, [2064](#)
- Fl_Clock_Output, [581](#)
 - draw, [589](#)
 - Fl_Clock_Output, [589](#)
 - hour, [589](#)
 - minute, [589](#)
 - second, [589](#)
 - shadow, [590](#)
 - value, [590](#), [591](#)
- FL_CLOSE
 - Enumerations.H, [2009](#)
- fl_close_display
 - Fl.cxx, [2277](#)
- fl_close_fd
 - Unicode and UTF-8 functions, [321](#)
- fl_cmap.h, [2281](#)
- fl_color
 - Color & Font functions, [263](#)
- fl_color.cxx, [2285](#)
- fl_color_average
 - Color & Font functions, [264](#)
- Fl_Color_Chooser, [591](#)
 - b, [601](#)
 - Fl_Color_Chooser, [601](#)
 - g, [601](#)
 - handle, [601](#)
 - hsv, [602](#)
 - hsv2rgb, [602](#)
 - hue, [602](#)
 - mode, [602](#), [603](#)
 - r, [603](#)
 - rgb, [603](#)
 - rgb2hsv, [603](#)
 - saturation, [603](#)
 - value, [604](#)
- fl_color_chooser
 - Common Dialog Classes and Functions, [340](#), [341](#)
- Fl_Color_Chooser.H, [2065](#), [2066](#)
- fl_color_cube
 - Enumerations.H, [2013](#)
- Fl_compose.cxx, [2285](#)
- fl_contrast
 - Color & Font functions, [264](#)
- fl_contrast.cxx, [2285](#)
- FL_CONTRAST_CIELAB
 - Enumerations.H, [2007](#)
- FL_CONTRAST_CUSTOM
 - Enumerations.H, [2007](#)
- Fl_Contrast_Function
 - Enumerations.H, [2003](#)
- fl_contrast_function
 - Color & Font functions, [265](#)
- FL_CONTRAST_LAST
 - Enumerations.H, [2007](#)
- FL_CONTRAST_LEGACY
 - Enumerations.H, [2007](#)
- fl_contrast_level
 - Color & Font functions, [266](#)
- Fl_Contrast_Mode
 - Enumerations.H, [2007](#)
- fl_contrast_mode
 - Color & Font functions, [267](#)
- FL_CONTRAST_NONE
 - Enumerations.H, [2007](#)
- fl_copy_offscreen
 - Drawing functions, [287](#)
- Fl_Copy_Surface, [604](#)
 - Fl_Copy_Surface, [606](#)
 - is_current, [606](#)
 - origin, [606](#)
 - printable_rect, [607](#)
 - set_current, [607](#)
 - translate, [607](#)
 - untranslate, [608](#)
- Fl_Copy_Surface.H, [2067](#)
- Fl_Counter, [608](#)
 - arrow_widths, [617](#)
 - draw, [617](#)
 - Fl_Counter, [617](#)
 - handle, [617](#)
 - lstep, [618](#)
 - step, [618](#)
- Fl_Counter.H, [2068](#)
- fl_create_offscreen
 - Drawing functions, [287](#)
- Fl_Cursor
 - Enumerations.H, [2007](#)
- fl_cursor
 - Drawing functions, [287](#)
- FL_CURSOR_ARROW
 - Enumerations.H, [2007](#)
- FL_CURSOR_CROSS
 - Enumerations.H, [2007](#)
- FL_CURSOR_DEFAULT
 - Enumerations.H, [2007](#)
- FL_CURSOR_E
 - Enumerations.H, [2007](#)
- FL_CURSOR_HAND
 - Enumerations.H, [2007](#)
- FL_CURSOR_HELP
 - Enumerations.H, [2007](#)
- FL_CURSOR_INSERT
 - Enumerations.H, [2007](#)
- FL_CURSOR_MOVE
 - Enumerations.H, [2007](#)
- FL_CURSOR_N
 - Enumerations.H, [2007](#)
- FL_CURSOR_NE
 - Enumerations.H, [2007](#)

- FL_CURSOR_NESW
 - Enumerations.H, [2007](#)
- FL_CURSOR_NONE
 - Enumerations.H, [2007](#)
- FL_CURSOR_NS
 - Enumerations.H, [2007](#)
- FL_CURSOR_NW
 - Enumerations.H, [2007](#)
- FL_CURSOR_NWSE
 - Enumerations.H, [2007](#)
- FL_CURSOR_S
 - Enumerations.H, [2007](#)
- FL_CURSOR_SE
 - Enumerations.H, [2007](#)
- FL_CURSOR_SW
 - Enumerations.H, [2007](#)
- FL_CURSOR_W
 - Enumerations.H, [2007](#)
- FL_CURSOR_WAIT
 - Enumerations.H, [2007](#)
- FL_CURSOR_WE
 - Enumerations.H, [2007](#)
- fl_curve
 - Drawing functions, [288](#)
- fl_curve.cxx, [2286](#)
- Fl_Damage
 - Enumerations.H, [2007](#)
- FL_DAMAGE_ALL
 - Enumerations.H, [2008](#)
- FL_DAMAGE_CHILD
 - Enumerations.H, [2008](#)
- FL_DAMAGE_EXPOSE
 - Enumerations.H, [2008](#)
- FL_DAMAGE_OVERLAY
 - Enumerations.H, [2008](#)
- FL_DAMAGE_SCROLL
 - Enumerations.H, [2008](#)
- FL_DAMAGE_USER1
 - Enumerations.H, [2008](#)
- FL_DAMAGE_USER2
 - Enumerations.H, [2008](#)
- FL_DASH
 - Drawing functions, [281](#)
- FL_DASHDOT
 - Drawing functions, [281](#)
- FL_DASHDOTDOT
 - Drawing functions, [281](#)
- FL_DEACTIVATE
 - Enumerations.H, [2010](#)
- fl_decode_uri
 - File names and URI utility functions, [352](#)
- fl_define_FL_EMBOSSSED_LABEL
 - Enumerations.H, [2013](#)
- fl_define_FL_ENGRAVED_LABEL
 - Enumerations.H, [2013](#)
- fl_define_FL_ICON_LABEL
 - Enumerations.H, [2013](#)
- fl_define_FL_IMAGE_LABEL
 - Enumerations.H, [2013](#)
- fl_define_FL_MULTI_LABEL
 - Enumerations.H, [2013](#)
- fl_define_FL_SHADOW_LABEL
 - Enumerations.H, [2013](#)
- fl_delete_offscreen
 - Drawing functions, [288](#)
- FL_DEPRECATED
 - fl_attr.h, [2040](#)
- Fl_Device.H, [2069](#)
- Fl_Device_Plugin, [619](#)
 - rectangle_capture, [620](#)
- Fl_Dial, [620](#)
 - angle1, [629](#)
 - draw, [629](#)
 - Fl_Dial, [629](#)
 - handle, [629](#)
- Fl_Dial.H, [2070](#)
- fl_dir_chooser
 - Common Dialog Classes and Functions, [342](#)
- fl_disable_wayland
 - Fl.cxx, [2277](#)
- Fl_Display_Device, [630](#)
- FL_DND_DRAG
 - Enumerations.H, [2010](#)
- FL_DND_ENTER
 - Enumerations.H, [2010](#)
- FL_DND_LEAVE
 - Enumerations.H, [2010](#)
- FL_DND_RELEASE
 - Enumerations.H, [2010](#)
- FL_DOT
 - Drawing functions, [281](#)
- Fl_Double_Window, [631](#)
 - ~Fl_Double_Window, [643](#)
 - as_double_window, [643](#)
 - flush, [643](#)
 - hide, [643](#)
 - resize, [644](#)
 - show, [644](#)
- Fl_Double_Window.cxx, [2286](#)
- Fl_Double_Window.H, [2071](#)
- fl_down
 - Enumerations.H, [2013](#)
- FL_DOWN_BOX
 - Enumerations.H, [2004](#)
- FL_DOWN_FRAME
 - Enumerations.H, [2004](#)
- FL_DRAG
 - Enumerations.H, [2009](#)
- fl_draw
 - Drawing functions, [288–290](#)
- fl_draw.H, [2071](#), [2077](#)
- fl_draw_arrow
 - Drawing functions, [290](#)
- fl_draw_box
 - Drawing functions, [290](#)
- fl_draw_box_focus

- Drawing functions, [291](#)
- `fl_draw_check`
 - Drawing functions, [291](#)
- `fl_draw_circle`
 - Drawing functions, [291](#)
- `fl_draw_image`
 - Drawing functions, [292](#)
- `fl_draw_image_mono`
 - Drawing functions, [293](#)
- `fl_draw_pixmap`
 - Drawing functions, [294](#)
- `fl_draw_radio`
 - Drawing functions, [294](#)
- `fl_draw_symbol`
 - Drawing functions, [295](#)
- `FL_EMBOSSSED_BOX`
 - Enumerations.H, [2004](#)
- `FL_EMBOSSSED_FRAME`
 - Enumerations.H, [2004](#)
- `FI_End`, [645](#)
- `FL_ENGRAVED_BOX`
 - Enumerations.H, [2004](#)
- `FL_ENGRAVED_FRAME`
 - Enumerations.H, [2004](#)
- `FL_ENTER`
 - Enumerations.H, [2008](#)
- `FI_EPS_File_Surface`, [645](#)
 - `~FI_EPS_File_Surface`, [647](#)
 - `close`, [648](#)
 - `FI_EPS_File_Surface`, [647](#)
 - `origin`, [648](#)
 - `printable_rect`, [648](#)
 - `translate`, [648](#)
 - `untranslate`, [649](#)
- `FI_Event`
 - Enumerations.H, [2008](#)
- `FI_Event_Dispatch`
 - Callback Function Typedefs, [232](#)
- `fl_eventnames`
 - Events handling functions, [249](#)
- `FL_EXCEPT`
 - Enumerations.H, [2003](#)
- `fl_expand_text`
 - Drawing functions, [295](#)
- `FI_Export.H`, [2082](#)
- `FI_File_Browser`, [649](#)
 - `errmsg`, [663](#)
 - `filetype`, [663](#)
 - `filter`, [663](#)
 - `FI_File_Browser`, [663](#)
 - `iconsize`, [663](#)
 - `load`, [664](#)
- `FI_File_Browser.H`, [2083](#)
- `FI_File_Chooser`, [664](#)
 - `add_extra`, [669](#)
 - `CREATE`, [669](#)
 - `DIRECTORY`, [669](#)
 - `filter`, [670](#)
 - `FI_File_Chooser`, [669](#)
 - `iconsize`, [670](#)
 - `MULTI`, [669](#)
 - `preview`, [670](#)
 - `showHiddenButton`, [671](#)
 - `shown`, [670](#)
 - `SINGLE`, [669](#)
 - `Type`, [669](#)
 - `value`, [670](#), [671](#)
- `fl_file_chooser`
 - Common Dialog Classes and Functions, [342](#)
- `FI_File_Chooser.H`, [2084](#)
- `fl_file_chooser_callback`
 - Common Dialog Classes and Functions, [343](#)
- `fl_file_chooser_ok_label`
 - Common Dialog Classes and Functions, [344](#)
- `FI_File_Icon`, [671](#)
 - `add`, [673](#)
 - `add_color`, [673](#)
 - `add_vertex`, [673](#), [674](#)
 - `draw`, [674](#)
 - `find`, [674](#)
 - `FI_File_Icon`, [673](#)
 - `label`, [674](#)
 - `labeltype`, [675](#)
 - `load`, [675](#)
 - `load_fti`, [675](#)
 - `load_image`, [675](#)
 - `load_system_icons`, [676](#)
 - `next`, [676](#)
 - `type`, [676](#)
- `FI_File_Icon.H`, [2086](#)
- `FI_File_Input`, [676](#)
 - `down_box`, [687](#)
 - `draw`, [687](#)
 - `errorcolor`, [687](#)
 - `FI_File_Input`, [687](#)
 - `handle`, [687](#)
 - `value`, [688](#)
- `FI_File_Input.H`, [2088](#)
- `FI_File_Sort_F`
 - File names and URI utility functions, [351](#)
- `fl_filename_absolute`
 - File names and URI utility functions, [352](#)
- `fl_filename_absolute_str`
 - File names and URI utility functions, [352](#), [353](#)
- `fl_filename_expand`
 - File names and URI utility functions, [353](#)
- `fl_filename_expand_str`
 - File names and URI utility functions, [354](#)
- `fl_filename_ext`
 - File names and URI utility functions, [354](#)
- `fl_filename_ext_str`
 - File names and URI utility functions, [354](#)
- `fl_filename_free_list`
 - File names and URI utility functions, [354](#)
- `fl_filename_isdir`
 - File names and URI utility functions, [355](#)

- fl_filename_list
 - File names and URI utility functions, [355](#)
- fl_filename_match
 - File names and URI utility functions, [356](#)
- fl_filename_name
 - File names and URI utility functions, [357](#)
- fl_filename_name_str
 - File names and URI utility functions, [357](#)
- fl_filename_path_str
 - File names and URI utility functions, [357](#)
- fl_filename_relative
 - File names and URI utility functions, [357](#), [358](#)
- fl_filename_relative_str
 - File names and URI utility functions, [359](#)
- fl_filename_setext
 - File names and URI utility functions, [359](#)
- fl_filename_setext_str
 - File names and URI utility functions, [360](#)
- FI_Fill_Dial, [688](#)
- FI_Fill_Dial.H, [2088](#)
- FI_Fill_Slider, [697](#)
- FI_Fill_Slider.H, [2089](#)
- fl_find
 - Fl.cxx, [2277](#)
- FL_FLAT_BOX
 - Enumerations.H, [2004](#)
- FI_Flex, [705](#)
 - alloc_size, [718](#)
 - COLUMN, [716](#)
 - draw, [718](#)
 - end, [718](#)
 - fixed, [718](#), [719](#)
 - FI_Flex, [716](#), [717](#)
 - gap, [719](#)
 - HORIZONTAL, [716](#)
 - horizontal, [719](#)
 - layout, [720](#)
 - margin, [720](#), [721](#)
 - need_layout, [721](#), [722](#)
 - on_remove, [722](#)
 - resize, [722](#)
 - ROW, [716](#)
 - spacing, [722](#)
 - VERTICAL, [716](#)
- FI_Flex.H, [2089](#)
- FI_Float_Input, [723](#)
 - FI_Float_Input, [733](#)
- FI_Float_Input.H, [2091](#)
- FL_FOCUS
 - Enumerations.H, [2009](#)
- fl_focus_rect
 - Drawing functions, [296](#)
- fl_font
 - Color & Font functions, [268](#)
- fl_fontnames
 - Events handling functions, [249](#)
- FI_Fontsize
 - Enumerations.H, [2003](#)
- fl_fopen
 - Unicode and UTF-8 functions, [322](#)
- FI_FormsBitmap, [733](#)
 - draw, [740](#)
 - set, [740](#)
- FI_FormsBitmap.H, [2091](#)
- FI_FormsPixmap, [740](#)
 - draw, [747](#)
 - FI_FormsPixmap, [747](#)
 - Pixmap, [748](#)
 - set, [748](#)
- FI_FormsPixmap.H, [2092](#)
- FI_FormsText, [748](#)
 - draw, [755](#)
- fl_frame
 - Drawing functions, [296](#)
 - Enumerations.H, [2014](#)
- fl_frame2
 - Drawing functions, [296](#)
- FI_Free, [755](#)
 - draw, [763](#)
 - FI_Free, [762](#)
 - handle, [763](#)
- FI_Free.H, [2092](#)
- FL_FREE_BOXTYPE
 - Enumerations.H, [2006](#)
- FL_FREE_LABELTYPE
 - Enumerations.H, [2011](#)
- FL_FULLSCREEN
 - Enumerations.H, [2011](#)
- FL_FUNCTION_CALLBACK_3
 - fl_callback_macros.H, [2052](#)
- fl_gap
 - Drawing functions, [297](#)
- FI_get_system_colors.cxx, [2286](#)
 - fl_parse_color, [2288](#)
- fl_getcwd
 - Unicode and UTF-8 functions, [322](#)
- fl_getcwd_str
 - File names and URI utility functions, [360](#)
- fl_getenv
 - Unicode and UTF-8 functions, [322](#)
- FI_GIF_Image, [764](#)
 - animate, [768](#)
 - FI_GIF_Image, [767](#), [768](#)
- FI_GIF_Image.H, [2093](#)
- FI_GIF_Image::GIF_FRAME, [1979](#)
- FI_GIF_Image::GIF_FRAME::CPAL, [365](#)
- FI_Gl_Choice, [769](#)
- FI_Gl_Choice.H, [2288](#)
- FI_Gl_Window, [769](#)
 - as_gl_window, [783](#)
 - can_do, [784](#)
 - can_do_overlay, [784](#)
 - context, [784](#)
 - context_valid, [784](#)
 - draw, [784](#)
 - draw_begin, [785](#)

- draw_end, 785
- Fl_Gl_Window, 783
- flush, 785
- handle, 786
- hide, 786
- make_current, 786
- make_overlay_current, 786
- mode, 786
- ortho, 787
- pixel_h, 787
- pixel_w, 788
- pixels_per_unit, 788
- redraw_overlay, 788
- resize, 788
- show, 788
- swap_buffers, 789
- swap_interval, 789
- valid, 789
- Fl_Gl_Window.H, 2094
- Fl_Gl_Window_Driver.H, 2289
- Fl_Glut_Bitmap_Font, 790
- Fl_Glut_StrokeChar, 790
- Fl_Glut_StrokeFont, 790
- Fl_Glut_StrokeStrip, 791
- Fl_Glut_StrokeVertex, 791
- Fl_Glut_Window, 791
 - draw, 805
 - draw_overlay, 806
 - handle, 806
- Fl_Graphics_Driver.cxx, 2290
- Fl_Graphics_Driver.H, 2095
- fl_gray_ramp
 - Enumerations.H, 2014
- Fl_Grid, 806
 - cell, 818, 819
 - clear_layout, 819
 - col_gap, 819
 - col_weight, 820
 - col_width, 821
 - debug, 821
 - draw, 822
 - draw_grid, 822
 - Fl_Grid, 818
 - gap, 822
 - layout, 823
 - margin, 823, 824
 - need_layout, 824
 - on_remove, 825
 - resize, 825
 - row_gap, 825
 - row_height, 825, 826
 - row_weight, 826
 - show_grid, 826, 827
 - widget, 827, 828
- Fl_Grid.cxx, 2290
- Fl_Grid.H, 2100, 2101
- Fl_Grid::Cell, 363
 - ~Cell, 363
 - next, 364
- Fl_Group, 828
 - ~Fl_Group, 837
 - array, 838
 - as_group, 838
 - begin, 838
 - bounds, 838
 - child, 839
 - clear, 839
 - clip_children, 839, 840
 - current, 840
 - delete_child, 840
 - draw, 841
 - draw_child, 841
 - draw_children, 841
 - end, 841
 - find, 841
 - Fl_Group, 837
 - focus, 842
 - handle, 842
 - init_sizes, 842
 - insert, 843
 - on_insert, 843
 - on_move, 843
 - on_remove, 844
 - remove, 844
 - resizable, 845
 - resize, 846
 - sizes, 847
 - update_child, 847
- Fl_Group.H, 2104
- fl_height
 - Color & Font functions, 268
- Fl_Help_Block, 848
- Fl_Help_Dialog, 848
 - load, 849
 - show, 849
 - textsize, 850
 - value, 850
- Fl_Help_Dialog.H, 2106
- Fl_Help_Font_Stack, 850
- Fl_Help_Font_Style, 851
- Fl_Help_Link, 851
- Fl_Help_Target, 852
- Fl_Help_View, 852
 - ~Fl_Help_View, 863
 - copy, 863
 - draw, 864
 - find, 864
 - handle, 864
 - leftline, 864
 - link, 865
 - load, 865
 - resize, 865
 - scrollbar_size, 865, 866
 - text_selected, 866
 - topline, 866
 - value, 866

- FI_Help_View.H, [2107](#)
- FL_HIDE
 - Enumerations.H, [2010](#)
- FI_Hold_Browser, [867](#)
 - FI_Hold_Browser, [880](#)
- FI_Hold_Browser.H, [2110](#)
- FI_Hor_Fill_Slider, [881](#)
- FI_Hor_Fill_Slider.H, [2110](#)
- FI_Hor_Nice_Slider, [889](#)
- FI_Hor_Nice_Slider.H, [2111](#)
- FI_Hor_Slider, [898](#)
- FI_Hor_Slider.H, [2111](#)
- FI_Hor_Value_Slider, [906](#)
- FI_Hor_Value_Slider.H, [2112](#)
- FI_ICO_Image, [915](#)
 - FI_ICO_Image, [918](#)
- FI_ICO_Image.H, [2112](#)
- FI_ICO_Image::IconDirEntry, [1979](#)
- FI_Image, [919](#)
 - as_shared_image, [921](#)
 - color_average, [921](#)
 - copy, [922](#)
 - count, [923](#)
 - d, [923](#)
 - data, [923](#)
 - desaturate, [924](#)
 - draw, [924](#)
 - draw_empty, [924](#)
 - draw_scaled, [924](#)
 - fail, [926](#)
 - FI_Image, [921](#)
 - h, [926](#)
 - inactive, [926](#)
 - label, [927](#)
 - ld, [927](#)
 - release, [927](#)
 - RGB_scaling, [927](#)
 - scale, [928](#)
 - scaling_algorithm, [928](#)
 - uncache, [928](#)
 - w, [928](#), [929](#)
- FI_Image.H, [2113](#)
 - FI_RGB_Scaling, [2113](#)
 - FL_RGB_SCALING_BILINEAR, [2113](#)
 - FL_RGB_SCALING_NEAREST, [2113](#)
- FL_IMAGE_LABEL
 - Enumerations.H, [2002](#)
- FI_Image_Reader, [929](#)
- FI_Image_Reader.h, [2291](#)
- FI_Image_Surface, [929](#)
 - FI_Image_Surface, [932](#)
 - highres_image, [932](#)
 - image, [932](#)
 - is_current, [932](#)
 - mask, [932](#)
 - offscreen, [933](#)
 - origin, [933](#), [934](#)
 - printable_rect, [934](#)
 - rescale, [934](#)
 - set_current, [934](#)
 - translate, [935](#)
 - untranslate, [935](#)
- FI_Image_Surface.H, [2115](#)
- FL_INLINE_CALLBACK_2
 - fl_callback_macros.H, [2052](#)
- FI_Input, [935](#)
 - draw, [947](#)
 - FI_Input, [946](#)
 - handle, [947](#)
 - handle_key, [947](#)
 - handle_rmb, [948](#)
- fl_input
 - Common Dialog Classes and Functions, [344](#)
- FI_Input.H, [2116](#)
- FI_Input_, [948](#)
 - ~FI_Input_, [958](#)
 - append, [958](#)
 - apply_undo, [958](#)
 - can_redo, [958](#)
 - can_undo, [959](#)
 - copy, [959](#)
 - copy_cuts, [959](#)
 - cursor_color, [959](#)
 - cut, [960](#)
 - drawtext, [960](#), [961](#)
 - dvalue, [961](#)
 - FI_Input_, [957](#)
 - handle_mouse, [961](#)
 - handletext, [962](#)
 - index, [962](#)
 - input_type, [962](#)
 - insert, [963](#)
 - insert_position, [963](#)
 - ivalue, [964](#)
 - line_end, [964](#)
 - line_start, [964](#)
 - mark, [965](#)
 - maximum_size, [965](#)
 - position, [965](#), [966](#)
 - readonly, [966](#)
 - redo, [966](#)
 - replace, [966](#)
 - resize, [967](#)
 - shortcut, [967](#)
 - size, [968](#)
 - static_value, [968](#)
 - tab_nav, [969](#)
 - textcolor, [969](#), [970](#)
 - textfont, [970](#)
 - textsize, [970](#)
 - undo, [972](#)
 - up_down_position, [972](#)
 - value, [972](#), [973](#)
 - word_end, [974](#)
 - word_start, [974](#)
 - wrap, [974](#)

- Fl_Input_.H, [2117](#)
- Fl_Input_Choice, [975](#)
 - add, [986](#)
 - draw, [986](#)
 - Fl_Input_Choice, [986](#)
 - inp_x, [986](#)
 - input, [986](#)
 - menu_x, [986](#)
 - menubutton, [986](#)
 - resize, [987](#)
 - update_menubutton, [987](#)
 - value, [987](#)
- Fl_Input_Choice.H, [2120](#)
- Fl_Int_Input, [988](#)
 - Fl_Int_Input, [998](#)
- Fl_Int_Input.H, [2122](#)
- Fl_Int_Vector.H, [2292](#)
- fl_internal_boxtype
 - fl_boxtype.cxx, [2281](#)
- fl_intptr_t
 - platform_types.h, [2265](#)
- FL_JOIN_BEVEL
 - Drawing functions, [281](#)
- FL_JOIN_MITER
 - Drawing functions, [281](#)
- FL_JOIN_ROUND
 - Drawing functions, [281](#)
- Fl_JPEG_Image, [998](#)
 - Fl_JPEG_Image, [1001](#)
- Fl_JPEG_Image.H, [2122](#)
- FL_KEYBOARD
 - Enumerations.H, [2009](#)
- FL_KEYDOWN
 - Enumerations.H, [2009](#)
- FL_KEYUP
 - Enumerations.H, [2009](#)
- Fl_Label, [1002](#)
 - draw, [1003](#)
 - measure, [1003](#)
 - type, [1003](#)
- Fl_Labeltype
 - Enumerations.H, [2011](#)
- fl_latin1_to_local
 - Color & Font functions, [269](#)
- FL_LEAVE
 - Enumerations.H, [2009](#)
- Fl_Light_Button, [1004](#)
 - draw, [1012](#)
 - Fl_Light_Button, [1011](#)
 - handle, [1012](#)
- Fl_Light_Button.H, [2123](#)
- fl_lightness
 - Color & Font functions, [269](#)
- Fl_Line_Dial, [1013](#)
- Fl_Line_Dial.H, [2123](#)
- fl_line_style
 - Drawing functions, [297](#)
- fl_load_matrix
 - Drawing functions, [297](#)
- fl_local_to_latin1
 - Color & Font functions, [269](#)
- fl_local_to_mac_roman
 - Color & Font functions, [270](#)
- fl_luminance
 - Color & Font functions, [270](#)
- Fl_Mac_App_Menu, [1021](#)
 - custom_application_menu_items, [1022](#)
 - print, [1023](#)
- fl_mac_os_version
 - Mac OS X-specific symbols, [336](#)
- fl_mac_roman_to_local
 - Color & Font functions, [270](#)
- fl_mac_set_about
 - Mac OS X-specific symbols, [335](#)
- FL_MAJOR_VERSION
 - Enumerations.H, [2002](#)
- fl_make_path
 - Unicode and UTF-8 functions, [323](#)
- fl_make_path_for_file
 - Unicode and UTF-8 functions, [323](#)
- FL_MAX_BOXTYPE
 - Enumerations.H, [2006](#)
- fl_measure
 - Drawing functions, [298](#)
- fl_measure_pixmap
 - Drawing functions, [298](#), [299](#)
- Fl_Menu.H, [2124](#)
- Fl_Menu_, [1023](#)
 - add, [1032](#)
 - clear, [1035](#)
 - clear_submenu, [1035](#)
 - copy, [1036](#)
 - down_box, [1036](#)
 - find_index, [1036](#), [1037](#)
 - find_item, [1037](#), [1038](#)
 - find_item_with_argument, [1038](#)
 - find_item_with_user_data, [1038](#)
 - Fl_Menu_, [1032](#)
 - global, [1039](#)
 - insert, [1039](#)
 - item_pathname, [1039](#)
 - menu, [1040](#)
 - menu_box, [1041](#)
 - menu_end, [1041](#)
 - mode, [1041](#), [1042](#)
 - mvalue, [1042](#)
 - picked, [1042](#)
 - prev_mvalue, [1042](#)
 - remove, [1042](#)
 - replace, [1042](#)
 - size, [1043](#)
 - test_shortcut, [1043](#)
 - text, [1043](#)
 - textcolor, [1043](#)
 - textfont, [1043](#), [1044](#)
 - textsize, [1044](#)

- value, [1044](#), [1045](#)
- Fl_Menu_.H, [2124](#)
- Fl_Menu_Bar, [1046](#)
 - draw, [1056](#)
 - Fl_Menu_Bar, [1056](#)
 - handle, [1056](#)
 - play_menu, [1057](#)
 - update, [1057](#)
- Fl_Menu_Bar.H, [2125](#)
- Fl_Menu_Button, [1057](#)
 - draw, [1067](#)
 - Fl_Menu_Button, [1067](#)
 - handle, [1067](#)
 - popup, [1068](#)
 - POPUP1, [1067](#)
 - POPUP12, [1067](#)
 - POPUP123, [1067](#)
 - POPUP13, [1067](#)
 - POPUP2, [1067](#)
 - POPUP23, [1067](#)
 - POPUP3, [1067](#)
 - popup_buttons, [1067](#)
- Fl_Menu_Button.H, [2126](#)
- FL_MENU_DIVIDER
 - Fl_Menu_Item.H, [2127](#)
- FL_MENU_HORIZONTAL
 - Fl_Menu_Item.H, [2127](#)
- FL_MENU_INACTIVE
 - Fl_Menu_Item.H, [2127](#)
- FL_MENU_INVISIBLE
 - Fl_Menu_Item.H, [2127](#)
- Fl_Menu_Item, [1068](#)
 - add, [1073](#)
 - argument, [1074](#)
 - callback, [1074](#)
 - check, [1075](#)
 - checkbox, [1075](#)
 - checked, [1075](#)
 - deactivate, [1075](#)
 - do_callback, [1075](#)
 - find_shortcut, [1076](#)
 - image, [1076](#)
 - image_label, [1076](#)
 - insert, [1076](#)
 - label, [1077](#), [1078](#)
 - labelcolor, [1078](#)
 - labelfont, [1078](#)
 - labeltype, [1079](#)
 - measure, [1079](#)
 - multi_label, [1079](#)
 - next, [1079](#)
 - popup, [1079](#)
 - pulldown, [1080](#)
 - radio, [1080](#)
 - set, [1080](#)
 - setonly, [1080](#)
 - shortcut, [1081](#)
 - size, [1081](#)
 - submenu, [1081](#)
 - test_shortcut, [1081](#)
 - uncheck, [1081](#)
 - value, [1082](#)
- Fl_Menu_Item.H, [2126](#), [2127](#)
 - FL_MENU_DIVIDER, [2127](#)
 - FL_MENU_HORIZONTAL, [2127](#)
 - FL_MENU_INACTIVE, [2127](#)
 - FL_MENU_INVISIBLE, [2127](#)
 - FL_MENU_RADIO, [2127](#)
 - FL_MENU_TOGGLE, [2127](#)
 - FL_MENU_VALUE, [2127](#)
 - FL_SUBMENU, [2127](#)
 - FL_SUBMENU_POINTER, [2127](#)
- FL_MENU_RADIO
 - Fl_Menu_Item.H, [2127](#)
- FL_MENU_TOGGLE
 - Fl_Menu_Item.H, [2127](#)
- FL_MENU_VALUE
 - Fl_Menu_Item.H, [2127](#)
- Fl_Menu_Window, [1082](#)
- Fl_Menu_Window.H, [2130](#)
- fl_message
 - Common Dialog Classes and Functions, [345](#)
- Fl_Message.h, [2293](#)
- fl_message.H, [2130](#)
- fl_message_hotspot
 - Common Dialog Classes and Functions, [345](#)
- fl_message_icon
 - Common Dialog Classes and Functions, [345](#)
- fl_message_icon_label
 - Common Dialog Classes and Functions, [346](#)
- fl_message_position
 - Common Dialog Classes and Functions, [346](#), [347](#)
 - fl_ask.H, [2038](#)
- fl_message_title
 - Common Dialog Classes and Functions, [348](#)
- fl_message_title_default
 - Common Dialog Classes and Functions, [348](#)
- FL_METHOD_CALLBACK_1
 - fl_callback_macros.H, [2053](#)
- FL_MINOR_VERSION
 - Enumerations.H, [2002](#)
- fl_mkdir
 - Unicode and UTF-8 functions, [323](#)
- FL_MOUSEWHEEL
 - Enumerations.H, [2010](#)
- FL_MOVE
 - Enumerations.H, [2010](#)
- fl_mult_matrix
 - Drawing functions, [299](#)
- Fl_Multi_Browser, [1095](#)
 - Fl_Multi_Browser, [1108](#)
- Fl_Multi_Browser.H, [2131](#)
- FL_MULTI_LABEL
 - Enumerations.H, [2002](#)
- Fl_Multi_Label, [1108](#)
 - label, [1110](#)

- labela, [1111](#)
- labelb, [1111](#)
- typea, [1111](#)
- typeb, [1111](#)
- Fl_Multi_Label.H, [2131](#)
- Fl_Multiline_Input, [1111](#)
 - Fl_Multiline_Input, [1121](#)
- Fl_Multiline_Input.H, [2132](#)
- Fl_Multiline_Output, [1121](#)
 - Fl_Multiline_Output, [1132](#)
- Fl_Multiline_Output.H, [2132](#)
- Fl_Native_File_Chooser, [1132](#)
 - ~Fl_Native_File_Chooser, [1135](#)
 - BROWSE_DIRECTORY, [1135](#)
 - BROWSE_FILE, [1135](#)
 - BROWSE_MULTI_DIRECTORY, [1135](#)
 - BROWSE_MULTI_FILE, [1135](#)
 - BROWSE_SAVE_DIRECTORY, [1135](#)
 - BROWSE_SAVE_FILE, [1135](#)
 - count, [1135](#)
 - directory, [1135](#)
 - errmsg, [1135](#)
 - filename, [1136](#)
 - filter, [1136](#)
 - filter_value, [1136](#), [1137](#)
 - Fl_Native_File_Chooser, [1135](#)
 - NEW_FOLDER, [1135](#)
 - NO_OPTIONS, [1134](#)
 - Option, [1134](#)
 - options, [1137](#)
 - preset_file, [1137](#)
 - PREVIEW, [1135](#)
 - SAVEAS_CONFIRM, [1134](#)
 - show, [1137](#)
 - title, [1137](#), [1138](#)
 - Type, [1135](#)
 - USE_FILTER_EXT, [1135](#)
- Fl_Native_File_Chooser.H, [2132](#), [2133](#)
- Fl_Native_File_Chooser_Kdialog.H, [2295](#)
- Fl_Native_File_Chooser_Zenity.H, [2295](#)
- Fl_Nice_Slider, [1138](#)
- Fl_Nice_Slider.H, [2135](#)
- FL_NO_BOX
 - Enumerations.H, [2004](#)
- FL_NO_EVENT
 - Enumerations.H, [2008](#)
- FL_NO_LABEL
 - Enumerations.H, [2011](#)
- fl_nonspacing
 - Unicode and UTF-8 functions, [324](#)
- FL_NORMAL_LABEL
 - Enumerations.H, [2011](#)
- FL_NORMAL_SIZE
 - Enumerations.H, [2014](#)
- fl_not_clipped
 - Drawing functions, [299](#)
- fl_numericsort
 - numeric_sort.c, [2315](#)
- Fl_Object.H, [2135](#)
- Fl_Offscreen
 - platform_types.h, [2265](#)
- fl_old_shortcut
 - Drawing functions, [299](#)
- fl_open
 - Unicode and UTF-8 functions, [324](#)
- fl_open_callback
 - Mac OS X-specific symbols, [335](#)
- fl_open_display
 - Fl.cxx, [2277](#)
- fl_open_ext
 - Unicode and UTF-8 functions, [324](#)
- fl_open_uri
 - File names and URI utility functions, [360](#)
- Fl_Option
 - Fl, [376](#)
- FL_ORIENT_DOWN
 - Enumerations.H, [2012](#)
- FL_ORIENT_LEFT
 - Enumerations.H, [2012](#)
- FL_ORIENT_NE
 - Enumerations.H, [2012](#)
- FL_ORIENT_NONE
 - Enumerations.H, [2012](#)
- FL_ORIENT_NW
 - Enumerations.H, [2012](#)
- FL_ORIENT_RIGHT
 - Enumerations.H, [2012](#)
- FL_ORIENT_SE
 - Enumerations.H, [2012](#)
- FL_ORIENT_SW
 - Enumerations.H, [2012](#)
- FL_ORIENT_UP
 - Enumerations.H, [2012](#)
- Fl_Orientation
 - Enumerations.H, [2011](#)
- Fl_Output, [1146](#)
 - Fl_Output, [1157](#)
- Fl_Output.H, [2136](#)
- fl_overlay_clear
 - Drawing functions, [300](#)
- fl_overlay_rect
 - Drawing functions, [300](#)
- Fl_Overlay_Window, [1157](#)
 - as_overlay_window, [1170](#)
 - draw_overlay, [1170](#)
 - Fl_Overlay_Window, [1170](#)
 - flush, [1170](#)
 - hide, [1170](#)
 - redraw_overlay, [1170](#)
 - resize, [1170](#)
 - show, [1171](#)
- Fl_Overlay_Window.H, [2136](#)
- FL_OVERRIDE
 - fl_attr.h, [2040](#)
- fl_override_scale
 - Drawing functions, [301](#)

- fl_oxy.h, [2296](#)
- Fl_Pack, [1171](#)
 - clear, [1181](#)
 - draw, [1181](#)
 - Fl_Pack, [1180](#)
 - horizontal, [1181](#)
 - resize, [1181](#)
- Fl_Pack.H, [2137](#)
- Fl_Paged_Device, [1182](#)
 - A0, [1184](#)
 - A1, [1184](#)
 - A2, [1184](#)
 - A3, [1184](#)
 - A4, [1185](#)
 - A5, [1185](#)
 - A6, [1185](#)
 - A7, [1185](#)
 - A8, [1185](#)
 - A9, [1185](#)
 - B0, [1185](#)
 - B1, [1185](#)
 - B10, [1185](#)
 - B2, [1185](#)
 - B3, [1185](#)
 - B4, [1185](#)
 - B5, [1185](#)
 - B6, [1185](#)
 - B7, [1185](#)
 - B8, [1185](#)
 - B9, [1185](#)
 - begin_job, [1185](#)
 - begin_page, [1186](#)
 - end_job, [1186](#)
 - end_page, [1186](#)
 - EXECUTIVE, [1185](#)
 - FOLIO, [1185](#)
 - LANDSCAPE, [1185](#)
 - LEDGER, [1185](#)
 - LEGAL, [1185](#)
 - LETTER, [1185](#)
 - margins, [1186](#)
 - ORIENTATION, [1185](#)
 - Page_Format, [1184](#)
 - Page_Layout, [1185](#)
 - PORTRAIT, [1185](#)
 - REVERSED, [1185](#)
 - rotate, [1187](#)
 - scale, [1187](#)
 - start_job, [1187](#)
 - start_page, [1188](#)
 - TABLOID, [1185](#)
- Fl_Paged_Device.cxx, [2296](#)
- Fl_Paged_Device.H, [2137](#), [2138](#)
- Fl_Paged_Device::page_format, [1982](#)
- fl_parse_color
 - Fl_get_system_colors.cxx, [2288](#)
- fl_password
 - Common Dialog Classes and Functions, [348](#), [349](#)
- FL_PASTE
 - Enumerations.H, [2010](#)
- FL_PATCH_VERSION
 - Enumerations.H, [2002](#)
- Fl_PDF_File_Surface, [1188](#)
 - begin_document, [1191](#)
 - begin_job, [1191](#)
 - begin_page, [1192](#)
 - end_job, [1192](#)
 - end_page, [1192](#)
 - is_current, [1192](#)
 - margins, [1192](#)
 - origin, [1193](#)
 - printable_rect, [1193](#)
 - rotate, [1194](#)
 - scale, [1194](#)
 - set_current, [1194](#)
 - translate, [1194](#)
 - untranslate, [1195](#)
- Fl_PDF_File_Surface.H, [2139](#)
- fl_pie
 - Drawing functions, [301](#)
- Fl_Pixmap, [1195](#)
 - color_average, [1198](#)
 - copy, [1198](#)
 - desaturate, [1199](#)
 - draw, [1199](#)
 - Fl_Pixmap, [1198](#)
 - label, [1199](#)
 - uncache, [1200](#)
- Fl_Pixmap.H, [2140](#)
- Fl_Plugin, [1200](#)
 - Fl_Plugin, [1201](#)
- Fl_Plugin.H, [2141](#)
- Fl_Plugin_Manager, [1201](#)
 - ~Fl_Plugin_Manager, [1205](#)
 - addPlugin, [1205](#)
 - load, [1205](#)
 - loadAll, [1205](#)
 - removePlugin, [1205](#)
- Fl_PNG_Image, [1206](#)
 - Fl_PNG_Image, [1209](#)
- Fl_PNG_Image.H, [2141](#)
- Fl_PNM_Image, [1209](#)
 - Fl_PNM_Image, [1212](#)
- Fl_PNM_Image.H, [2142](#)
- fl_polygon
 - Drawing functions, [302](#)
- fl_pop_clip
 - Drawing functions, [302](#)
- Fl_Positioner, [1212](#)
 - draw, [1220](#)
 - Fl_Positioner, [1220](#)
 - handle, [1220](#)
- Fl_Positioner.H, [2142](#)
- Fl_PostScript.H, [2143](#), [2144](#)
 - Fl_PostScript_Close_Command, [2143](#)
- Fl_PostScript_Close_Command

- FI_PostScript.H, 2143
- FI_PostScript_File_Device, 1221
 - begin_job, 1225, 1226
 - begin_page, 1226
 - end_current, 1226
 - end_job, 1226
 - end_page, 1227
 - margins, 1227
 - origin, 1227
 - printable_rect, 1228
 - rotate, 1228
 - scale, 1228
 - set_current, 1229
 - start_job, 1229
 - translate, 1229
 - untranslate, 1229
- FI_Preferences, 1230
 - ~FI_Preferences, 1238
 - C_LOCALE, 1235
 - CLEAR, 1235
 - CORE, 1235
 - CORE_READ_OK, 1252
 - CORE_SYSTEM, 1235
 - CORE_SYSTEM_L, 1235
 - CORE_USER, 1235
 - CORE_USER_L, 1235
 - CORE_WRITE_OK, 1252
 - delete_entry, 1239
 - delete_group, 1239
 - dirty, 1239
 - entries, 1239
 - entry, 1239
 - entry_exists, 1241
 - file_access, 1241
 - filename, 1241, 1243
 - FI_Preferences, 1235–1238
 - flush, 1243
 - get, 1243–1247
 - get_userdata_path, 1247
 - group, 1248
 - group_exists, 1248
 - groups, 1248
 - ID, 1234
 - MEMORY, 1235
 - new_UUID, 1248
 - NONE, 1252
 - Root, 1234
 - ROOT_MASK, 1235
 - set, 1249–1251
 - size, 1251
 - SYSTEM, 1235
 - SYSTEM_L, 1235
 - UNKNOWN_ROOT_TYPE, 1234
 - USER, 1235
 - USER_L, 1235
- FI_Preferences.H, 2145
- FI_Preferences::Entry, 365
- FI_Preferences::Name, 1980
 - Name, 1981
- FI_Preferences::Node, 1981
- FI_Preferences::RootNode, 1984
- FI_Printer, 1252
 - begin_job, 1257
 - begin_page, 1257
 - end_job, 1257
 - end_page, 1257
 - is_current, 1258
 - margins, 1258
 - origin, 1258
 - printable_rect, 1259
 - rotate, 1259
 - scale, 1259
 - set_current, 1259
 - translate, 1260
 - untranslate, 1260
- FI_Printer.H, 2148
- FI_Progress, 1260
 - draw, 1267
 - FI_Progress, 1267
 - maximum, 1267, 1268
 - minimum, 1268
 - value, 1268
- FI_Progress.H, 2149
- FL_PUSH
 - Enumerations.H, 2008
- fl_push_clip
 - Drawing functions, 302
- fl_push_matrix
 - Drawing functions, 303
- fl_putenv
 - Unicode and UTF-8 functions, 325
- FI_Radio_Button, 1268
 - FI_Radio_Button, 1276
- FI_Radio_Button.H, 2149
- FI_Radio_Light_Button, 1276
- FI_Radio_Light_Button.H, 2150
- FI_Radio_Round_Button, 1284
 - FI_Radio_Round_Button, 1292
- FI_Radio_Round_Button.H, 2150
- FL_READ
 - Enumerations.H, 2003
- fl_read_image
 - Drawing functions, 303
- FL_REASON_CANCELLED
 - Enumerations.H, 2006
- FL_REASON_CHANGED
 - Enumerations.H, 2006
- FL_REASON_CLOSED
 - Enumerations.H, 2006
- FL_REASON_DESELECTED
 - Enumerations.H, 2006
- FL_REASON_DRAGGED
 - Enumerations.H, 2006
- FL_REASON_ENTER_KEY
 - Enumerations.H, 2007
- FL_REASON_GOT_FOCUS

- Enumerations.H, [2006](#)
- FL_REASON_LOST_FOCUS
 - Enumerations.H, [2006](#)
- FL_REASON_OPENED
 - Enumerations.H, [2006](#)
- FL_REASON_RELEASED
 - Enumerations.H, [2006](#)
- FL_REASON_RESELECTED
 - Enumerations.H, [2006](#)
- FL_REASON_SELECTED
 - Enumerations.H, [2006](#)
- FL_REASON_UNKNOWN
 - Enumerations.H, [2006](#)
- FL_REASON_USER
 - Enumerations.H, [2007](#)
- Fl_Rect, [1292](#)
 - b, [1294](#)
 - Fl_Rect, [1294](#)
 - inset, [1294](#)
 - r, [1294](#)
- fl_rect
 - Drawing functions, [303](#), [304](#)
- fl_rect.cxx, [2296](#)
- Fl_Rect.H, [2151](#)
- fl_rectbound
 - fl_boxtype.cxx, [2281](#)
- fl_rectf
 - Drawing functions, [304](#), [305](#)
- Fl_Region
 - platform_types.h, [2265](#)
- fl_register_images
 - Fl_Shared_Image.H, [2159](#)
- FL_RELEASE
 - Enumerations.H, [2008](#)
- fl_rename
 - Unicode and UTF-8 functions, [325](#)
- Fl_Repeat_Button, [1296](#)
 - Fl_Repeat_Button, [1303](#)
 - handle, [1303](#)
- Fl_Repeat_Button.H, [2152](#)
- fl_rescale_offscreen
 - Drawing functions, [305](#)
- FL_RESERVED_TYPE
 - Fl_Widget.H, [2226](#)
- fl_reset_spot
 - Drawing functions, [305](#)
- fl_restore_scale
 - Drawing functions, [305](#)
- Fl_Return_Button, [1304](#)
 - draw, [1312](#)
 - Fl_Return_Button, [1312](#)
 - handle, [1312](#)
- Fl_Return_Button.H, [2152](#)
- Fl_RGB_Image, [1313](#)
 - array, [1320](#)
 - as_svg_image, [1317](#)
 - color_average, [1317](#)
 - copy, [1318](#)
 - desaturate, [1318](#)
 - draw, [1318](#)
 - Fl_RGB_Image, [1316](#), [1317](#)
 - label, [1319](#)
 - max_size, [1319](#)
 - normalize, [1319](#)
 - uncache, [1319](#)
- Fl_RGB_Image.H, [2153](#)
- Fl_RGB_Scaling
 - Fl_Image.H, [2113](#)
- FL_RGB_SCALING_BILINEAR
 - Fl_Image.H, [2113](#)
- FL_RGB_SCALING_NEAREST
 - Fl_Image.H, [2113](#)
- fl_rmdir
 - Unicode and UTF-8 functions, [325](#)
- Fl_Roller, [1320](#)
 - draw, [1328](#)
 - Fl_Roller, [1328](#)
 - handle, [1328](#)
- Fl_Roller.H, [2153](#)
- fl_rotate
 - Drawing functions, [305](#)
- Fl_Round_Button, [1329](#)
 - Fl_Round_Button, [1337](#)
- Fl_Round_Button.H, [2154](#)
- Fl_Round_Clock, [1337](#)
 - Fl_Round_Clock, [1345](#)
- Fl_Round_Clock.H, [2154](#)
- fl_rounded_rect
 - Drawing functions, [306](#)
- fl_rounded_rectf
 - Drawing functions, [306](#)
- fl_scale
 - Drawing functions, [306](#)
- Fl_Scheme, [1346](#)
 - add_scheme_name, [1346](#)
 - names, [1346](#)
 - num_schemes, [1347](#)
- Fl_Scheme.H, [2154](#)
- Fl_Scheme_Choice, [1347](#)
 - Fl_Scheme_Choice, [1357](#)
 - handle, [1357](#)
 - init_value, [1357](#)
 - scheme_cb_, [1357](#)
- Fl_Scheme_Choice.H, [2155](#)
- FL_SCREEN_CONFIGURATION_CHANGED
 - Enumerations.H, [2011](#)
- Fl_Screen_Driver.H, [2297](#)
- Fl_Scroll, [1358](#)
 - ~Fl_Scroll, [1368](#)
 - bbox, [1368](#)
 - delete_child, [1368](#)
 - draw, [1369](#)
 - fix_scrollbar_order, [1369](#)
 - Fl_Scroll, [1368](#)
 - handle, [1369](#)
 - on_insert, [1370](#)

- on_move, [1371](#)
- recalc_scrollbars, [1371](#)
- resize, [1371](#)
- scroll_to, [1372](#)
- scrollbar_size, [1372](#)
- xposition, [1373](#)
- yposition, [1373](#)
- fl_scroll
 - Drawing functions, [306](#)
- Fl_Scroll.H, [2156](#)
- Fl_Scroll::Fl_Region_LRTB, [1295](#)
- Fl_Scroll::Fl_Region_XYWH, [1295](#)
- Fl_Scroll::Fl_Scrollbar_Data, [1384](#)
- Fl_Scroll::ScrollInfo, [1984](#)
- Fl_Scrollbar, [1373](#)
 - draw, [1382](#)
 - Fl_Scrollbar, [1382](#)
 - handle, [1383](#)
 - linesize, [1383](#)
 - value, [1383](#), [1384](#)
- Fl_Scrollbar.H, [2157](#)
- Fl_Secret_Input, [1385](#)
 - Fl_Secret_Input, [1394](#)
 - handle, [1395](#)
- Fl_Secret_Input.H, [2158](#)
- Fl_Select_Browser, [1395](#)
 - Fl_Select_Browser, [1408](#)
- Fl_Select_Browser.H, [2158](#)
- FL_SELECTIONCLEAR
 - Enumerations.H, [2010](#)
- fl_set_spot
 - Drawing functions, [307](#)
- fl_set_status
 - Drawing functions, [307](#)
- Fl_Shared_Handler
 - Fl_Shared_Image.H, [2159](#)
- Fl_Shared_Image, [1409](#)
 - ~Fl_Shared_Image, [1413](#)
 - add, [1413](#)
 - add_handler, [1413](#)
 - as_shared_image, [1413](#)
 - color_average, [1413](#)
 - compare, [1414](#)
 - copy, [1414](#), [1415](#)
 - copy_, [1415](#)
 - desaturate, [1415](#)
 - draw, [1416](#)
 - find, [1416](#)
 - Fl_Shared_Image, [1412](#)
 - get, [1416](#), [1417](#)
 - image, [1417](#)
 - images, [1418](#)
 - num_images, [1418](#)
 - original, [1418](#)
 - refcount, [1418](#)
 - release, [1418](#)
 - uncache, [1419](#)
 - update, [1419](#)
- Fl_Shared_Image.H, [2158](#), [2160](#)
 - fl_register_images, [2159](#)
 - Fl_Shared_Handler, [2159](#)
- FL_SHORTCUT
 - Enumerations.H, [2010](#)
- Fl_Shortcut
 - fl_types.h, [2217](#)
- Fl_Shortcut_Button, [1419](#)
 - draw, [1427](#)
 - Fl_Shortcut_Button, [1427](#)
 - handle, [1427](#)
 - value, [1427](#), [1428](#)
- Fl_Shortcut_Button.H, [2161](#)
- fl_shortcut_label
 - Drawing functions, [307](#), [308](#)
- FL_SHOW
 - Enumerations.H, [2010](#)
- fl_show_colormap
 - Color & Font functions, [271](#)
- fl_show_colormap.H, [2162](#)
- fl_show_input.H, [2162](#)
- Fl_Simple_Counter, [1428](#)
- Fl_Simple_Counter.H, [2162](#)
- Fl_Single_Window, [1437](#)
 - flush, [1449](#)
 - show, [1449](#)
- Fl_Single_Window.H, [2163](#)
- fl_size
 - Color & Font functions, [272](#)
- Fl_Slider, [1450](#)
 - bounds, [1459](#)
 - draw, [1459](#)
 - Fl_Slider, [1458](#)
 - handle, [1459](#)
 - scrollvalue, [1459](#)
 - slider_size, [1460](#)
- Fl_Slider.H, [2163](#)
- FL_SOLID
 - Drawing functions, [281](#)
- Fl_Spinner, [1460](#)
 - draw, [1470](#)
 - Fl_Spinner, [1470](#)
 - handle, [1470](#)
 - resize, [1471](#)
 - step, [1471](#)
 - type, [1471](#), [1472](#)
 - value, [1472](#)
 - wrap, [1472](#)
- Fl_Spinner.H, [2164](#)
- Fl_Spinner::Fl_Spinner_Input, [1473](#)
 - handle, [1483](#)
- fl_stat
 - Unicode and UTF-8 functions, [326](#)
- fl_strdup
 - String handling functions, [334](#)
- Fl_String.H, [2299](#)
- fl_string_functions.h, [2166](#)
- FL_SUBMENU

- FI_Menu_Item.H, 2127
- FL_SUBMENU_POINTER
 - FI_Menu_Item.H, 2127
- FI_Surface_Device, 1483
 - end_current, 1484
 - is_current, 1484
 - pop_current, 1484
 - push_current, 1484
 - set_current, 1485
 - surface, 1485
- FI_SVG_File_Surface, 1485
 - ~FI_SVG_File_Surface, 1488
 - close, 1488
 - FI_SVG_File_Surface, 1488
 - origin, 1488
 - printable_rect, 1489
 - translate, 1489
 - untranslate, 1489
- FI_SVG_File_Surface.H, 2167
- FI_SVG_Image, 1489
 - as_svg_image, 1495
 - color_average, 1495
 - copy, 1495
 - desaturate, 1496
 - draw, 1496
 - FI_SVG_Image, 1494
 - normalize, 1496
 - proportional, 1496
 - resize, 1496
- FI_SVG_Image.H, 2167
- FL_SYMBOL_LABEL
 - Enumerations.H, 2002
- FI_Sys_Menu_Bar, 1497
 - about, 1508
 - add, 1508, 1509
 - clear, 1509
 - clear_submenu, 1509
 - create_window_menu, 1510
 - draw, 1510
 - FI_Sys_Menu_Bar, 1508
 - insert, 1510
 - menu, 1511
 - mode, 1511
 - no_window_menu, 1508
 - play_menu, 1511
 - remove, 1511
 - replace, 1511
 - tabbing_mode_automatic, 1508
 - tabbing_mode_none, 1508
 - tabbing_mode_preferred, 1508
 - update, 1512
 - window_menu_style, 1512
 - window_menu_style_enum, 1508
- FI_Sys_Menu_Bar.H, 2168
- FI_Sys_Menu_Bar_Driver.H, 2301
- fl_system
 - Unicode and UTF-8 functions, 326
- FI_System_Driver.H, 2302
- FI_Table, 1512
 - ~FI_Table, 1528
 - array, 1528
 - callback, 1528
 - callback_col, 1529
 - callback_context, 1529
 - callback_row, 1529
 - child, 1529
 - children, 1530
 - clear, 1530
 - col_header, 1530
 - col_resize, 1530
 - col_resize_min, 1530
 - col_width, 1530
 - col_width_all, 1530
 - CONTEXT_CELL, 1528
 - CONTEXT_COL_HEADER, 1528
 - CONTEXT_ENDPAGE, 1528
 - CONTEXT_NONE, 1528
 - CONTEXT_RC_RESIZE, 1528
 - CONTEXT_ROW_HEADER, 1528
 - CONTEXT_STARTPAGE, 1528
 - CONTEXT_TABLE, 1528
 - cursor2rowcol, 1531
 - damage_zone, 1531
 - do_callback, 1531
 - draw, 1531
 - draw_cell, 1531
 - find_cell, 1533
 - FI_Table, 1528
 - get_selection, 1533
 - handle, 1533
 - init_sizes, 1533
 - insert, 1534
 - is_interactive_resize, 1534
 - is_selected, 1534
 - move_cursor, 1534
 - recalc_dimensions, 1534
 - redraw_range, 1534
 - resize, 1535
 - row_col_clamp, 1535
 - row_header, 1535
 - row_height, 1535
 - row_height_all, 1535
 - row_resize, 1535
 - row_resize_min, 1535
 - rows, 1536
 - scrollbar_size, 1536
 - set_selection, 1536
 - tab_cell_nav, 1537
 - table_box, 1537
 - table_resized, 1537
 - table_scrolled, 1537
 - TableContext, 1528
 - top_row, 1537, 1538
 - visible_cells, 1538
 - when, 1538
- FI_Table.H, 2169

- FI_Table_Row, 1539
 - ~FI_Table_Row, 1552
 - clear, 1553
 - FI_Table_Row, 1552
 - handle, 1553
 - row_selected, 1553
 - rows, 1553
 - select_all_rows, 1553
 - select_row, 1554
 - type, 1554
- FI_Table_Row.H, 2175
- FI_Tabs, 1555
 - clear_tab_positions, 1568
 - client_area, 1568
 - draw, 1569
 - draw_tab, 1569
 - FI_Tabs, 1568
 - handle, 1569
 - handle_overflow, 1570
 - handle_overflow_menu, 1570
 - hit_close, 1570
 - hit_overflow_menu, 1571
 - hit_tabs_area, 1571
 - maybe_do_callback, 1571
 - on_insert, 1571
 - on_move, 1572
 - on_remove, 1572
 - OVERFLOW_CLIP, 1568
 - OVERFLOW_COMPRESS, 1568
 - OVERFLOW_DRAG, 1568
 - OVERFLOW_PULLDOWN, 1568
 - overflow_type, 1576
 - push, 1572
 - redraw_tabs, 1572
 - resize, 1572
 - show, 1573
 - tab_align, 1573
 - tab_count, 1576
 - tab_flags, 1576
 - tab_height, 1573
 - tab_pos, 1576
 - tab_positions, 1573
 - tab_width, 1577
 - take_focus, 1574
 - value, 1574
 - which, 1576
- FI_Tabs.H, 2177
- FI_Terminal, 1577
 - _RESERVED_1, 1598
 - _RESERVED_2, 1598
 - ~FI_Terminal, 1600
 - ansi, 1600
 - append, 1601
 - append_ascii, 1601
 - append_utf8, 1601
 - Attrib, 1598
 - BG_XTERM, 1598
 - BOLD, 1598
 - box, 1602
 - CharFlags, 1598
 - clear, 1602
 - clear_screen, 1602
 - clear_screen_home, 1602
 - color, 1603
 - CR_TO_LF, 1599
 - cursor_col, 1603
 - cursor_cr, 1603
 - cursor_down, 1603
 - cursor_right, 1603
 - cursor_row, 1604
 - cursor_up, 1604
 - delete_rows, 1604
 - DIM, 1598
 - display_columns, 1604
 - display_rows, 1604
 - draw, 1604
 - draw_buff, 1605
 - draw_row, 1605
 - draw_row_bg, 1605
 - EOL, 1598
 - error_char, 1605, 1606
 - FG_XTERM, 1598
 - FI_Terminal, 1599, 1600
 - get_selection, 1606
 - h_to_row, 1606
 - handle, 1606
 - handle_unknown_char, 1607
 - history_lines, 1607
 - history_use, 1607
 - hscrollbar, 1619
 - hscrollbar_style, 1607, 1608
 - insert_char, 1608
 - insert_rows, 1608
 - INVERSE, 1598
 - is_inside_selection, 1608
 - ITALIC, 1598
 - LF_TO_CR, 1599
 - LF_TO_CRLF, 1599
 - NO_REDRAW, 1599
 - NORMAL, 1598
 - OFF, 1599
 - OutFlags, 1598
 - output_translate, 1608
 - PER_WRITE, 1599
 - plot_char, 1609
 - print_char, 1609, 1610
 - printf, 1610
 - RATE_LIMITED, 1599
 - redraw_rate, 1610
 - redraw_style, 1611
 - RedrawStyle, 1599
 - reset_terminal, 1611
 - resize, 1611
 - scroll, 1612
 - scrollbar, 1619
 - scrollbar_actual_size, 1612

- SCROLLBAR_AUTO, 1599
- SCROLLBAR_OFF, 1599
- SCROLLBAR_ON, 1599
- scrollbar_size, 1612
- ScrollbarStyle, 1599
- selection_extend, 1613
- selection_text, 1613
- selection_text_len, 1613
- show_unknown, 1613
- STRIKEOUT, 1598
- text, 1613
- textattrib, 1614
- textbgcolor, 1614
- textbgcolor_default, 1614, 1615
- textbgcolor_xterm, 1615
- textcolor, 1615
- textfgcolor, 1616
- textfgcolor_default, 1616
- textfgcolor_xterm, 1616
- textfont, 1617
- textsize, 1617
- u8c_disp_row, 1617
- u8c_hist_row, 1618
- u8c_hist_use_row, 1618
- u8c_ring_row, 1618
- UNDERLINE, 1598
- vprintf, 1618
- w_to_col, 1619
- walk_selection, 1619
- Fl_Terminal Technical Documentation, 167
- Fl_Terminal.H, 2178
- Fl_Terminal::CharStyle, 364
- Fl_Terminal::Cursor, 365
- Fl_Terminal::EscapeSeq, 366
- Fl_Terminal::Margin, 1980
- Fl_Terminal::PartialUtf8Buf, 1983
- Fl_Terminal::RingBuffer, 1983
- Fl_Terminal::Selection, 1985
 - get_selection, 1985
- Fl_Terminal::Utf8Char, 1986
- Fl_Text_Buffer, 1620
 - add_modify_callback, 1625
 - address, 1626
 - append, 1626
 - appendfile, 1626
 - byte_at, 1626
 - can_redo, 1627
 - can_undo, 1627
 - canUndo, 1627
 - char_at, 1627
 - copy, 1627
 - count_displayed_characters, 1628
 - count_lines, 1628
 - estimate_lines, 1628
 - file_encoding_warning_message, 1637
 - findchar_backward, 1628
 - findchar_forward, 1628
 - Fl_Text_Buffer, 1625
 - highlight_text, 1629
 - insert, 1629
 - insert_, 1629
 - insertfile, 1629
 - is_word_separator, 1630
 - length, 1630
 - line_end, 1630
 - line_start, 1630
 - line_text, 1631
 - loadfile, 1631
 - mTabDist, 1637
 - next_char, 1631
 - outputfile, 1631
 - prev_char, 1632
 - printf, 1632
 - remove, 1632
 - remove_, 1632
 - replace, 1632
 - rewind_lines, 1633
 - savefile, 1633
 - search_backward, 1633
 - search_forward, 1634
 - secondary_selection_text, 1634
 - selection_text, 1634
 - skip_displayed_characters, 1634
 - tab_distance, 1635
 - text, 1635
 - text_range, 1635
 - transcoding_warning_action, 1637
 - undo, 1635
 - vprintf, 1636
 - word_end, 1636
 - word_start, 1636
- Fl_Text_Buffer.H, 2188
- Fl_Text_Display, 1637
 - ~Fl_Text_Display, 1655
 - absolute_top_line_number, 1655
 - ATTR_BGCOLOR, 1654
 - ATTR_BGCOLOR_EXT, 1654
 - ATTR_BGCOLOR_EXT_, 1654
 - ATTR_GRAMMAR, 1654
 - ATTR_LINES_MASK, 1655
 - ATTR_SPELLING, 1655
 - ATTR_STRIKE_THROUGH, 1655
 - ATTR_UNDERLINE, 1654
 - BLOCK_CURSOR, 1654
 - buffer, 1655, 1656
 - buffer_modified_cb, 1656
 - buffer_predelete_cb, 1656
 - calc_last_char, 1657
 - calc_line_starts, 1657
 - CARET_CURSOR, 1654
 - clear_rect, 1657
 - col_to_x, 1657
 - count_lines, 1658
 - cursor_color, 1658
 - cursor_style, 1658
 - DIM_CURSOR, 1654

display_insert, 1659
display_needs_recalc, 1659
draw, 1659
draw_cursor, 1659
draw_line_numbers, 1659
draw_range, 1661
draw_string, 1661
draw_text, 1661
draw_vline, 1662
empty_vlines, 1662
extend_range_for_styles, 1662
find_line_end, 1662
find_wrap_range, 1663
find_x, 1663
FI_Text_Display, 1655
get_absolute_top_line_number, 1664
grammar_underline_color, 1664
handle, 1664
handle_rmb, 1664
handle_vline, 1664
HEAVY_CURSOR, 1654
highlight_data, 1665
in_selection, 1666
insert, 1666
insert_position, 1666, 1667
line_end, 1667
line_start, 1667
linenumber_align, 1668
linenumber_bgcolor, 1668
linenumber_fgcolor, 1668
linenumber_font, 1668
linenumber_format, 1668
linenumber_size, 1669
linenumber_width, 1669
longest_vline, 1669
maintain_absolute_top_line_number, 1669
maintaining_absolute_top_line_number, 1669
measure_deleted_lines, 1669
measure_proportional_character, 1670
measure_vline, 1670
move_down, 1670
move_left, 1671
move_right, 1671
move_up, 1671
NORMAL_CURSOR, 1654
offset_line_starts, 1671
overstrike, 1671
position_style, 1671
position_to_line, 1672
position_to_linecol, 1672
position_to_xy, 1673
recalc_display, 1673
redisplay_range, 1673
reset_absolute_top_line_number, 1674
resize, 1674
rewind_lines, 1674
scroll, 1674
scroll_, 1674
scroll_timer_cb, 1675
scrollbar_align, 1675
scrollbar_size, 1675
scrollbar_width, 1676
secondary_selection_color, 1676
shortcut, 1676, 1677
show_cursor, 1677
show_insert_position, 1677
SIMPLE_CURSOR, 1654
skip_lines, 1677
spelling_underline_color, 1678
string_width, 1678
style_buffer, 1678
textcolor, 1678, 1679
textfont, 1679
textsize, 1679
update_h_scrollbar, 1679
update_line_starts, 1680
update_v_scrollbar, 1680
vline_length, 1680
word_end, 1680
word_start, 1681
WRAP_AT_BOUNDS, 1654
WRAP_AT_COLUMN, 1654
WRAP_AT_PIXEL, 1654
wrap_mode, 1681
WRAP_NONE, 1654
wrap_uses_character, 1681
wrapped_column, 1682
wrapped_line_counter, 1682
wrapped_row, 1684
x_to_col, 1684
xy_to_position, 1684
xy_to_rowcol, 1685
FI_Text_Display.H, 2192
FI_Text_Display::Style_Table_Entry, 1986
FI_Text_Editor, 1685
 add_key_binding, 1703
 global_key_bindings, 1708
 handle, 1703
 insert_mode, 1703
 kf_backspace, 1703
 kf_c_s_move, 1703
 kf_copy, 1703
 kf_ctrl_move, 1704
 kf_cut, 1704
 kf_default, 1704
 kf_delete, 1704
 kf_down, 1704
 kf_end, 1704
 kf_enter, 1704
 kf_home, 1705
 kf_ignore, 1705
 kf_insert, 1705
 kf_left, 1705
 kf_m_s_move, 1705
 kf_meta_move, 1705
 kf_move, 1706

- kf_page_down, 1706
 - kf_page_up, 1706
 - kf_paste, 1706
 - kf_redo, 1706
 - kf_right, 1706
 - kf_select_all, 1706
 - kf_shift_move, 1707
 - kf_undo, 1707
 - kf_up, 1707
 - remove_key_binding, 1707
 - tab_nav, 1707, 1708
- Fl_Text_Editor.H, 2197
- Fl_Text_Editor::Key_Binding, 1980
- fl_text_extents
 - Color & Font functions, 272
- Fl_Text_Selection, 1708
 - end, 1710
 - includes, 1710
 - length, 1710
 - position, 1710
 - selected, 1710, 1711
 - set, 1711
 - start, 1711
 - update, 1712
- FL_THIN_DOWN_BOX
 - Enumerations.H, 2004
- FL_THIN_DOWN_FRAME
 - Enumerations.H, 2004
- FL_THIN_UP_BOX
 - Enumerations.H, 2004
- FL_THIN_UP_FRAME
 - Enumerations.H, 2004
- Fl_Tile, 1712
 - cursor, 1723
 - drag_intersection, 1723
 - Fl_Tile, 1723
 - handle, 1724
 - init_size_range, 1724
 - move_intersection, 1725
 - on_insert, 1725
 - on_move, 1725
 - on_remove, 1725
 - position, 1725
 - request_grow_b, 1725
 - request_grow_l, 1726
 - request_grow_r, 1726
 - request_grow_t, 1726
 - request_shrink_b, 1727
 - request_shrink_l, 1727
 - request_shrink_r, 1727
 - request_shrink_t, 1728
 - resize, 1728
 - set_cursor, 1728
 - size_range, 1728, 1729
- Fl_Tile.H, 2198
- Fl_Tile::Size_Range, 1986
- Fl_Tiled_Image, 1729
 - color_average, 1732
 - copy, 1732
 - desaturate, 1733
 - draw, 1733
 - Fl_Tiled_Image, 1732
- Fl_Tiled_Image.H, 2199
- Fl_Timeout, 1734
 - add_timeout, 1735
 - current, 1736
 - current_timeout, 1740
 - elapse_timeouts, 1736
 - first_timeout, 1740
 - free_timeout, 1740
 - get, 1736
 - has_timeout, 1737
 - insert, 1737
 - make_current, 1737
 - release, 1738
 - remove_next_timeout, 1738
 - remove_timeout, 1738
 - repeat_timeout, 1739
 - time_to_wait, 1739
- Fl_Timeout.cxx, 2304
- Fl_Timeout.h, 2304, 2305
- Fl_Timeout_Handler
 - Callback Function Typedefs, 232
- Fl_Timer, 1740
 - direction, 1747, 1748
 - draw, 1748
 - Fl_Timer, 1747
 - handle, 1748
 - suspended, 1748
- Fl_Timer.H, 2200
- Fl_Timestamp
 - platform_types.h, 2266
- Fl_Toggle_Button, 1749
 - Fl_Toggle_Button, 1756
- Fl_Toggle_Button.H, 2200
- Fl_Toggle_Light_Button.H, 2201
- Fl_Toggle_Round_Button.H, 2201
- Fl_Tooltip, 1757
 - color, 1758
 - current, 1759
 - delay, 1759
 - disable, 1759
 - enable, 1759
 - enabled, 1759
 - enter_area, 1759
 - font, 1760
 - hidedelay, 1760
 - hoverdelay, 1760
 - margin_height, 1760
 - margin_width, 1761
 - size, 1761
 - textcolor, 1761
 - wrap_width, 1761
- Fl_Tooltip.H, 2201
- fl_transform_dx
 - Drawing functions, 309

- fl_transform_dy
 - Drawing functions, 309
- fl_transform_x
 - Drawing functions, 309
- fl_transform_y
 - Drawing functions, 309
- fl_transformed_vertex
 - Drawing functions, 309
- fl_translate
 - Drawing functions, 311
- FI_Tree, 1762
 - add, 1780, 1781
 - calc_dimensions, 1781
 - calc_tree, 1781
 - callback_item, 1782
 - callback_reason, 1782
 - clear, 1783
 - clear_children, 1783
 - close, 1783
 - closeicon, 1784
 - connectorstyle, 1784
 - deselect, 1784, 1785
 - deselect_all, 1785
 - display, 1786
 - displayed, 1786
 - draw, 1786
 - extend_selection, 1786
 - extend_selection_dir, 1787
 - find_clicked, 1787
 - find_item, 1788
 - first, 1788
 - first_selected_item, 1789
 - first_visible, 1789
 - first_visible_item, 1789
 - get_selected_items, 1789
 - handle, 1790
 - hposition, 1790
 - insert, 1791
 - insert_above, 1791
 - is_close, 1792
 - is_hscroll_visible, 1792
 - is_open, 1793
 - is_scrollbar, 1793
 - is_selected, 1794
 - is_vscroll_visible, 1794
 - item_clicked, 1795
 - item_draw_mode, 1795
 - item_labelbgcolor, 1795, 1796
 - item_labelfgcolor, 1796
 - item_labelfont, 1796
 - item_labelsize, 1796
 - item_pathname, 1796
 - item_reselect_mode, 1797
 - last, 1797
 - last_selected_item, 1797
 - last_visible, 1798
 - last_visible_item, 1798
 - load, 1798
 - next, 1798
 - next_item, 1799
 - next_selected_item, 1800
 - next_visible_item, 1800
 - open, 1801
 - open_toggle, 1802
 - openicon, 1802
 - prev, 1803
 - recalc_tree, 1803
 - remove, 1803
 - resize, 1803
 - root, 1804
 - root_label, 1804
 - scrollbar_size, 1804
 - select, 1804, 1805
 - select_all, 1805
 - select_only, 1806
 - select_toggle, 1806
 - selectbox, 1807
 - selectmode, 1807
 - set_item_focus, 1807
 - show_item, 1807, 1808
 - show_item_bottom, 1808
 - show_item_middle, 1808
 - show_item_top, 1808
 - show_self, 1809
 - showcollapse, 1809
 - showroot, 1809
 - sortorder, 1809
 - usericon, 1810
 - vposition, 1810
- FI_Tree.H, 2202, 2203
 - FI_Tree_Reason, 2203
 - FL_TREE_REASON_CLOSED, 2203
 - FL_TREE_REASON_DESELECTED, 2203
 - FL_TREE_REASON_DRAGGED, 2203
 - FL_TREE_REASON_NONE, 2203
 - FL_TREE_REASON_OPENED, 2203
 - FL_TREE_REASON_RESELECTED, 2203
 - FL_TREE_REASON_SELECTED, 2203
- FI_Tree_Connector
 - FI_Tree_Prefs.H, 2212
- FL_TREE_CONNECTOR_DOTTED
 - FI_Tree_Prefs.H, 2212
- FL_TREE_CONNECTOR_NONE
 - FI_Tree_Prefs.H, 2212
- FL_TREE_CONNECTOR_SOLID
 - FI_Tree_Prefs.H, 2212
- FI_Tree_Item, 1810
 - activate, 1816
 - add, 1816, 1817
 - calc_item_height, 1817
 - child, 1818
 - deactivate, 1818
 - deparent, 1818
 - depth, 1818
 - deselect_all, 1818
 - draw, 1818

- draw_horizontal_connector, 1819
- draw_item_content, 1819
- draw_vertical_connector, 1820
- drawbgcolor, 1820
- drawfgcolor, 1821
- find_child, 1821
- find_child_item, 1821
- find_clicked, 1822
- find_item, 1822
- Fl_Tree_Item, 1816
- hide_widgets, 1822
- insert, 1822
- insert_above, 1823
- is_visible_r, 1823
- label, 1823
- label_h, 1823
- label_w, 1823
- label_x, 1823
- label_y, 1824
- labelbgcolor, 1824
- move, 1824
- move_above, 1825
- move_below, 1825
- move_into, 1825
- next, 1826
- next_displayed, 1826
- next_sibling, 1826
- next_visible, 1826
- parent, 1826
- prefs, 1826
- prev, 1827
- prev_displayed, 1827
- prev_sibling, 1827
- prev_visible, 1827
- recalc_tree, 1827
- remove_child, 1827, 1828
- reparent, 1828
- replace, 1828
- replace_child, 1829
- select, 1829
- select_all, 1829
- show_self, 1829
- show_widgets, 1829
- swap_children, 1830
- tree, 1830
- update_prev_next, 1830
- userdeicon, 1831
- usericon, 1831
- visible_r, 1831
- Fl_Tree_Item.H, 2206
- Fl_Tree_Item_Array, 1832
 - add, 1833
 - clear, 1833
 - deparent, 1833
 - Fl_Tree_Item_Array, 1833
 - insert, 1833
 - manage_item_destroy, 1833
 - move, 1834
 - remove, 1834
 - reparent, 1834
 - replace, 1834
- Fl_Tree_Item_Array.H, 2210
- FL_TREE_ITEM_DRAW_DEFAULT
 - Fl_Tree_Prefs.H, 2212
- FL_TREE_ITEM_DRAW_LABEL_AND_WIDGET
 - Fl_Tree_Prefs.H, 2212
- Fl_Tree_Item_Draw_Mode
 - Fl_Tree_Prefs.H, 2212
- FL_TREE_ITEM_HEIGHT_FROM_WIDGET
 - Fl_Tree_Prefs.H, 2212
- Fl_Tree_Item_Reselect_Mode
 - Fl_Tree_Prefs.H, 2212
- Fl_Tree_Prefs, 1835
 - closedeicon, 1838
 - closeicon, 1838
 - item_draw_mode, 1838
 - item_labelbgcolor, 1838
 - marginbottom, 1838
 - opendeicon, 1838
 - openicon, 1838, 1839
 - selectmode, 1839
 - showcollapse, 1839
 - showroot, 1839
 - sortorder, 1839
 - userdeicon, 1839
- Fl_Tree_Prefs.H, 2211, 2213
 - Fl_Tree_Connector, 2212
 - FL_TREE_CONNECTOR_DOTTED, 2212
 - FL_TREE_CONNECTOR_NONE, 2212
 - FL_TREE_CONNECTOR_SOLID, 2212
 - FL_TREE_ITEM_DRAW_DEFAULT, 2212
 - FL_TREE_ITEM_DRAW_LABEL_AND_WIDGET, 2212
 - Fl_Tree_Item_Draw_Mode, 2212
 - FL_TREE_ITEM_HEIGHT_FROM_WIDGET, 2212
 - Fl_Tree_Item_Reselect_Mode, 2212
 - Fl_Tree_Select, 2213
 - FL_TREE_SELECT_MULTI, 2213
 - FL_TREE_SELECT_NONE, 2213
 - FL_TREE_SELECT_SINGLE, 2213
 - FL_TREE_SELECT_SINGLE_DRAGGABLE, 2213
 - FL_TREE_SELECTABLE_ALWAYS, 2213
 - FL_TREE_SELECTABLE_ONCE, 2213
 - Fl_Tree_Sort, 2213
 - FL_TREE_SORT_ASCENDING, 2213
 - FL_TREE_SORT_DESCENDING, 2213
 - FL_TREE_SORT_NONE, 2213
- Fl_Tree_Reason
 - Fl_Tree.H, 2203
- FL_TREE_REASON_CLOSED
 - Fl_Tree.H, 2203
- FL_TREE_REASON_DESELECTED
 - Fl_Tree.H, 2203
- FL_TREE_REASON_DRAGGED
 - Fl_Tree.H, 2203

- FL_TREE_REASON_NONE
 - FI_Tree.H, [2203](#)
- FL_TREE_REASON_OPENED
 - FI_Tree.H, [2203](#)
- FL_TREE_REASON_RESELECTED
 - FI_Tree.H, [2203](#)
- FL_TREE_REASON_SELECTED
 - FI_Tree.H, [2203](#)
- FI_Tree_Select
 - FI_Tree_Prefs.H, [2213](#)
- FL_TREE_SELECT_MULTI
 - FI_Tree_Prefs.H, [2213](#)
- FL_TREE_SELECT_NONE
 - FI_Tree_Prefs.H, [2213](#)
- FL_TREE_SELECT_SINGLE
 - FI_Tree_Prefs.H, [2213](#)
- FL_TREE_SELECT_SINGLE_DRAGGABLE
 - FI_Tree_Prefs.H, [2213](#)
- FL_TREE_SELECTABLE_ALWAYS
 - FI_Tree_Prefs.H, [2213](#)
- FL_TREE_SELECTABLE_ONCE
 - FI_Tree_Prefs.H, [2213](#)
- FI_Tree_Sort
 - FI_Tree_Prefs.H, [2213](#)
- FL_TREE_SORT_ASCENDING
 - FI_Tree_Prefs.H, [2213](#)
- FL_TREE_SORT_DESCENDING
 - FI_Tree_Prefs.H, [2213](#)
- FL_TREE_SORT_NONE
 - FI_Tree_Prefs.H, [2213](#)
- fl_types.h, [2217](#)
 - FI_Shortcut, [2217](#)
- fl_ucs_to_Utf16
 - Unicode and UTF-8 functions, [326](#)
- fl_uintptr_t
 - platform_types.h, [2266](#)
- FL_UNFOCUS
 - Enumerations.H, [2009](#)
- fl_unlink
 - Unicode and UTF-8 functions, [327](#)
- FL_UP_BOX
 - Enumerations.H, [2004](#)
- FL_UP_FRAME
 - Enumerations.H, [2004](#)
- fl_utf8.h, [2218](#), [2220](#)
- fl_utf8back
 - Unicode and UTF-8 functions, [327](#)
- fl_utf8bytes
 - Unicode and UTF-8 functions, [327](#)
- fl_utf8decode
 - Unicode and UTF-8 functions, [328](#)
- fl_utf8encode
 - Unicode and UTF-8 functions, [328](#)
- fl_utf8from_mb
 - Unicode and UTF-8 functions, [328](#)
- fl_utf8froma
 - Unicode and UTF-8 functions, [329](#)
- fl_utf8fromwc
 - Unicode and UTF-8 functions, [329](#)
- fl_utf8fwd
 - Unicode and UTF-8 functions, [329](#)
- fl_utf8len
 - Unicode and UTF-8 functions, [329](#)
- fl_utf8len1
 - Unicode and UTF-8 functions, [330](#)
- fl_utf8locale
 - Unicode and UTF-8 functions, [330](#)
- fl_utf8strlen
 - Unicode and UTF-8 functions, [330](#)
- fl_utf8test
 - Unicode and UTF-8 functions, [330](#)
- fl_utf8to_mb
 - Unicode and UTF-8 functions, [331](#)
- fl_utf8toa
 - Unicode and UTF-8 functions, [331](#)
- fl_utf8toUtf16
 - Unicode and UTF-8 functions, [331](#)
- fl_utf8towc
 - Unicode and UTF-8 functions, [332](#)
- fl_utf_nb_char
 - Unicode and UTF-8 functions, [332](#)
- fl_utf_strcasecmp
 - Unicode and UTF-8 functions, [332](#)
- fl_utf_strncasecmp
 - Unicode and UTF-8 functions, [333](#)
- fl_utf_tolower
 - Unicode and UTF-8 functions, [333](#)
- fl_utf_toupper
 - Unicode and UTF-8 functions, [333](#)
- FI_Valuator, [1840](#)
 - FI_Valuator, [1848](#)
 - format, [1848](#)
 - increment, [1849](#)
 - maximum, [1849](#)
 - minimum, [1849](#)
 - precision, [1849](#)
 - range, [1849](#)
 - round, [1850](#)
 - step, [1850](#)
 - value, [1850](#)
 - value_damage, [1850](#)
- FI_Valuator.H, [2222](#)
- FI_Value_Input, [1850](#)
 - cursor_color, [1859](#), [1860](#)
 - draw, [1860](#)
 - FI_Value_Input, [1859](#)
 - handle, [1860](#)
 - resize, [1860](#)
 - shortcut, [1861](#)
 - soft, [1861](#)
 - textcolor, [1861](#)
 - textfont, [1861](#), [1862](#)
 - textsize, [1862](#)
- FI_Value_Input.H, [2223](#)
- FI_Value_Output, [1862](#)
 - draw, [1871](#)

- FL_Value_Output, 1870
 - handle, 1871
 - soft, 1871
 - textcolor, 1872
 - textfont, 1872
 - textsize, 1872
- FL_Value_Output.H, 2224
- FL_Value_Slider, 1872
 - draw, 1882
 - FL_Value_Slider, 1882
 - handle, 1882
 - value_height, 1882, 1883
 - value_width, 1883
- FL_Value_Slider.H, 2225
- FL_VERSION
 - Enumerations.H, 2002
- fl_vertex
 - Drawing functions, 311
- fl_vertex.cxx, 2306
- fl_vsnprintf
 - vsnprintf.c, 2318
- fl_wcwidth
 - Unicode and UTF-8 functions, 333
- fl_wcwidth_
 - Unicode and UTF-8 functions, 334
- FL_When
 - Enumerations.H, 2012
- FL_WHEN_CHANGED
 - Enumerations.H, 2012
- FL_WHEN_CLOSED
 - Enumerations.H, 2012
- FL_WHEN_ENTER_KEY
 - Enumerations.H, 2012
- FL_WHEN_ENTER_KEY_ALWAYS
 - Enumerations.H, 2012
- FL_WHEN_ENTER_KEY_CHANGED
 - Enumerations.H, 2012
- FL_WHEN_NEVER
 - Enumerations.H, 2012
- FL_WHEN_NOT_CHANGED
 - Enumerations.H, 2012
- FL_WHEN_RELEASE
 - Enumerations.H, 2012
- FL_WHEN_RELEASE_ALWAYS
 - Enumerations.H, 2012
- Fl_Widget, 1884
 - ~Fl_Widget, 1892
 - activate, 1892
 - active, 1892
 - active_r, 1892
 - align, 1893
 - argument, 1893
 - as_gl_window, 1894
 - as_group, 1894
 - as_window, 1894
 - AUTO_DELETE_USER_DATA, 1891
 - bind_deimage, 1895
 - bind_image, 1895, 1897
 - box, 1897
 - callback, 1897, 1898
 - CHANGED, 1891
 - changed, 1899
 - clear_active, 1899
 - clear_changed, 1899
 - clear_damage, 1899
 - clear_output, 1900
 - clear_visible, 1900
 - clear_visible_focus, 1900
 - CLIP_CHILDREN, 1891
 - color, 1900, 1901
 - color2, 1901
 - contains, 1901
 - COPIED_LABEL, 1891
 - COPIED_TOOLTIP, 1891
 - copy_label, 1902
 - copy_tooltip, 1902
 - damage, 1902, 1904
 - deactivate, 1904
 - default_callback, 1904
 - deimage, 1905
 - DEIMAGE_BOUND, 1891
 - deimage_bound, 1905
 - do_callback, 1906
 - draw, 1907
 - draw_focus, 1907, 1908
 - draw_label, 1908
 - Fl_Widget, 1892
 - FORCE_POSITION, 1891
 - FULLSCREEN, 1891
 - GROUP_RELATIVE, 1891
 - h, 1909
 - handle, 1909
 - hide, 1910
 - horizontal_label_margin, 1910
 - image, 1910, 1911
 - IMAGE_BOUND, 1891
 - image_bound, 1911
 - INACTIVE, 1891
 - inside, 1911
 - INVISIBLE, 1891
 - is_label_copied, 1911
 - label, 1912
 - label_image_spacing, 1912
 - label_shortcut, 1913
 - labelcolor, 1913
 - labelfont, 1913, 1914
 - labelsize, 1914
 - labeltype, 1914, 1915
 - MAC_USE_ACCENTS_MENU, 1891
 - MAXIMIZED, 1891
 - measure_label, 1915
 - MENU_WINDOW, 1891
 - MODAL, 1891
 - NEEDS_KEYBOARD, 1891
 - needs_keyboard, 1915
 - NO_OVERLAY, 1891

- NOBORDER, 1891
- NON_MODAL, 1891
- OUTPUT, 1891
- output, 1916
- OVERRIDE, 1891
- parent, 1916
- POPUP, 1891
- position, 1916
- redraw, 1918
- redraw_label, 1918
- resize, 1918
- selection_color, 1918
- set_active, 1919
- set_changed, 1919
- set_output, 1919
- set_visible, 1919
- set_visible_focus, 1919
- SHORTCUT_LABEL, 1891
- shortcut_label, 1919, 1920
- show, 1920
- size, 1920
- take_focus, 1920
- takeevents, 1921
- test_shortcut, 1921
- tooltip, 1922
- TOOLTIP_WINDOW, 1891
- top_window, 1922
- top_window_offset, 1922
- type, 1923
- user_data, 1923
- USERFLAG1, 1891
- USERFLAG2, 1891
- USERFLAG3, 1891
- vertical_label_margin, 1923
- visible, 1923
- VISIBLE_FOCUS, 1891
- visible_focus, 1924
- visible_r, 1924
- w, 1924
- when, 1925
- window, 1925
- x, 1926
- y, 1926
- Fl_Widget.H, 2225, 2226
 - FL_RESERVED_TYPE, 2226
- Fl_Widget_Surface, 1927
 - draw, 1928
 - draw_decorated_window, 1929
 - Fl_Widget_Surface, 1928
 - origin, 1929
 - print_window_part, 1929
 - printable_rect, 1930
 - translate, 1930
 - untranslate, 1930
- Fl_Widget_Surface.H, 2231
- Fl_Widget_Tracker, 1930
 - deleted, 1931
 - exists, 1931
 - widget, 1931
- fl_width
 - Color & Font functions, 272
- Fl_Window, 1932
 - ~Fl_Window, 1945
 - allow_expand_outside_parent, 1945
 - as_double_window, 1945
 - as_overlay_window, 1945
 - as_window, 1945
 - border, 1946
 - clear_border, 1946
 - clear_modal_states, 1946
 - current, 1946
 - current_, 1963
 - cursor, 1947
 - decorated_h, 1947
 - decorated_w, 1948
 - default_cursor, 1948
 - default_icon, 1948
 - default_icons, 1948, 1949
 - default_size_range, 1949
 - default_xclass, 1950
 - draw, 1951
 - Fl_Window, 1944
 - flush, 1951
 - force_position, 1951
 - free_icons, 1951
 - free_position, 1952
 - fullscreen, 1952
 - fullscreen_screens, 1952
 - get_size_range, 1952
 - handle, 1953
 - hide, 1953
 - hotspot, 1953
 - icon, 1954
 - iconize, 1954
 - icons, 1955
 - is_a_rescale, 1955
 - is_resizable, 1955
 - make_current, 1956
 - maximize, 1956
 - modal, 1956
 - os_id, 1956
 - resize, 1957
 - screen_num, 1957
 - set_menu_window, 1958
 - set_modal, 1958
 - set_non_modal, 1958
 - set_tooltip_window, 1958
 - shape, 1959
 - show, 1960
 - show_next_window_iconic, 1961
 - shown, 1961
 - size_range, 1961
 - un_maximize, 1962
 - wait_for_expose, 1962
 - xclass, 1963
- Fl_Window.H, 2232

- Fl_Window_Driver.H, [2306](#)
- Fl_Wizard, [1964](#)
 - draw, [1972](#)
 - Fl_Wizard, [1972](#)
 - next, [1972](#)
- Fl_Wizard.H, [2235](#)
- fl_wl_compositor
 - wayland.H, [2268](#)
- FL_WRITE
 - Enumerations.H, [2003](#)
- fl_write_png
 - fl_write_png.cxx, [2309](#), [2310](#)
- fl_write_png.cxx, [2309](#)
 - fl_write_png, [2309](#), [2310](#)
- fl_x11_find
 - x11.H, [2271](#)
- fl_x11_gc
 - x11.H, [2271](#)
- fl_x11_use_display
 - x11.H, [2271](#)
- fl_x11_xid
 - x11.H, [2272](#)
- Fl_XBM_Image, [1973](#)
 - Fl_XBM_Image, [1975](#)
- Fl_XBM_Image.H, [2236](#)
- Fl_XColor.H, [2311](#)
- Fl_XPM_Image, [1976](#)
 - Fl_XPM_Image, [1978](#)
- Fl_XPM_Image.H, [2236](#)
- FL_ZOOM_EVENT
 - Enumerations.H, [2011](#)
- FL_ZOOM_GESTURE
 - Enumerations.H, [2011](#)
- Flags
 - Fl_Anim_GIF_Image, [413](#)
- flstring.h, [2311](#)
- FLTK Basics, [10](#)
- FLTK Programming Manual, [1](#)
- FLTK Runtime Options, [98](#)
- flush
 - Fl, [385](#)
 - Fl_Double_Window, [643](#)
 - Fl_Gl_Window, [785](#)
 - Fl_Overlay_Window, [1170](#)
 - Fl_Preferences, [1243](#)
 - Fl_Single_Window, [1449](#)
 - Fl_Window, [1951](#)
- focus
 - Events handling functions, [246](#)
 - Fl_Group, [842](#)
- FOLIO
 - Fl_Paged_Device, [1185](#)
- font
 - Fl_Tooltip, [1760](#)
- FORCE_POSITION
 - Fl_Widget, [1891](#)
- force_position
 - Fl_Window, [1951](#)
- format
 - Fl_Valuator, [1848](#)
- format_char
 - Fl_Browser, [458](#), [459](#)
- Forms Compatibility, [122](#)
- forms.H, [2237](#)
- frame
 - Fl_Anim_GIF_Image, [417](#)
- frame_count
 - Fl_Anim_GIF_Image, [417](#)
- frame_h
 - Fl_Anim_GIF_Image, [418](#)
- frame_uncache
 - Fl_Anim_GIF_Image, [418](#)
- frame_w
 - Fl_Anim_GIF_Image, [419](#)
- frame_x
 - Fl_Anim_GIF_Image, [419](#)
- frame_y
 - Fl_Anim_GIF_Image, [419](#)
- frames
 - Fl_Anim_GIF_Image, [419](#)
- free_color
 - Color & Font functions, [272](#)
- free_icons
 - Fl_Window, [1951](#)
- free_position
 - Fl_Window, [1952](#)
- free_timeout
 - Fl_Timeout, [1740](#)
- freeglut_teapot_data.h, [2312](#)
- full_height
 - Fl_Browser, [459](#)
 - Fl_Browser_, [485](#)
- full_width
 - Fl_Browser_, [485](#)
- FULLSCREEN
 - Fl_Widget, [1891](#)
- fullscreen
 - Fl_Window, [1952](#)
- fullscreen_screens
 - Fl_Window, [1952](#)
- g
 - Fl_Color_Chooser, [601](#)
- gap
 - Fl_Flex, [719](#)
 - Fl_Grid, [822](#)
- gb2312.h, [2515](#)
- georgian_academy.h, [2545](#)
- georgian_ps.h, [2546](#)
- get
 - Fl_Preferences, [1243–1247](#)
 - Fl_Shared_Image, [1416](#), [1417](#)
 - Fl_Timeout, [1736](#)
- get_absolute_top_line_number
 - Fl_Text_Display, [1664](#)
- get_color
 - Color & Font functions, [273](#)

- get_font
 - Color & Font functions, 273
- get_font_name
 - Color & Font functions, 273
- get_font_sizes
 - Color & Font functions, 274
- get_key
 - Events handling functions, 246
- get_mouse
 - Events handling functions, 247
- get_selected_items
 - Fl_Tree, 1789
- get_selection
 - Fl_Table, 1533
 - Fl_Terminal, 1606
 - Fl_Terminal::Selection, 1985
- get_size_range
 - Fl_Window, 1952
- get_system_colors
 - Fl, 385
- get_userdata_path
 - Fl_Preferences, 1247
- gl.h, 2246, 2250
 - gl_color, 2247
 - gl_draw, 2247–2249
 - gl_font, 2249
 - gl_rect, 2249
 - gl_rectf, 2249
 - gl_texture_pile_height, 2249, 2250
- gl2opengl.h, 2251
- gl_color
 - gl.h, 2247
- gl_draw
 - gl.h, 2247–2249
- gl_draw.H, 2252
- gl_font
 - gl.h, 2249
- gl_rect
 - gl.h, 2249
- gl_rectf
 - gl.h, 2249
- gl_texture_pile_height
 - gl.h, 2249, 2250
- gl_visual
 - Fl, 385
- GLContext
 - platform_types.h, 2266
- global
 - Fl_Menu_, 1039
- global_key_bindings
 - Fl_Text_Editor, 1708
- glu.h, 2252
- GLUT Compatibility, 119
- glut.H, 2252
- grab
 - Windows handling functions, 233
- grammar_underline_color
 - Fl_Text_Display, 1664
- group
 - Fl_Preferences, 1248
- group_exists
 - Fl_Preferences, 1248
- GROUP_RELATIVE
 - Fl_Widget, 1891
- groups
 - Fl_Preferences, 1248
- h
 - Fl_Image, 926
 - Fl_Widget, 1909
- h_to_row
 - Fl_Terminal, 1606
- handle
 - Events handling functions, 247
 - Fl_Adjuster, 407
 - Fl_Box, 440
 - Fl_Browser_, 485
 - Fl_Button, 505
 - Fl_Check_Browser, 548
 - Fl_Choice, 570
 - Fl_Clock, 580
 - Fl_Color_Chooser, 601
 - Fl_Counter, 617
 - Fl_Dial, 629
 - Fl_File_Input, 687
 - Fl_Free, 763
 - Fl_Gl_Window, 786
 - Fl_Glut_Window, 806
 - Fl_Group, 842
 - Fl_Help_View, 864
 - Fl_Input, 947
 - Fl_Light_Button, 1012
 - Fl_Menu_Bar, 1056
 - Fl_Menu_Button, 1067
 - Fl_Positioner, 1220
 - Fl_Repeat_Button, 1303
 - Fl_Return_Button, 1312
 - Fl_Roller, 1328
 - Fl_Scheme_Choice, 1357
 - Fl_Scroll, 1369
 - Fl_Scrollbar, 1383
 - Fl_Secret_Input, 1395
 - Fl_Shortcut_Button, 1427
 - Fl_Slider, 1459
 - Fl_Spinner, 1470
 - Fl_Spinner::Fl_Spinner_Input, 1483
 - Fl_Table, 1533
 - Fl_Table_Row, 1553
 - Fl_Tabs, 1569
 - Fl_Terminal, 1606
 - Fl_Text_Display, 1664
 - Fl_Text_Editor, 1703
 - Fl_Tile, 1724
 - Fl_Timer, 1748
 - Fl_Tree, 1790
 - Fl_Value_Input, 1860
 - Fl_Value_Output, 1871

- FI_Value_Slider, 1882
 - FI_Widget, 1909
 - FI_Window, 1953
- handle_
 - Events handling functions, 247
- handle_key
 - FI_Input, 947
- handle_mouse
 - FI_Input_, 961
- handle_overflow
 - FI_Tabs, 1570
- handle_overflow_menu
 - FI_Tabs, 1570
- handle_rmb
 - FI_Input, 948
 - FI_Text_Display, 1664
- handle_unknown_char
 - FI_Terminal, 1607
- handle_vline
 - FI_Text_Display, 1664
- handletext
 - FI_Input_, 962
- Handling Events, 74
- has_scrollbar
 - FI_Browser_, 485
- has_timeout
 - FI, 386
 - FI_Timeout, 1737
- HEAVY_CURSOR
 - FI_Text_Display, 1654
- help
 - FI, 397
- hide
 - FI_Browser, 460
 - FI_Double_Window, 643
 - FI_GI_Window, 786
 - FI_Overlay_Window, 1170
 - FI_Widget, 1910
 - FI_Window, 1953
- hide_all_windows
 - FI, 386
- hide_widgets
 - FI_Tree_Item, 1822
- hidedelay
 - FI_Tooltip, 1760
- highlight_data
 - FI_Text_Display, 1665
- highlight_text
 - FI_Text_Buffer, 1629
- highres_image
 - FI_Image_Surface, 932
- history_lines
 - FI_Terminal, 1607
- history_use
 - FI_Terminal, 1607
- hit_close
 - FI_Tabs, 1570
- hit_overflow_menu
 - FI_Tabs, 1571
- hit_tabs_area
 - FI_Tabs, 1571
- HORIZONTAL
 - FI_Browser_, 483
 - FI_Flex, 716
- horizontal
 - FI_Flex, 719
 - FI_Pack, 1181
- HORIZONTAL_ALWAYS
 - FI_Browser_, 483
- horizontal_label_margin
 - FI_Widget, 1910
- hotspot
 - FI_Window, 1953
- hour
 - FI_Clock_Output, 589
- hoverdelay
 - FI_Tooltip, 1760
- How Does Resizing Work?, 33
- hposition
 - FI_Browser_, 486
 - FI_Tree, 1790
- hscrollbar
 - FI_Browser_, 495
 - FI_Terminal, 1619
- hscrollbar_style
 - FI_Terminal, 1607, 1608
- hsv
 - FI_Color_Chooser, 602
- hsv2rgb
 - FI_Color_Chooser, 602
- hue
 - FI_Color_Chooser, 602
- icon
 - FI_Browser, 460
 - FI_Window, 1954
- iconize
 - FI_Window, 1954
- icons
 - FI_Window, 1955
- iconsize
 - FI_File_Browser, 663
 - FI_File_Chooser, 670
- ID
 - FI_Preferences, 1234
- idle
 - FI, 397
- image
 - FI_Anim_GIF_Image, 419, 420
 - FI_Image_Surface, 932
 - FI_Menu_Item, 1076
 - FI_Shared_Image, 1417
 - FI_Widget, 1910, 1911
- IMAGE_BOUND
 - FI_Widget, 1891
- image_bound
 - FI_Widget, 1911

- image_label
 - FI_Menu_Item, 1076
- images
 - FI_Shared_Image, 1418
- imKStoUCS.c, 2383
- in_selection
 - FI_Text_Display, 1666
- INACTIVE
 - FI_Widget, 1891
- inactive
 - FI_Image, 926
- includes
 - FI_Text_Selection, 1710
- incr_height
 - FI_Browser, 460
 - FI_Browser_, 486
- increment
 - FI_Valuator, 1849
- index
 - FI_Input_, 962
- init_size_range
 - FI_Tile, 1724
- init_sizes
 - FI_Group, 842
 - FI_Table, 1533
- init_value
 - FI_Scheme_Choice, 1357
- inp_x
 - FI_Input_Choice, 986
- input
 - FI_Input_Choice, 986
- input_type
 - FI_Input_, 962
- insert
 - FI_Browser, 461
 - FI_Chart, 534
 - FI_Group, 843
 - FI_Input_, 963
 - FI_Menu_, 1039
 - FI_Menu_Item, 1076
 - FI_Sys_Menu_Bar, 1510
 - FI_Table, 1534
 - FI_Text_Buffer, 1629
 - FI_Text_Display, 1666
 - FI_Timeout, 1737
 - FI_Tree, 1791
 - FI_Tree_Item, 1822
 - FI_Tree_Item_Array, 1833
- insert_
 - FI_Text_Buffer, 1629
- insert_above
 - FI_Tree, 1791
 - FI_Tree_Item, 1823
- insert_char
 - FI_Terminal, 1608
- insert_mode
 - FI_Text_Editor, 1703
- insert_position
 - FI_Input_, 963
 - FI_Text_Display, 1666, 1667
- insert_rows
 - FI_Terminal, 1608
- insertfile
 - FI_Text_Buffer, 1629
- inserting
 - FI_Browser_, 486
- inset
 - FI_Rect, 1294
- inside
 - FI_Widget, 1911
- Introduction to FLTK, 4
- INVERSE
 - FI_Terminal, 1598
- INVISIBLE
 - FI_Widget, 1891
- is_a_rescale
 - FI_Window, 1955
- is_animated
 - FI_Anim_GIF_Image, 420
- is_close
 - FI_Tree, 1792
- is_current
 - FI_Copy_Surface, 606
 - FI_Image_Surface, 932
 - FI_PDF_File_Surface, 1192
 - FI_Printer, 1258
 - FI_Surface_Device, 1484
- is_hscroll_visible
 - FI_Tree, 1792
- is_inside_selection
 - FI_Terminal, 1608
- is_interactive_resize
 - FI_Table, 1534
- is_label_copied
 - FI_Widget, 1911
- is_open
 - FI_Tree, 1793
- is_resizable
 - FI_Window, 1955
- is_scheme
 - FI, 386
- is_scrollbar
 - FI_Tree, 1793
- is_selected
 - FI_Table, 1534
 - FI_Tree, 1794
- is_visible_r
 - FI_Tree_Item, 1823
- is_vscroll_visible
 - FI_Tree, 1794
- is_word_separator
 - FI_Text_Buffer, 1630
- iso8859_1.h, 2547
- iso8859_10.h, 2547
- iso8859_11.h, 2548
- iso8859_13.h, 2549

- iso8859_14.h, [2551](#)
- iso8859_15.h, [2552](#)
- iso8859_16.h, [2553](#)
- iso8859_2.h, [2554](#)
- iso8859_3.h, [2555](#)
- iso8859_4.h, [2557](#)
- iso8859_5.h, [2558](#)
- iso8859_6.h, [2559](#)
- iso8859_7.h, [2560](#)
- iso8859_8.h, [2561](#)
- iso8859_9.h, [2562](#)
- iso8859_9e.h, [2563](#)
- ITALIC
 - FI_Terminal, [1598](#)
- item_at
 - FI_Browser, [461](#)
 - FI_Browser_, [487](#)
 - FI_Check_Browser, [548](#)
- item_clicked
 - FI_Tree, [1795](#)
- item_draw
 - FI_Browser, [462](#)
 - FI_Browser_, [487](#)
 - FI_Check_Browser, [548](#)
- item_draw_mode
 - FI_Tree, [1795](#)
 - FI_Tree_Prefs, [1838](#)
- item_first
 - FI_Browser, [462](#)
 - FI_Browser_, [487](#)
 - FI_Check_Browser, [548](#)
- item_height
 - FI_Browser, [462](#)
 - FI_Browser_, [487](#)
 - FI_Check_Browser, [549](#)
- item_labelbgcolor
 - FI_Tree, [1795](#), [1796](#)
 - FI_Tree_Prefs, [1838](#)
- item_labelfgcolor
 - FI_Tree, [1796](#)
- item_labelfont
 - FI_Tree, [1796](#)
- item_labelsize
 - FI_Tree, [1796](#)
- item_last
 - FI_Browser, [463](#)
 - FI_Browser_, [487](#)
- item_next
 - FI_Browser, [463](#)
 - FI_Browser_, [488](#)
 - FI_Check_Browser, [549](#)
- item_pathname
 - FI_Menu_, [1039](#)
 - FI_Tree, [1796](#)
- item_prev
 - FI_Browser, [463](#)
 - FI_Browser_, [488](#)
 - FI_Check_Browser, [549](#)
- item_quick_height
 - FI_Browser_, [488](#)
- item_reselect_mode
 - FI_Tree, [1797](#)
- item_select
 - FI_Browser, [463](#)
 - FI_Browser_, [488](#)
 - FI_Check_Browser, [549](#)
- item_selected
 - FI_Browser, [464](#)
 - FI_Browser_, [489](#)
 - FI_Check_Browser, [550](#)
- item_swap
 - FI_Browser, [464](#)
 - FI_Browser_, [489](#)
 - FI_Check_Browser, [550](#)
- item_text
 - FI_Browser, [464](#)
 - FI_Browser_, [489](#)
 - FI_Check_Browser, [550](#)
- item_width
 - FI_Browser, [465](#)
 - FI_Browser_, [489](#)
 - FI_Check_Browser, [550](#)
- ivalue
 - FI_Input_, [964](#)
- jisx0201.h, [2564](#)
- jisx0208.h, [2565](#)
- jisx0212.h, [2592](#)
- keyboard_screen_scaling
 - Screen functions, [255](#)
- kf_backspace
 - FI_Text_Editor, [1703](#)
- kf_c_s_move
 - FI_Text_Editor, [1703](#)
- kf_copy
 - FI_Text_Editor, [1703](#)
- kf_ctrl_move
 - FI_Text_Editor, [1704](#)
- kf_cut
 - FI_Text_Editor, [1704](#)
- kf_default
 - FI_Text_Editor, [1704](#)
- kf_delete
 - FI_Text_Editor, [1704](#)
- kf_down
 - FI_Text_Editor, [1704](#)
- kf_end
 - FI_Text_Editor, [1704](#)
- kf_enter
 - FI_Text_Editor, [1704](#)
- kf_home
 - FI_Text_Editor, [1705](#)
- kf_ignore
 - FI_Text_Editor, [1705](#)
- kf_insert
 - FI_Text_Editor, [1705](#)

- kf_left
 - FI_Text_Editor, [1705](#)
- kf_m_s_move
 - FI_Text_Editor, [1705](#)
- kf_meta_move
 - FI_Text_Editor, [1705](#)
- kf_move
 - FI_Text_Editor, [1706](#)
- kf_page_down
 - FI_Text_Editor, [1706](#)
- kf_page_up
 - FI_Text_Editor, [1706](#)
- kf_paste
 - FI_Text_Editor, [1706](#)
- kf_redo
 - FI_Text_Editor, [1706](#)
- kf_right
 - FI_Text_Editor, [1706](#)
- kf_select_all
 - FI_Text_Editor, [1706](#)
- kf_shift_move
 - FI_Text_Editor, [1707](#)
- kf_undo
 - FI_Text_Editor, [1707](#)
- kf_up
 - FI_Text_Editor, [1707](#)
- koi8_c.h, [2617](#)
- koi8_r.h, [2619](#)
- koi8_u.h, [2620](#)
- ksc5601.h, [2622](#)
- label
 - FI_Bitmap, [428](#)
 - FI_File_Icon, [674](#)
 - FI_Image, [927](#)
 - FI_Menu_Item, [1077](#), [1078](#)
 - FI_Multi_Label, [1110](#)
 - FI_Pixmap, [1199](#)
 - FI_RGB_Image, [1319](#)
 - FI_Tree_Item, [1823](#)
 - FI_Widget, [1912](#)
- label_h
 - FI_Tree_Item, [1823](#)
- label_image_spacing
 - FI_Widget, [1912](#)
- label_shortcut
 - FI_Widget, [1913](#)
- label_w
 - FI_Tree_Item, [1823](#)
- label_x
 - FI_Tree_Item, [1823](#)
- label_y
 - FI_Tree_Item, [1824](#)
- labela
 - FI_Multi_Label, [1111](#)
- labelb
 - FI_Multi_Label, [1111](#)
- labelbgcolor
 - FI_Tree_Item, [1824](#)
- labelcolor
 - FI_Menu_Item, [1078](#)
 - FI_Widget, [1913](#)
- labelfont
 - FI_Menu_Item, [1078](#)
 - FI_Widget, [1913](#), [1914](#)
- labelsize
 - FI_Widget, [1914](#)
- labeltype
 - FI_File_Icon, [675](#)
 - FI_Menu_Item, [1079](#)
 - FI_Widget, [1914](#), [1915](#)
- LANDSCAPE
 - FI_Paged_Device, [1185](#)
- last
 - FI_Tree, [1797](#)
- last_handler
 - Events handling functions, [248](#)
- last_selected_item
 - FI_Tree, [1797](#)
- last_visible
 - FI_Tree, [1798](#)
- last_visible_item
 - FI_Tree, [1798](#)
- layout
 - FI_Flex, [720](#)
 - FI_Grid, [823](#)
- ld
 - FI_Image, [927](#)
- LEDGER
 - FI_Paged_Device, [1185](#)
- leftedge
 - FI_Browser_, [490](#)
- leftline
 - FI_Help_View, [864](#)
- LEGAL
 - FI_Paged_Device, [1185](#)
- length
 - FI_Text_Buffer, [1630](#)
 - FI_Text_Selection, [1710](#)
- LETTER
 - FI_Paged_Device, [1185](#)
- LF_TO_CR
 - FI_Terminal, [1599](#)
- LF_TO_CRLF
 - FI_Terminal, [1599](#)
- line_end
 - FI_Input_, [964](#)
 - FI_Text_Buffer, [1630](#)
 - FI_Text_Display, [1667](#)
- line_start
 - FI_Input_, [964](#)
 - FI_Text_Buffer, [1630](#)
 - FI_Text_Display, [1667](#)
- line_text
 - FI_Text_Buffer, [1631](#)
- lineno
 - FI_Browser, [465](#)

- linenumber_align
 - FI_Text_Display, 1668
- linenumber_bgcolor
 - FI_Text_Display, 1668
- linenumber_fgcolor
 - FI_Text_Display, 1668
- linenumber_font
 - FI_Text_Display, 1668
- linenumber_format
 - FI_Text_Display, 1668
- linenumber_size
 - FI_Text_Display, 1669
- linenumber_width
 - FI_Text_Display, 1669
- lineposition
 - FI_Browser, 465
- linesize
 - FI_Scrollbar, 1383
- linespacing
 - FI_Browser_, 490
- link
 - FI_Help_View, 865
- load
 - FI_Anim_GIF_Image, 420
 - FI_Browser, 466
 - FI_File_Browser, 664
 - FI_File_Icon, 675
 - FI_Help_Dialog, 849
 - FI_Help_View, 865
 - FI_Plugin_Manager, 1205
 - FI_Tree, 1798
- load_fti
 - FI_File_Icon, 675
- load_image
 - FI_File_Icon, 675
- load_system_icons
 - FI_File_Icon, 676
- loadAll
 - FI_Plugin_Manager, 1205
- loadfile
 - FI_Text_Buffer, 1631
- lock
 - Multithreading support functions, 312
- LOG_FLAG
 - FI_Anim_GIF_Image, 414
- longest_vline
 - FI_Text_Display, 1669
- loop
 - FI_Anim_GIF_Image, 423
- lstep
 - FI_Counter, 618
- Mac OS X-specific symbols, 335
 - fl_mac_os_version, 336
 - fl_mac_set_about, 335
 - fl_open_callback, 335
- mac.H, 2259
- MAC_USE_ACCENTS_MENU
 - FI_Widget, 1891
- maintain_absolute_top_line_number
 - FI_Text_Display, 1669
- maintaining_absolute_top_line_number
 - FI_Text_Display, 1669
- make_current
 - FI_GL_Window, 786
 - FI_Timeout, 1737
 - FI_Window, 1956
- make_overlay_current
 - FI_GL_Window, 786
- make_visible
 - FI_Browser, 466
- manage_item_destroy
 - FI_Tree_Item_Array, 1833
- margin
 - FI_Flex, 720, 721
 - FI_Grid, 823, 824
- margin_height
 - FI_Tooltip, 1760
- margin_width
 - FI_Tooltip, 1761
- marginbottom
 - FI_Tree_Prefs, 1838
- margins
 - FI_Paged_Device, 1186
 - FI_PDF_File_Surface, 1192
 - FI_PostScript_File_Device, 1227
 - FI_Printer, 1258
- mark
 - FI_Input_, 965
- mask
 - FI_Image_Surface, 932
- math.h, 2261
- max_size
 - FI_RGB_Image, 1319
- maximize
 - FI_Window, 1956
- MAXIMIZED
 - FI_Widget, 1891
- maximum
 - FI_Progress, 1267, 1268
 - FI_Valuator, 1849
- maximum_size
 - FI_Input_, 965
- maxsize
 - FI_Chart, 534
- maybe_do_callback
 - FI_Tabs, 1571
- measure
 - FI_Label, 1003
 - FI_Menu_Item, 1079
- measure_deleted_lines
 - FI_Text_Display, 1669
- measure_label
 - FI_Widget, 1915
- measure_proportional_character
 - FI_Text_Display, 1670
- measure_vline

- FI_Text_Display, 1670
- mediumarrow.h, 2314
- MEMORY
 - FI_Preferences, 1235
- menu
 - FI_Menu_, 1040
 - FI_Sys_Menu_Bar, 1511
- menu_box
 - FI_Menu_, 1041
- menu_end
 - FI_Menu_, 1041
- menu_linespacing
 - FI, 387
- MENU_WINDOW
 - FI_Widget, 1891
- menu_x
 - FI_Input_Choice, 986
- menubutton
 - FI_Input_Choice, 986
- middleline
 - FI_Browser, 466
- Migrating Code from FLTK 1.3 to 1.4, 143
- min_delay
 - FI_Anim_GIF_Image, 423
- minimum
 - FI_Progress, 1268
 - FI_Valuator, 1849
- minute
 - FI_Clock_Output, 589
- mk_wcwidth.c, 2665
- MODAL
 - FI_Widget, 1891
- modal
 - FI_Window, 1956
 - Windows handling functions, 233
- mode
 - FI_Color_Chooser, 602, 603
 - FI_Gl_Window, 786
 - FI_Menu_, 1041, 1042
 - FI_Sys_Menu_Bar, 1511
- move
 - FI_Browser, 466
 - FI_Tree_Item, 1824
 - FI_Tree_Item_Array, 1834
- move_above
 - FI_Tree_Item, 1825
- move_below
 - FI_Tree_Item, 1825
- move_cursor
 - FI_Table, 1534
- move_down
 - FI_Text_Display, 1670
- move_intersection
 - FI_Tile, 1725
- move_into
 - FI_Tree_Item, 1825
- move_left
 - FI_Text_Display, 1671
- move_right
 - FI_Text_Display, 1671
- move_up
 - FI_Text_Display, 1671
- mTabDist
 - FI_Text_Buffer, 1637
- mulelao.h, 2657
- MULTI
 - FI_File_Chooser, 669
- multi_label
 - FI_Menu_Item, 1079
- Multithreading support functions, 311
 - awake, 312
 - lock, 312
 - thread_message, 312
 - unlock, 312
- mvalue
 - FI_Menu_, 1042
- Name
 - FI_Preferences::Name, 1981
- name
 - FI_Anim_GIF_Image, 420
- names
 - FI_Scheme, 1346
- names.h, 2262
- need_layout
 - FI_Flex, 721, 722
 - FI_Grid, 824
- NEEDS_KEYBOARD
 - FI_Widget, 1891
- needs_keyboard
 - FI_Widget, 1915
- NEW_FOLDER
 - FI_Native_File_Chooser, 1135
- new_list
 - FI_Browser_, 490
- new_UUID
 - FI_Preferences, 1248
- next
 - FI_Anim_GIF_Image, 421
 - FI_File_Icon, 676
 - FI_Grid::Cell, 364
 - FI_Menu_Item, 1079
 - FI_Tree, 1798
 - FI_Tree_Item, 1826
 - FI_Wizard, 1972
- next_char
 - FI_Text_Buffer, 1631
- next_displayed
 - FI_Tree_Item, 1826
- next_item
 - FI_Tree, 1799
- next_selected_item
 - FI_Tree, 1800
- next_sibling
 - FI_Tree_Item, 1826
- next_visible
 - FI_Tree_Item, 1826

- next_visible_item
 - FI_Tree, [1800](#)
- next_window
 - Windows handling functions, [234](#)
- nitems
 - FI_Check_Browser, [551](#)
- NO_OPTIONS
 - FI_Native_File_Chooser, [1134](#)
- NO_OVERLAY
 - FI_Widget, [1891](#)
- NO_REDRAW
 - FI_Terminal, [1599](#)
- no_window_menu
 - FI_Sys_Menu_Bar, [1508](#)
- NOBORDER
 - FI_Widget, [1891](#)
- NON_MODAL
 - FI_Widget, [1891](#)
- NONE
 - FI_Preferences, [1252](#)
- NORMAL
 - FI_Terminal, [1598](#)
- NORMAL_CURSOR
 - FI_Text_Display, [1654](#)
- normalize
 - FI_RGB_Image, [1319](#)
 - FI_SVG_Image, [1496](#)
- now
 - FI, [387](#)
- num_images
 - FI_Shared_Image, [1418](#)
- num_schemes
 - FI_Scheme, [1347](#)
- numeric_sort.c, [2314](#)
 - fl_casnumeric_sort, [2314](#)
 - fl_numeric_sort, [2315](#)
- OFF
 - FI_Terminal, [1599](#)
- offscreen
 - FI_Image_Surface, [933](#)
- offset_line_starts
 - FI_Text_Display, [1671](#)
- on_extension_data
 - FI_Anim_GIF_Image, [421](#)
- on_frame_data
 - FI_Anim_GIF_Image, [421](#)
- on_insert
 - FI_Group, [843](#)
 - FI_Scroll, [1370](#)
 - FI_Tabs, [1571](#)
 - FI_Tile, [1725](#)
- on_move
 - FI_Group, [843](#)
 - FI_Scroll, [1371](#)
 - FI_Tabs, [1572](#)
 - FI_Tile, [1725](#)
- on_remove
 - FI_Flex, [722](#)
 - FI_Grid, [825](#)
 - FI_Group, [844](#)
 - FI_Tabs, [1572](#)
 - FI_Tile, [1725](#)
- open
 - FI_Tree, [1801](#)
- open_toggle
 - FI_Tree, [1802](#)
- opendeicon
 - FI_Tree_Prefs, [1838](#)
- openicon
 - FI_Tree, [1802](#)
 - FI_Tree_Prefs, [1838](#), [1839](#)
- Operating System Issues, [127](#)
- OPTIMIZE_MEMORY
 - FI_Anim_GIF_Image, [414](#)
- Option
 - FI_Native_File_Chooser, [1134](#)
- option
 - FI, [388](#), [389](#)
- OPTION_ARROW_FOCUS
 - FI, [376](#)
- OPTION_DND_TEXT
 - FI, [376](#)
- OPTION_FNFC_USES_GTK
 - FI, [376](#)
- OPTION_FNFC_USES_KDIALOG
 - FI, [376](#)
- OPTION_FNFC_USES_ZENITY
 - FI, [376](#)
- OPTION_LAST
 - FI, [377](#)
- OPTION_PRINTER_USES_GTK
 - FI, [376](#)
- OPTION_SHOW_SCALING
 - FI, [376](#)
- OPTION_SHOW_TOOLTIPS
 - FI, [376](#)
- OPTION_SIMPLE_ZOOM_SHORTCUT
 - FI, [377](#)
- OPTION_VISIBLE_FOCUS
 - FI, [376](#)
- options
 - FI_Native_File_Chooser, [1137](#)
- ORIENTATION
 - FI_Paged_Device, [1185](#)
- origin
 - FI_Copy_Surface, [606](#)
 - FI_EPS_File_Surface, [648](#)
 - FI_Image_Surface, [933](#), [934](#)
 - FI_PDF_File_Surface, [1193](#)
 - FI_PostScript_File_Device, [1227](#)
 - FI_Printer, [1258](#)
 - FI_SVG_File_Surface, [1488](#)
 - FI_Widget_Surface, [1929](#)
- original
 - FI_Shared_Image, [1418](#)
- ortho

- FI_GI_Window, [787](#)
- os_id
 - FI_Window, [1956](#)
- OutFlags
 - FI_Terminal, [1598](#)
- OUTPUT
 - FI_Widget, [1891](#)
- output
 - FI_Widget, [1916](#)
- output_translate
 - FI_Terminal, [1608](#)
- outputfile
 - FI_Text_Buffer, [1631](#)
- OVERFLOW_CLIP
 - FI_Tabs, [1568](#)
- OVERFLOW_COMPRESS
 - FI_Tabs, [1568](#)
- OVERFLOW_DRAG
 - FI_Tabs, [1568](#)
- OVERFLOW_PULLDOWN
 - FI_Tabs, [1568](#)
- overflow_type
 - FI_Tabs, [1576](#)
- OVERRIDE
 - FI_Widget, [1891](#)
- overstrike
 - FI_Text_Display, [1671](#)
- own_colormap
 - FI, [389](#)
- Page_Format
 - FI_Paged_Device, [1184](#)
- Page_Layout
 - FI_Paged_Device, [1185](#)
- parent
 - FI_Tree_Item, [1826](#)
 - FI_Widget, [1916](#)
- paste
 - Selection & Clipboard functions, [252](#)
- PER_WRITE
 - FI_Terminal, [1599](#)
- picked
 - FI_Menu_, [1042](#)
- pixel_h
 - FI_GI_Window, [787](#)
- pixel_w
 - FI_GI_Window, [788](#)
- pixels_per_unit
 - FI_GI_Window, [788](#)
- Pixmap
 - FI_FormsPixmap, [748](#)
- platform.H, [2263](#)
- platform_types.h, [2264](#), [2266](#)
 - fl_intptr_t, [2265](#)
 - FI_Offscreen, [2265](#)
 - FI_Region, [2265](#)
 - FI_Timestamp, [2266](#)
 - fl_uintptr_t, [2266](#)
 - GLContext, [2266](#)
- play_menu
 - FI_Menu_Bar, [1057](#)
 - FI_Sys_Menu_Bar, [1511](#)
- playing
 - FI_Anim_GIF_Image, [421](#)
- plot_char
 - FI_Terminal, [1609](#)
- pop_current
 - FI_Surface_Device, [1484](#)
- POPUP
 - FI_Widget, [1891](#)
- popup
 - FI_Menu_Button, [1068](#)
 - FI_Menu_Item, [1079](#)
- POPUP1
 - FI_Menu_Button, [1067](#)
- POPUP12
 - FI_Menu_Button, [1067](#)
- POPUP123
 - FI_Menu_Button, [1067](#)
- POPUP13
 - FI_Menu_Button, [1067](#)
- POPUP2
 - FI_Menu_Button, [1067](#)
- POPUP23
 - FI_Menu_Button, [1067](#)
- POPUP3
 - FI_Menu_Button, [1067](#)
- popup_buttons
 - FI_Menu_Button, [1067](#)
- PORTRAIT
 - FI_Paged_Device, [1185](#)
- position
 - FI_Browser_, [490](#)
 - FI_Input_, [965](#), [966](#)
 - FI_Text_Selection, [1710](#)
 - FI_Tile, [1725](#)
 - FI_Widget, [1916](#)
- position_style
 - FI_Text_Display, [1671](#)
- position_to_line
 - FI_Text_Display, [1672](#)
- position_to_linecol
 - FI_Text_Display, [1672](#)
- position_to_xy
 - FI_Text_Display, [1673](#)
- precision
 - FI_Valuator, [1849](#)
- Preface, [2](#)
- prefs
 - FI_Tree_Item, [1826](#)
- preset_file
 - FI_Native_File_Chooser, [1137](#)
- prev
 - FI_Tree, [1803](#)
 - FI_Tree_Item, [1827](#)
- prev_char
 - FI_Text_Buffer, [1632](#)

- prev_displayed
 - FI_Tree_Item, 1827
- prev_mvalue
 - FI_Menu_, 1042
- prev_sibling
 - FI_Tree_Item, 1827
- prev_visible
 - FI_Tree_Item, 1827
- PREVIEW
 - FI_Native_File_Chooser, 1135
- preview
 - FI_File_Chooser, 670
- print
 - FI_Mac_App_Menu, 1023
- print_button.h, 2316
- print_char
 - FI_Terminal, 1609, 1610
- print_panel.h, 2316
- print_window_part
 - FI_Widget_Surface, 1929
- printable_rect
 - FI_Copy_Surface, 607
 - FI_EPS_File_Surface, 648
 - FI_Image_Surface, 934
 - FI_PDF_File_Surface, 1193
 - FI_PostScript_File_Device, 1228
 - FI_Printer, 1259
 - FI_SVG_File_Surface, 1489
 - FI_Widget_Surface, 1930
- printf
 - FI_Terminal, 1610
 - FI_Text_Buffer, 1632
- program_should_quit
 - FI, 389
- proportional
 - FI_SVG_Image, 1496
- pulldown
 - FI_Menu_Item, 1080
- push
 - FI_Tabs, 1572
- push_current
 - FI_Surface_Device, 1484
- pushed
 - Events handling functions, 248
- r
 - FI_Color_Chooser, 603
 - FI_Rect, 1294
- radio
 - FI_Menu_Item, 1080
- range
 - FI_Valuator, 1849
- RATE_LIMITED
 - FI_Terminal, 1599
- readonly
 - FI_Input_, 966
- readqueue
 - FI, 390
- ready
 - FI, 390
- recalc_dimensions
 - FI_Table, 1534
- recalc_display
 - FI_Text_Display, 1673
- recalc_scrollbars
 - FI_Scroll, 1371
- recalc_tree
 - FI_Tree, 1803
 - FI_Tree_Item, 1827
- rectangle_capture
 - FI_Device_Plugin, 620
- redisplay_range
 - FI_Text_Display, 1673
- redo
 - FI_Input_, 966
- redraw
 - FI_Widget, 1918
- redraw_label
 - FI_Widget, 1918
- redraw_line
 - FI_Browser_, 491
- redraw_lines
 - FI_Browser_, 491
- redraw_overlay
 - FI_GL_Window, 788
 - FI_Overlay_Window, 1170
- redraw_range
 - FI_Table, 1534
- redraw_rate
 - FI_Terminal, 1610
- redraw_style
 - FI_Terminal, 1611
- redraw_tabs
 - FI_Tabs, 1572
- RedrawStyle
 - FI_Terminal, 1599
- refcount
 - FI_Shared_Image, 1418
- release
 - FI, 390
 - FI_Image, 927
 - FI_Shared_Image, 1418
 - FI_Timeout, 1738
- release_widget_pointer
 - Safe widget deletion support functions, 314
- reload_scheme
 - FI, 390
- remove
 - FI_Browser, 467
 - FI_Check_Browser, 551
 - FI_Group, 844
 - FI_Menu_, 1042
 - FI_Sys_Menu_Bar, 1511
 - FI_Text_Buffer, 1632
 - FI_Tree, 1803
 - FI_Tree_Item_Array, 1834
- remove_

- FI_Text_Buffer, 1632
- remove_check
 - FI, 390
- remove_child
 - FI_Tree_Item, 1827, 1828
- remove_handler
 - Events handling functions, 248
- remove_icon
 - FI_Browser, 467
- remove_key_binding
 - FI_Text_Editor, 1707
- remove_next_timeout
 - FI, 391
 - FI_Timeout, 1738
- remove_system_handler
 - Events handling functions, 248
- remove_timeout
 - FI, 391
 - FI_Timeout, 1738
- removePlugin
 - FI_Plugin_Manager, 1205
- reparent
 - FI_Tree_Item, 1828
 - FI_Tree_Item_Array, 1834
- repeat_timeout
 - FI, 392
 - FI_Timeout, 1739
- replace
 - FI_Chart, 535
 - FI_Input_, 966
 - FI_Menu_, 1042
 - FI_Sys_Menu_Bar, 1511
 - FI_Text_Buffer, 1632
 - FI_Tree_Item, 1828
 - FI_Tree_Item_Array, 1834
- replace_child
 - FI_Tree_Item, 1829
- replacing
 - FI_Browser_, 491
- request_grow_b
 - FI_Tile, 1725
- request_grow_l
 - FI_Tile, 1726
- request_grow_r
 - FI_Tile, 1726
- request_grow_t
 - FI_Tile, 1726
- request_shrink_b
 - FI_Tile, 1727
- request_shrink_l
 - FI_Tile, 1727
- request_shrink_r
 - FI_Tile, 1727
- request_shrink_t
 - FI_Tile, 1728
- rescale
 - FI_Image_Surface, 934
- reset_absolute_top_line_number
 - FI_Text_Display, 1674
- reset_terminal
 - FI_Terminal, 1611
- resizable
 - FI_Group, 845
- resize
 - FI_Anim_GIF_Image, 421
 - FI_Browser_, 491
 - FI_Double_Window, 644
 - FI_Flex, 722
 - FI_Gl_Window, 788
 - FI_Grid, 825
 - FI_Group, 846
 - FI_Help_View, 865
 - FI_Input_, 967
 - FI_Input_Choice, 987
 - FI_Overlay_Window, 1170
 - FI_Pack, 1181
 - FI_Scroll, 1371
 - FI_Spinner, 1471
 - FI_SVG_Image, 1496
 - FI_Table, 1535
 - FI_Tabs, 1572
 - FI_Terminal, 1611
 - FI_Text_Display, 1674
 - FI_Tile, 1728
 - FI_Tree, 1803
 - FI_Value_Input, 1860
 - FI_Widget, 1918
 - FI_Window, 1957
- REVERSED
 - FI_Paged_Device, 1185
- rewind_lines
 - FI_Text_Buffer, 1633
 - FI_Text_Display, 1674
- rgb
 - FI_Color_Chooser, 603
- rgb2hsv
 - FI_Color_Chooser, 603
- RGB_scaling
 - FI_Image, 927
- Root
 - FI_Preferences, 1234
- root
 - FI_Tree, 1804
- root_label
 - FI_Tree, 1804
- ROOT_MASK
 - FI_Preferences, 1235
- rotate
 - FI_Paged_Device, 1187
 - FI_PDF_File_Surface, 1194
 - FI_PostScript_File_Device, 1228
 - FI_Printer, 1259
- round
 - FI_Valuator, 1850
- ROW
 - FI_Flex, 716

- row_col_clamp
 - FI_Table, [1535](#)
- row_gap
 - FI_Grid, [825](#)
- row_header
 - FI_Table, [1535](#)
- row_height
 - FI_Grid, [825](#), [826](#)
 - FI_Table, [1535](#)
- row_height_all
 - FI_Table, [1535](#)
- row_resize
 - FI_Table, [1535](#)
- row_resize_min
 - FI_Table, [1535](#)
- row_selected
 - FI_Table_Row, [1553](#)
- row_weight
 - FI_Grid, [826](#)
- rows
 - FI_Table, [1536](#)
 - FI_Table_Row, [1553](#)
- run
 - FI, [392](#)
- Safe widget deletion support functions, [313](#)
 - clear_widget_pointer, [314](#)
 - delete_widget, [314](#)
 - do_widget_deletion, [314](#)
 - release_widget_pointer, [314](#)
 - watch_widget_pointer, [315](#)
- saturation
 - FI_Color_Chooser, [603](#)
- SAVEAS_CONFIRM
 - FI_Native_File_Chooser, [1134](#)
- savefile
 - FI_Text_Buffer, [1633](#)
- scale
 - FI_Image, [928](#)
 - FI_Paged_Device, [1187](#)
 - FI_PDF_File_Surface, [1194](#)
 - FI_PostScript_File_Device, [1228](#)
 - FI_Printer, [1259](#)
- scaling_algorithm
 - FI_Image, [928](#)
- scheme
 - FI, [392](#)
- scheme_cb_
 - FI_Scheme_Choice, [1357](#)
- Screen functions, [254](#)
 - keyboard_screen_scaling, [255](#)
 - screen_count, [255](#)
 - screen_dpi, [256](#)
 - screen_num, [256](#)
 - screen_scale, [257](#)
 - screen_scaling_supported, [257](#)
 - screen_work_area, [258](#)
 - screen_xywh, [260](#), [261](#)
- Screen functions, [255](#)
- screen_dpi
 - Screen functions, [256](#)
- screen_num
 - FI_Window, [1957](#)
 - Screen functions, [256](#)
- screen_scale
 - Screen functions, [257](#)
- screen_scaling_supported
 - Screen functions, [257](#)
- screen_work_area
 - Screen functions, [258](#)
- screen_xywh
 - Screen functions, [260](#), [261](#)
- scroll
 - FI_Terminal, [1612](#)
 - FI_Text_Display, [1674](#)
- scroll_
 - FI_Text_Display, [1674](#)
- scroll_timer_cb
 - FI_Text_Display, [1675](#)
- scroll_to
 - FI_Scroll, [1372](#)
- scrollbar
 - FI_Browser_, [495](#)
 - FI_Terminal, [1619](#)
- scrollbar_actual_size
 - FI_Terminal, [1612](#)
- scrollbar_align
 - FI_Text_Display, [1675](#)
- SCROLLBAR_AUTO
 - FI_Terminal, [1599](#)
- scrollbar_left
 - FI_Browser_, [491](#)
- SCROLLBAR_OFF
 - FI_Terminal, [1599](#)
- SCROLLBAR_ON
 - FI_Terminal, [1599](#)
- scrollbar_right
 - FI_Browser_, [492](#)
- scrollbar_size
 - FI, [393](#)
 - FI_Browser_, [492](#)
 - FI_Help_View, [865](#), [866](#)
 - FI_Scroll, [1372](#)
 - FI_Table, [1536](#)
 - FI_Terminal, [1612](#)
 - FI_Text_Display, [1675](#)
 - FI_Tree, [1804](#)
- scrollbar_width
 - FI_Browser_, [492](#)
 - FI_Text_Display, [1676](#)
- ScrollbarStyle
 - FI_Terminal, [1599](#)
- scrollvalue
 - FI_Slider, [1459](#)
- search_backward
 - FI_Text_Buffer, [1633](#)
- screen_count

- search_forward
 - FI_Text_Buffer, 1634
- second
 - FI_Clock_Output, 589
- secondary_selection_color
 - FI_Text_Display, 1676
- secondary_selection_text
 - FI_Text_Buffer, 1634
- seconds_between
 - FI, 393
- seconds_since
 - FI, 394
- select
 - FI_Browser, 467
 - FI_Browser_, 493
 - FI_Tree, 1804, 1805
 - FI_Tree_Item, 1829
- select_all
 - FI_Tree, 1805
 - FI_Tree_Item, 1829
- select_all_rows
 - FI_Table_Row, 1553
- select_only
 - FI_Browser_, 493
 - FI_Tree, 1806
- select_row
 - FI_Table_Row, 1554
- select_toggle
 - FI_Tree, 1806
- selectbox
 - FI_Tree, 1807
- selected
 - FI_Browser, 468
 - FI_Text_Selection, 1710, 1711
- selection
 - FI_Browser_, 493
 - Selection & Clipboard functions, 253
- Selection & Clipboard functions, 250
 - add_clipboard_notify, 251
 - clipboard_contains, 251
 - copy, 251
 - dnd, 252
 - paste, 252
 - selection, 253
 - selection_owner, 253
 - selection_to_clipboard, 253
- selection_color
 - FI_Widget, 1918
- selection_extend
 - FI_Terminal, 1613
- selection_owner
 - Selection & Clipboard functions, 253
- selection_text
 - FI_Terminal, 1613
 - FI_Text_Buffer, 1634
- selection_text_len
 - FI_Terminal, 1613
- selection_to_clipboard
 - Selection & Clipboard functions, 253
- selectmode
 - FI_Tree, 1807
 - FI_Tree_Prefs, 1839
- set
 - FI_Button, 506
 - FI_FormsBitmap, 740
 - FI_FormsPixmap, 748
 - FI_Menu_Item, 1080
 - FI_Preferences, 1249–1251
 - FI_Text_Selection, 1711
- set_active
 - FI_Widget, 1919
- set_atclose
 - Windows handling functions, 234
- set_box_color
 - FI, 394
- set_boxtype
 - FI, 394
- set_changed
 - FI_Widget, 1919
- set_color
 - Color & Font functions, 274
- set_current
 - FI_Copy_Surface, 607
 - FI_Image_Surface, 934
 - FI_PDF_File_Surface, 1194
 - FI_PostScript_File_Device, 1229
 - FI_Printer, 1259
 - FI_Surface_Device, 1485
- set_cursor
 - FI_Tile, 1728
- set_draw_cb
 - FI_Cairo_Window, 521
- set_font
 - Color & Font functions, 274
- set_fonts
 - Color & Font functions, 275
- set_idle
 - FI, 395
- set_item_focus
 - FI_Tree, 1807
- set_menu_window
 - FI_Window, 1958
- set_modal
 - FI_Window, 1958
- set_non_modal
 - FI_Window, 1958
- set_output
 - FI_Widget, 1919
- set_selection
 - FI_Table, 1536
- set_tooltip_window
 - FI_Window, 1958
- set_visible
 - FI_Widget, 1919
- set_visible_focus
 - FI_Widget, 1919

- setonly
 - FI_Menu_Item, 1080
- shadow
 - FI_Clock_Output, 590
- shape
 - FI_Window, 1959
- shortcut
 - FI_Button, 506
 - FI_Input_, 967
 - FI_Menu_Item, 1081
 - FI_Text_Display, 1676, 1677
 - FI_Value_Input, 1861
- SHORTCUT_LABEL
 - FI_Widget, 1891
- shortcut_label
 - FI_Widget, 1919, 1920
- show
 - FI_Browser, 468
 - FI_Double_Window, 644
 - FI_GL_Window, 788
 - FI_Help_Dialog, 849
 - FI_Native_File_Chooser, 1137
 - FI_Overlay_Window, 1171
 - FI_Single_Window, 1449
 - FI_Tabs, 1573
 - FI_Widget, 1920
 - FI_Window, 1960
- show_cursor
 - FI_Text_Display, 1677
- show_grid
 - FI_Grid, 826, 827
- show_insert_position
 - FI_Text_Display, 1677
- show_item
 - FI_Tree, 1807, 1808
- show_item_bottom
 - FI_Tree, 1808
- show_item_middle
 - FI_Tree, 1808
- show_item_top
 - FI_Tree, 1808
- show_next_window_iconic
 - FI_Window, 1961
- show_self
 - FI_Tree, 1809
 - FI_Tree_Item, 1829
- show_unknown
 - FI_Terminal, 1613
- show_widgets
 - FI_Tree_Item, 1829
- showcollapse
 - FI_Tree, 1809
 - FI_Tree_Prefs, 1839
- showHiddenButton
 - FI_File_Chooser, 671
- shown
 - FI_File_Chooser, 670
 - FI_Window, 1961
- showroot
 - FI_Tree, 1809
 - FI_Tree_Prefs, 1839
- SIMPLE_CURSOR
 - FI_Text_Display, 1654
- SINGLE
 - FI_File_Chooser, 669
- size
 - FI_Browser, 468
 - FI_Chart, 535
 - FI_Input_, 968
 - FI_Menu_, 1043
 - FI_Menu_Item, 1081
 - FI_Preferences, 1251
 - FI_Tooltip, 1761
 - FI_Widget, 1920
- size_range
 - FI_Tile, 1728, 1729
 - FI_Window, 1961
- sizes
 - FI_Group, 847
- skip_displayed_characters
 - FI_Text_Buffer, 1634
- skip_lines
 - FI_Text_Display, 1677
- slider_size
 - FI_Slider, 1460
- slowarrow.h, 2317
- soft
 - FI_Adjuster, 407, 408
 - FI_Value_Input, 1861
 - FI_Value_Output, 1871
- Software License, 149
- sort
 - FI_Browser_, 493
- sortorder
 - FI_Tree, 1809
 - FI_Tree_Prefs, 1839
- spacing
 - FI_Flex, 722
- spacing.h, 2347
- speed
 - FI_Anim_GIF_Image, 422
- spelling_underline_color
 - FI_Text_Display, 1678
- start
 - FI_Anim_GIF_Image, 422
 - FI_Text_Selection, 1711
- start_job
 - FI_Paged_Device, 1187
 - FI_PostScript_File_Device, 1229
- start_page
 - FI_Paged_Device, 1188
- static_value
 - FI_Input_, 968
- step
 - FI_Counter, 618
 - FI_Spinner, 1471

- FI_Valuator, [1850](#)
- stop
 - FI_Anim_GIF_Image, [422](#)
- STRICT_RFC3629
 - Unicode and UTF-8 functions, [320](#)
- STRIKEOUT
 - FI_Terminal, [1598](#)
- String handling functions, [334](#)
 - fl_strdup, [334](#)
- string_width
 - FI_Text_Display, [1678](#)
- style_buffer
 - FI_Text_Display, [1678](#)
- submenu
 - FI_Menu_Item, [1081](#)
- surface
 - FI_Surface_Device, [1485](#)
- suspended
 - FI_Timer, [1748](#)
- swap
 - FI_Browser, [468](#), [469](#)
- swap_buffers
 - FI_GL_Window, [789](#)
- swap_children
 - FI_Tree_Item, [1830](#)
- swap_interval
 - FI_GL_Window, [789](#)
- swapping
 - FI_Browser_, [494](#)
- symbol_.h, [2370](#)
- SYSTEM
 - FI_Preferences, [1235](#)
- SYSTEM_L
 - FI_Preferences, [1235](#)
- tab_align
 - FI_Tabs, [1573](#)
- tab_cell_nav
 - FI_Table, [1537](#)
- tab_count
 - FI_Tabs, [1576](#)
- tab_distance
 - FI_Text_Buffer, [1635](#)
- tab_flags
 - FI_Tabs, [1576](#)
- tab_height
 - FI_Tabs, [1573](#)
- tab_nav
 - FI_Input_, [969](#)
 - FI_Text_Editor, [1707](#), [1708](#)
- tab_pos
 - FI_Tabs, [1576](#)
- tab_positions
 - FI_Tabs, [1573](#)
- tab_width
 - FI_Tabs, [1577](#)
- tabbing_mode_automatic
 - FI_Sys_Menu_Bar, [1508](#)
- tabbing_mode_none
 - FI_Sys_Menu_Bar, [1508](#)
- tabbing_mode_preferred
 - FI_Sys_Menu_Bar, [1508](#)
- table_box
 - FI_Table, [1537](#)
- table_resized
 - FI_Table, [1537](#)
- table_scrolled
 - FI_Table, [1537](#)
- TableContext
 - FI_Table, [1528](#)
- TABLOID
 - FI_Paged_Device, [1185](#)
- take_focus
 - FI_Tabs, [1574](#)
 - FI_Widget, [1920](#)
- takeevents
 - FI_Widget, [1921](#)
- tatar_cyr.h, [2658](#)
- tcvn.h, [2659](#)
- test_shortcut
 - Events handling functions, [248](#)
 - FI_Menu_, [1043](#)
 - FI_Menu_Item, [1081](#)
 - FI_Widget, [1921](#)
- text
 - FI_Browser, [469](#)
 - FI_Menu_, [1043](#)
 - FI_Terminal, [1613](#)
 - FI_Text_Buffer, [1635](#)
- text_range
 - FI_Text_Buffer, [1635](#)
- text_selected
 - FI_Help_View, [866](#)
- textattrib
 - FI_Terminal, [1614](#)
- textbgcolor
 - FI_Terminal, [1614](#)
- textbgcolor_default
 - FI_Terminal, [1614](#), [1615](#)
- textbgcolor_xterm
 - FI_Terminal, [1615](#)
- textcolor
 - FI_Input_, [969](#), [970](#)
 - FI_Menu_, [1043](#)
 - FI_Terminal, [1615](#)
 - FI_Text_Display, [1678](#), [1679](#)
 - FI_Tooltip, [1761](#)
 - FI_Value_Input, [1861](#)
 - FI_Value_Output, [1872](#)
- textfgcolor
 - FI_Terminal, [1616](#)
- textfgcolor_default
 - FI_Terminal, [1616](#)
- textfgcolor_xterm
 - FI_Terminal, [1616](#)
- textfont
 - FI_Browser_, [494](#)

- FI_Input_, 970
 - FI_Menu_, 1043, 1044
 - FI_Terminal, 1617
 - FI_Text_Display, 1679
 - FI_Value_Input, 1861, 1862
 - FI_Value_Output, 1872
- textsize
 - FI_Browser, 469
 - FI_Help_Dialog, 850
 - FI_Input_, 970
 - FI_Menu_, 1044
 - FI_Terminal, 1617
 - FI_Text_Display, 1679
 - FI_Value_Input, 1862
 - FI_Value_Output, 1872
- The Wayland backend for its developer, 175
- thread_message
 - Multithreading support functions, 312
- ticks_between
 - FI, 395
- ticks_since
 - FI, 395
- time_to_wait
 - FI_Timeout, 1739
- tis620.h, 2661
- title
 - FI_Native_File_Chooser, 1137, 1138
- Todo List, 203
- tooltip
 - FI_Widget, 1922
- TOOLTIP_WINDOW
 - FI_Widget, 1891
- top_row
 - FI_Table, 1537, 1538
- top_window
 - FI_Widget, 1922
- top_window_offset
 - FI_Widget, 1922
- topline
 - FI_Browser, 470
 - FI_Help_View, 866
- transcoding_warning_action
 - FI_Text_Buffer, 1637
- translate
 - FI_Copy_Surface, 607
 - FI_EPS_File_Surface, 648
 - FI_Image_Surface, 935
 - FI_PDF_File_Surface, 1194
 - FI_PostScript_File_Device, 1229
 - FI_Printer, 1260
 - FI_SVG_File_Surface, 1489
 - FI_Widget_Surface, 1930
- tree
 - FI_Tree_Item, 1830
- Type
 - FI_File_Chooser, 669
 - FI_Native_File_Chooser, 1135
- type
 - FI_File_Icon, 676
 - FI_Label, 1003
 - FI_Spinner, 1471, 1472
 - FI_Table_Row, 1554
 - FI_Widget, 1923
- typea
 - FI_Multi_Label, 1111
- typeb
 - FI_Multi_Label, 1111
- u8c_disp_row
 - FI_Terminal, 1617
- u8c_hist_row
 - FI_Terminal, 1618
- u8c_hist_use_row
 - FI_Terminal, 1618
- u8c_ring_row
 - FI_Terminal, 1618
- ucs2be.h, 2662
- ucs2fontmap.c, 2669
- un_maximize
 - FI_Window, 1962
- uncache
 - FI_Anim_GIF_Image, 422
 - FI_Bitmap, 428
 - FI_Image, 928
 - FI_Pixmap, 1200
 - FI_RGB_Image, 1319
 - FI_Shared_Image, 1419
- uncheck
 - FI_Menu_Item, 1081
- UNDERLINE
 - FI_Terminal, 1598
- undo
 - FI_Input_, 972
 - FI_Text_Buffer, 1635
- Unicode and UTF-8 functions, 318
 - ERRORS_TO_CP1252, 320
 - ERRORS_TO_ISO8859_1, 320
- fl_access, 320
- fl_chdir, 321
- fl_chmod, 321
- fl_close_fd, 321
- fl_fopen, 322
- fl_getcwd, 322
- fl_getenv, 322
- fl_make_path, 323
- fl_make_path_for_file, 323
- fl_mkdir, 323
- fl_nonspacing, 324
- fl_open, 324
- fl_open_ext, 324
- fl_putenv, 325
- fl_rename, 325
- fl_rmdir, 325
- fl_stat, 326
- fl_system, 326
- fl_ucs_to_Utf16, 326
- fl_unlink, 327

- [fl_utf8back](#), 327
 - [fl_utf8bytes](#), 327
 - [fl_utf8decode](#), 328
 - [fl_utf8encode](#), 328
 - [fl_utf8from_mb](#), 328
 - [fl_utf8froma](#), 329
 - [fl_utf8fromwc](#), 329
 - [fl_utf8fwd](#), 329
 - [fl_utf8len](#), 329
 - [fl_utf8len1](#), 330
 - [fl_utf8locale](#), 330
 - [fl_utf8strlen](#), 330
 - [fl_utf8test](#), 330
 - [fl_utf8to_mb](#), 331
 - [fl_utf8toa](#), 331
 - [fl_utf8toUtf16](#), 331
 - [fl_utf8towc](#), 332
 - [fl_utf_nb_char](#), 332
 - [fl_utf_strcasecmp](#), 332
 - [fl_utf_strncasecmp](#), 333
 - [fl_utf_tolower](#), 333
 - [fl_utf_toupper](#), 333
 - [fl_wcwidth](#), 333
 - [fl_wcwidth_](#), 334
- [STRICT_RFC3629](#), 320
- [Unicode and UTF-8 Support](#), 104
- [UNKNOWN_ROOT_TYPE](#)
 - [Fl_Preferences](#), 1234
- [unlock](#)
 - [Multithreading support functions](#), 312
- [untranslate](#)
 - [Fl_Copy_Surface](#), 608
 - [Fl_EPS_File_Surface](#), 649
 - [Fl_Image_Surface](#), 935
 - [Fl_PDF_File_Surface](#), 1195
 - [Fl_PostScript_File_Device](#), 1229
 - [Fl_Printer](#), 1260
 - [Fl_SVG_File_Surface](#), 1489
 - [Fl_Widget_Surface](#), 1930
- [up_down_position](#)
 - [Fl_Input_](#), 972
- [update](#)
 - [Fl_Menu_Bar](#), 1057
 - [Fl_Shared_Image](#), 1419
 - [Fl_Sys_Menu_Bar](#), 1512
 - [Fl_Text_Selection](#), 1712
- [update_child](#)
 - [Fl_Group](#), 847
- [update_h_scrollbar](#)
 - [Fl_Text_Display](#), 1679
- [update_line_starts](#)
 - [Fl_Text_Display](#), 1680
- [update_menubutton](#)
 - [Fl_Input_Choice](#), 987
- [update_prev_next](#)
 - [Fl_Tree_Item](#), 1830
- [update_v_scrollbar](#)
 - [Fl_Text_Display](#), 1680
- [USE_FILTER_EXT](#)
 - [Fl_Native_File_Chooser](#), 1135
- [use_high_res_GL](#)
 - [Fl](#), 396
- [USER](#)
 - [Fl_Preferences](#), 1235
- [user_data](#)
 - [Fl_Widget](#), 1923
- [USER_L](#)
 - [Fl_Preferences](#), 1235
- [userdeicon](#)
 - [Fl_Tree_Item](#), 1831
 - [Fl_Tree_Prefs](#), 1839
- [USERFLAG1](#)
 - [Fl_Widget](#), 1891
- [USERFLAG2](#)
 - [Fl_Widget](#), 1891
- [USERFLAG3](#)
 - [Fl_Widget](#), 1891
- [usericon](#)
 - [Fl_Tree](#), 1810
 - [Fl_Tree_Item](#), 1831
- [Using OpenGL](#), 90
- [utf8.h](#), 2662
- [utf8_internal.h](#), 2317
- [utf8Utils.c](#), 2673
- [valid](#)
 - [Fl_Anim_GIF_Image](#), 422
 - [Fl_Gl_Window](#), 789
- [value](#)
 - [Fl_Browser](#), 470
 - [Fl_Button](#), 506
 - [Fl_Choice](#), 571
 - [Fl_Clock_Output](#), 590, 591
 - [Fl_Color_Chooser](#), 604
 - [Fl_File_Chooser](#), 670, 671
 - [Fl_File_Input](#), 688
 - [Fl_Help_Dialog](#), 850
 - [Fl_Help_View](#), 866
 - [Fl_Input_](#), 972, 973
 - [Fl_Input_Choice](#), 987
 - [Fl_Menu_](#), 1044, 1045
 - [Fl_Menu_Item](#), 1082
 - [Fl_Progress](#), 1268
 - [Fl_Scrollbar](#), 1383, 1384
 - [Fl_Shortcut_Button](#), 1427, 1428
 - [Fl_Spinner](#), 1472
 - [Fl_Tabs](#), 1574
 - [Fl_Valuator](#), 1850
- [value_damage](#)
 - [Fl_Adjuster](#), 408
 - [Fl_Valuator](#), 1850
- [value_height](#)
 - [Fl_Value_Slider](#), 1882, 1883
- [value_width](#)
 - [Fl_Value_Slider](#), 1883
- [version](#)
 - [Fl](#), 396

- VERTICAL
 - FI_Browser_, 483
 - FI_Flex, 716
- VERTICAL_ALWAYS
 - FI_Browser_, 483
- vertical_label_margin
 - FI_Widget, 1923
- viscii.h, 2664
- visible
 - FI_Browser, 470
 - FI_Widget, 1923
- visible_cells
 - FI_Table, 1538
- VISIBLE_FOCUS
 - FI_Widget, 1891
- visible_focus
 - FI, 396
 - FI_Widget, 1924
- visible_r
 - FI_Tree_Item, 1831
 - FI_Widget, 1924
- visual
 - FI, 396
- vline_length
 - FI_Text_Display, 1680
- vposition
 - FI_Browser_, 494
 - FI_Tree, 1810
- vprintf
 - FI_Terminal, 1618
 - FI_Text_Buffer, 1636
- vsprintf.c, 2317
 - fl_vsprintf, 2318
- w
 - FI_Image, 928, 929
 - FI_Widget, 1924
- w_to_col
 - FI_Terminal, 1619
- wait
 - FI, 397
- wait_for_expose
 - FI_Window, 1962
- walk_selection
 - FI_Terminal, 1619
- warning
 - Common Dialog Classes and Functions, 350
- watch_widget_pointer
 - Safe widget deletion support functions, 315
- wayland.H, 2268
 - fl_wl_compositor, 2268
- when
 - FI_Table, 1538
 - FI_Widget, 1925
- which
 - FI_Tabs, 1576
- widget
 - FI_Grid, 827, 828
 - FI_Widget_Tracker, 1931
- win32.H, 2269
- window
 - FI_Widget, 1925
- window_menu_style
 - FI_Sys_Menu_Bar, 1512
- window_menu_style_enum
 - FI_Sys_Menu_Bar, 1508
- Windows handling functions, 232
 - atclose, 234
 - default_atclose, 233
 - first_window, 233
 - grab, 233
 - modal, 233
 - next_window, 234
 - set_atclose, 234
- word_end
 - FI_Input_, 974
 - FI_Text_Buffer, 1636
 - FI_Text_Display, 1680
- word_start
 - FI_Input_, 974
 - FI_Text_Buffer, 1636
 - FI_Text_Display, 1681
- wrap
 - FI_Input_, 974
 - FI_Spinner, 1472
- WRAP_AT_BOUNDS
 - FI_Text_Display, 1654
- WRAP_AT_COLUMN
 - FI_Text_Display, 1654
- WRAP_AT_PIXEL
 - FI_Text_Display, 1654
- wrap_mode
 - FI_Text_Display, 1681
- WRAP_NONE
 - FI_Text_Display, 1654
- wrap_uses_character
 - FI_Text_Display, 1681
- wrap_width
 - FI_Tooltip, 1761
- wrapped_column
 - FI_Text_Display, 1682
- wrapped_line_counter
 - FI_Text_Display, 1682
- wrapped_row
 - FI_Text_Display, 1684
- x
 - FI_Widget, 1926
- x.H, 2270
- x11.H, 2271, 2272
 - fl_x11_find, 2271
 - fl_x11_gc, 2271
 - fl_x11_use_display, 2271
 - fl_x11_xid, 2272
- x_to_col
 - FI_Text_Display, 1684
- xclass
 - FI_Window, 1963

Ximint.h, [2676](#)
Xlibint.h, [2676](#)
xposition
 FI_Scroll, [1373](#)
Xutf8.h, [2318](#)
xy_to_position
 FI_Text_Display, [1684](#)
xy_to_rowcol
 FI_Text_Display, [1685](#)

y
 FI_Widget, [1926](#)
yposition
 FI_Scroll, [1373](#)